Applied Intelligence (2019) 49:2807–2822
https://doi.org/10.1007/s10489-019-01423-6

Ensemble classification for imbalanced data based on feature space partitioning and hybrid metaheuristics

Pedro Lopez-Garcia[1,2] · Antonio D. Masegosa[1,2,3] · Eneko Osaba[1,2,4] · Enrique Onieva[1,2] · Asier Perallos[1,2]

Published online: 7 February 2019

Abstract

One of the most challenging issues when facing a classification problem is to deal with imbalanced datasets. Recently, ensemble classification techniques have proven to be very successful in addressing this problem. We present an ensemble classification approach based on feature space partitioning for imbalanced classification. A hybrid metaheuristic called GACE is used to optimize the different parameters related to the feature space partitioning. To assess the performance of the proposal, an extensive experimentation over imbalanced and real-world datasets compares different configurations and base classifiers. Its performance is competitive with that of reference techniques in the literature.

Keywords Ensemble classification · Imbalanced classification · Feature space partitioning · Hybrid metaheuristics

1 Introduction

The classification task is one of the most important and basic tasks in the field of machine learning. Many approaches to this task have been developed over the years. Some of the classical methods for this are decision trees, artificial neural networks, K-nearest neighbours, and support vector machines, among others. These techniques operate under the assumption that the data contains a faithful balance between each of the classes represented in the problem [13]. However, in many real-world problems, this assumption leads to poor performance when the number of instances of one class is much lower than those for the other classes. If this situation occurs, the dataset is said to be imbalanced. In these kind of datasets, the class with the largest number of instances is called the majority class, while a class with fewer instances is called a minority class. Imbalanced data is present in real-world problems, such as disease diagnosis [53], traffic congestion [43], astronomy [54] and image classification [65, 67]. When machine learning methods are applied to imbalanced data, they should focus on achieving a good classification of the minority class due to the fact that the cost of misclassifying them is usually higher [21, 72]. Using the traffic congestion forecasting problem as an example, it is more important to achieve a higher accuracy regarding instances of congestion (a minority class) than for instances of a normal state of traffic (the majority class) due to the loss of time this can involve for drivers in a real-world scenario.

Many approaches have been proposed in the literature to deal with learning from imbalanced data: e.g., sampling methods, cost-sensitive algorithms, one-class classifiers, and ensemble classification techniques. These approaches can be placed into three different categories:

– Data-level approaches, which are focused on restructuring the training datasets in order to balance them. Oversampling and undersampling methods are the most common examples of this category;

✉ Pedro Lopez-Garcia
p.lopez@deusto.es

Antonio D. Masegosa
ad.masegosa@deusto.es

Eneko Osaba
eneko.osaba@tecnalia.com

Enrique Onieva
enrique.onieva@deusto.es

Asier Perallos
perallos@deusto.es

[1] DeustoTech-Fundacion Deusto, Deusto Foundation, 48007, Bilbao, Spain

[2] Faculty of Engineering, University of Deusto, 48007, Bilbao, Spain

[3] IKERBASQUE, Basque Foundation for Science, 48011, Bilbao, Spain

[4] TECNALIA Research and Innovation, 48160 Derio, Spain

- Algorithmic level approaches, which introduce modifications in the classification methods to improve their performance when classifying the minority class; and
- Ensemble methods, which combine the estimation of a set of individual classifiers trained over the same data. The most widely used approach of this class is the so-called boosting algorithm, which works under the premise that a set of weak classifiers works better than a strong one. Examples of boosting algorithms are SMOTEBoost [9], RUSBoost [52], and AdaBoost [57].

In the present paper, we focus on ensemble methods because they have been shown to be one of the most successful approaches to deal with imbalance classification so far. However, the correct design of the ensemble plays a pivotal role in obtaining good performace and involves, among other aspects, decisions related to the selection of the base classifiers and the method for aggregating the output of the base classifiers [50]. This paper is centred on a particular approach to designing ensembles called AdaSS (Adaptive Splitting and Selection), which simultaneously divides the feature space into partitions and establishes a different classifier for each partition by adjusting the weights of the different base classifiers within the discriminant function of the collective decision-making method. AdaSS was proposed by [28] and recently used in other papers, such as [29, 30], with very promising results. This method for building ensembles entails the resolution of a complex optimization problem whose objective is the minimization of the error of the whole system.

The proposal presented in this paper is related to the use of ensemble methods based on AdaSS as a powerful tool to deal with imbalance datasets because, as far as we know, it has not been applied before in this context. The motivations behind this research are:

- To select the best area possible to create the partitions for each ensemble by optimizing the positions of the centroids of the clusters that delimit the partitions of the feature space.
- To optimize the weights of each base classifier within the discriminant function of the collective decision-making method of each ensemble. This optimization will be unsupervised. In this way, neither expert knowledge nor an external validation set will be necessary to determine the initial values of the weights. This also arranges that the classifiers work in a non-restrictive way, i.e. the final weights will be based on the level of expertise that the classifier had obtained along the execution in every class.
- To address the last two tasks (feature space centroids and individual weight optimization for each partition and ensemble, respectively) in one integrated process. This approach has been proven to obtain very good results in other papers, such as [28–30]. The main novelty of our proposal w.r.t. these publications is the incorporation of a more powerful method to solve the underlying optimization problem. This optimization problem becomes even more complex in the context of imbalanced data because the prediction of the majority classes leads to local minima with big basins of attraction, which is a characteristic that is known to lead optimization methods to perform more poorly. For this reason, in our opinion, the use of more advanced optimization algorithms is a must in order to ensure the best possible performance of the resulting ensemble. To this end, we have used a hybrid metaheuristic, called GACE, that combines a genetic algorithm (GA) with a cross entropy (CE) method for the resolution of the mentioned optimization problem. The main advantage of this technique is the combination of the exploration ability of the GA and the exploitation ability of the CE. This method was successfully applied to the optimization of hierarchical fuzzy rule-based systems in [43], and continuous functions in [42].
- To be able to deal with imbalanced data without the use of data-level methods such as SMOTE, which is currently one of the most successful ones. As mentioned before, data-level approaches to deal with imbalance necessitate the modification of the training set, which leads to an extra cost in terms of the time required for the application of the technique. Our aim is to design methods that provide similar or better results without this extra cost by avoiding modifications of the training set.

This paper is an extension of the work presented in [44]. The main novelties are:

- A wider and more realistic benchmark: the total number of datasets has increased from 12 to 40 by incorporating new imbalanced datasets. Furthermore, 10 out of 40 correspond to real-world datasets for traffic congestion prediction.
- A new analysis of the algorithm's behaviour: a study of the influence of the size of the sub-population on the performance of the ensemble method has been included.
- An extension of the comparative study: the proposal has been compared with new high-performance and well-known methods from the literature on imbalanced classification.

The rest of this paper is structured as follows. Section 2 reviews different publications related to ensemble methods and hybrid techniques with a special focus on its application to imbalanced datasets. The ensemble methodology based on AdaSS and GACE is exposited in Section 3. The experimental set-up is presented in Section 4. Lastly, Section 5 contains the conclusions and presents avenues for future research.

2 Background

In this section, different approaches in the literature related to our proposal are reviewed. We focus this section on research in the state of the art from the three different areas that form part of the problem and, the proposed solution is exposited in this paper: Section 2.1 contains a brief introduction to the definition of ensembles, their use, and different design approaches; ensemble methods applied not only to general themes but also to imbalanced datasets in Section 2.2; metaheuristics applied to imbalanced datasets in Section 2.3, and hybrid methods applied to both imbalanced and balanced datasets in Section 2.4.

2.1 Ensemble learning: general approach

Ensemble classification can be defined as the combination of a group of classifiers whose individual decisions are joined in some manner to provide a final output [33]. The principal idea behind the use of ensemble classification is to learn from data using multiple individual classifiers.

Generally, ensemble classification has proved to obtain better results than an individual classifier on its own, when they are applied to the same problems [11], and it has also been presented as a method to improve the performance of a single classifier [5, 11, 32, 71]. When an ensemble is created, there are some design decisions to make, such as the algorithm or algorithms to use as individual classifiers (also called the base classifiers), the sampling strategy, and the collective decision making method for the final output, i.e. the method for combining the outputs of the base classifiers. Other aspects to take into account could be the generation of diversity or the way each classifier will be trained (with the whole training set or a part of it). This area of research has attracted significant interest in recent years. The interested reader is referred to [50] for a tutorial on this topic, describing a taxonomy for characterizing ensemble methods, and the general process of constructing classification ensembles.

Focusing on the selection method used to choose the most appropriate individual classifiers given a classification problem, we can find two approaches: static classifier selection, where the same ensemble is applied to all test samples; and dynamic classifier selection, where a different ensemble may be applied to each test sample [19]. Within dynamic classifier selection, an interesting concept is the local specialization of the base classifiers on specific partitions of the feature space [37]. Some proposals in this direction assume the local specialization of the individual classifiers while others divide the feature space into partitions and establish a different classifier for each of them.

Regarding the choice of a collective decision-making method, two main groups of methods can be defined for this task. The first one includes algorithms that join the answers of their classifiers. Majority voting [51] and other kinds of popular voting variants [37, 39, 59] are part of this group. Advanced techniques include weighting the importance of the decisions coming from the base classifiers. Treating the process of weight selection as a separate learning process is an alternative method [22, 28, 38]. One of the advantages of these techniques is that they effectively counteract any overtraining of the base classifiers [28]. The second group is formed by the procedures that use a posteriori probability estimator to fuse classifiers at the level of their discriminating functions. These methods do not require a learning procedure. However, they can be only used in clearly defined conditions [15].

As mentioned in the introduction, the present paper is focused on the ensemble design method called AdaSS. According to the previous taxonomy, it uses a dynamic classifier selection method based on local specialization, since different individual classifiers are applied depending on the feature space partition they belong to; and weighted voting as the collective decision-making method.

2.2 Ensemble learning applied to imbalanced classification

Ensemble learning can be defined as the use of multiple learning algorithms to obtain better predictive performance than could be obtained from any of these algorithms alone [47, 50]. Over the last decade, much research related to this approach has been presented in the literature, focusing on the classification problem for imbalanced datasets. For example, in [49], a resampling ensemble algorithm is developed focused on the imbalanced classification problem. In this case, the minority classes are oversampled while the majority classes are undersampled. To construct the ensemble, machine learning methods are selected.

Another example can be found in [58], which presents a bagging technique where two learning algorithms are used to construct the ensemble to deal with an on-line imbalanced learning problem.

In [40], a resampling ensemble algorithm is developed focused on the classification problems for imbalanced datasets. The optimization technique used in this case is the BAT algorithm, where the accuracy rate of all the classes is optimized at the same time. From the experimental results, the system can be used to reduce the time complexity as well as enhance the accuracy rate of the imbalanced classification process.

Another ensemble-based method is presented in [17], where Synthetic Minority Over-sampling Technique (SMOTE) and Rotation Forest algorithm are used to address the class imbalance problem. Twenty KEEL imbalanced datasets are used in the experimentation, where the proposal is compared with different classification ensemble methods, such as SMOTE-Boost, SMOTE-Bagging, and SMOTE-random sub-space.

There are many papers related to this theme, which means that it is an active issue in the literature. For this reason, the state-of-the-art of ensemble imbalance classification is wide. Interested readers are referred to [33, 46], and [57] for different surveys of this issue.

2.3 Metaheuristics applied to imbalanced classification

Metaheuristic techniques have been used in many different fields over the last decades. This category includes algorithms such as Particle Swarmn Optimization (PSO) [34], Ant Colony Optimization (ACO) [14], Genetic Algorithm (GA) [25], Bat Algorithm (BA) [62], Data Gravitation Classification (DGC) [66], and others [68]. Focusing on classification, and especially on imbalanced problems, these methods have been widely used on their own as well as in combination with other techniques in the literature. In [60], PSO is proposed for omics data classification. The algorithm is designed to handle the different characteristics of omics data, such as high dimensionality, small sample size, and class imbalance.

For example, in [63] there is proposed an undersampling method based on ACO for an imbalanced problem involving DNA microarray data. The proposal is evaluated on four benchmark skewed DNA microarray datasets. It outperforms many other sampling approaches.

In [6] there is developed a cost-sensitive feature selection method using a type of GA called a chaos genetic algorithm. The evaluation function considers both the costs of acquiring each feature and the costs of misclassification, in the field of network security, weakening the influence of the many instances from the majority classes in large-scale datasets. The proposal is tested on a large-scale dataset of network security, using two kinds of classifiers: C4.5 and K-nearest neighbours.

Other paper related to this topic can be found in [45], which analyses the performance of evolving diverse ensembles using genetic programming for software defect prediction with imbalanced data.

2.4 Hybrid algorithms applied to imbalanced classification

In this subsection, different hybrid approaches in the literature are mentioned. Hybrid algorithms are a way of dealing with the weaknesses of the different methods combined and, at the same time, maximizing their strengths. These algorithms have been used in many and varied domains, such as medicine [24], scheduling optimization [41], transportation systems [43], and astronomy [53].

In the imbalanced classification field, many publications can be found that use hybrid algorithms to deal with this problem. For example, in [61], a PSO is proposed for dealing with the class imbalance problem in medical and biological data mining. A PSO is combined with multiple classifiers and a performance metric for evaluation fusion. The majority classes are ranked using multiple objectives according to their merit and then combined with the minority class to create a balanced dataset.

In [55] there is presented a soft-hybrid algorithm to improve classification performance. The hybrid algorithm is formed by different modified machine learning techniques whose results were combined at the end of an experimentation phase. Measures such as the true positive rate, the F-measure, and the G-mean were used as quality measures.

Another example can be found in [8], where a hybrid algorithm formed by a GA and an undersampling method is created to improve the accuracy of support vector machines on skewed datasets.

Lastly, in [3], the authors developed a hybrid Adaboost-SVM method using Gaussian Mixture Modeling (GMM) to investigate the effect of using GMM with the boosted SVM in a multi-class phoneme recognition problem with the aim of improving the classification of imbalanced data.

There are two main novelties in the present paper with respect to the methods reviewed in this and the previous subsections. Using feature space partitioning as the approach to build the ensemble for imbalanced datasets has not been used in this context before. GACE, the optimization method for the ensemble approach in the present paper, is a more powerful optimizer than those used in previous work on ensemble-based feature space partitioning. As mentioned before, the main motivation behind this is that the problem becomes harder when the datasets are imbalanced due to the big basins of attraction created by the majority classes.

The concept of feature space partitioning has been applied to imbalance classification in [35] and [36]. In [35], the feature space partitioning consists in clustering strategies, such as c-means or fuzzy c-means, and the weights of the base classifiers are based on a heuristic function that takes into account the Euclidean distance between the object and the boundary of the respective class. In [36], the feature space partitioning is based on random sub-spaces; the weights of the base classifiers are set in the same way as before. Here, the main difference from these two papers is the use of AdaSS as the feature space partitioning technique: its main advantage is that it simultaneously optimizes the partitions, assigns classifiers to the partitions, and determines the weights of the base classifiers for inferring the output class.

3 Description of the ensemble approach

In this section, we describe the different elements that make up the proposed approach. First, we describe the AdaSS algorithm for the simultaneous partitioning of the feature space and assignment of classifiers to the partitions (Section 3.1). Then, the details of the training algorithm based on the GACE hybrid metaheuristic will be presented in Section 3.2.

3.1 Description of the adaptive splitting and selection algorithm

The Adaptive Splitting and Selection Algorithm exploits the local competencies of given classifiers. Let us assume the feature space $\mathcal{X}$ is divided into a set of H clusters,

$$\mathcal{X} = \bigcup_{h=1}^{H} \hat{\mathcal{X}}_h, \quad \forall k, l \in \{1, ..., H\}, \quad k \neq l, \quad \hat{\mathcal{X}}_k \cap \hat{\mathcal{X}}_l = \varnothing \tag{1}$$

where $\hat{\mathcal{X}}_h$ denotes the h-th constituent (cluster). The clusters are defined by their centroids $C_h = \{c_h^1, \ldots, c_h^d\}$, where d is the feature space dimension. With this information we define:

$$\text{member}(C, x) = \arg\min_{h=1}^{H} dist(x, C_h) \tag{2}$$

as the function that returns the index of the cluster where $C = \{C_1, \ldots, C_h\}$ is the set of centroids and $dist$ refers to the Euclidean distance. In the case of a draw, the cluster with the lower index is selected. Then, the decision rule for the combined classifier Ψ is given by the formula

$$\Psi(x) = \bar{\Psi}_{member(C, x_n)}(x_n) \tag{3}$$

where $\bar{\Psi}_h$ is the classifier assigned to the h-th cluster (called an area classifier). In this way, the compound classifier returns the output of the classifier assigned to the cluster to which the instance x belongs. It could be a single classifier or an ensemble classifier, which is the case of our proposal. It is important to take into account that the parameter H plays a fundamental role in the performance of the ensemble. On the one hand, a larger number of clusters makes possible a wider exploration of the local competencies of the area classifiers, but on the other hand, it could lead to overfitting. In the present paper, the parameter H is kept fixed throughout the experimentation.

We will now present the classification rules for the area or local classifiers $\bar{\Psi}_h$, which in turn are also ensembles. Let us assume that we have k (base) classifiers $\Psi^1, \Psi^2, \ldots, \Psi^k$ with which to build these local (ensemble) classifiers (note that we use $\bar{\Psi}$ with a subscript to refer to the local classifier and Ψ with a superscript to refer to the base classifiers that we use to build the local classifiers). For a given instance $x \in X$, each local classifier decides, based on a discriminant function, whether x belongs to class $i \in M = \{1, \ldots, m\}$. Let $F^{(l)}(i, x)$ denote a function that is assigned to class i for a given value of x, and that is used by the l-th classifier. To calculate the response of each of the classifiers, a matrix W is defined that represents the weights used for the discriminant function F. This matrix has k rows and m columns of values, where k is the number of classifiers used in the ensemble and m is the number of classes in the dataset. In this way, the weight matrix corresponding to the h-th cluster W_h can be formulated as follows:

$$W_h = [[w_h^1(1), \ldots, w_h^1(m)], \ldots, [w_h^k(1), \ldots, w_h^k(m)]] \tag{4}$$

Having said this, the local classifier Ψ_h uses the next decision rule:

$$\bar{\Psi}_h(x) = i \text{ if } \hat{F}_h(i, x) = max_{j \in M} \hat{F}_h(j, x) \tag{5}$$

where

$$\hat{F}_h(i, x) = \sum_{l=1}^{k} w_h^l(i) F^{(l)}(i, x), \tag{6}$$

and

$$\sum_{l=1}^{k} w_h^l = 1, \forall i \in M \tag{7}$$

Finally, let us assume that for the training of the classifier we have a learning set LS, that consists in N learning objects. Then, LS is defined as

$$LS = (x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N) \tag{8}$$

where x_n denotes the values for the nth object, and y_n denotes its correct class label. As usual, LS is divided into two subsets: the training set $|TS| = T$, used during training, and the validation set $|VS| = N - T$. The optimization criteria for the global combined classifier is

$$Q(\Psi) = \hat{Q}(\Psi, TS) \tag{9}$$

where $\hat{Q}$ refers to a specific performance metric of the classifier Ψ in the training set TS (e.g. accuracy, area under the ROC curve, etc.) that is defined according to the user's preferences. In the next subsection, we explain the description and workflow of the training algorithm based on the hybrid metaheuristic GACE.

3.2 Description of the training algorithm

The objective of the training algorithm is to learn the best combination of cluster centroids $C = \{C_1, \ldots, C_H\}$ and ensemble weights $W = \{W_1, \ldots, W_H\}$ that minimizes the objective function described in (9), given a set of base classifiers Ψ^1, Ψ^2, ..., Ψ^k. To solve this optimization problem, we used the GACE method, as mentioned before. GACE is a hybrid algorithm that combines a genetic algorithm with cross entropy in order to take advantage of the exploratory ability of GA as a search algorithm and the exploitation capability of CE, creating a synergy between them. The benefits of GACE as an optimization method are supported by its good results in topics such as the optimization of hierarchical fuzzy rule-based systems [43] or continuous functions [42].

The general working of the training method is as follows (its pseudocode is given in Algorithm 1). First, the initial population (POP), with POP_{size} individuals, is randomly generated following the structure of the codification of the solution. In each generation, the population is then divided into two sub-populations, GA_{pop} and CE_{pop}, with GA_{size} and CE_{size} individuals ($POP_{\text{size}} = GA_{\text{size}} + CE_{\text{size}}$), respectively. The individuals of GA_{pop} are chosen using the corresponding selection operator, while the individuals in CE_{pop} are the CE_{size} best individuals in the current population POP_t. The crossover and mutation operators of the GA are applied to GA_{pop}, while the CE method is used to evolve the corresponding sub-population in CE_{pop}. Both sub-populations of new individuals are joined into a single population that then completely replaces the previous one. This process is iteratively repeated until a specified stop condition is reached. Interested readers are referred to [43] for more information about the hybrid algorithm.

Algorithm 1 Pseudocode of the workflow followed by the optimization method GACE

```
Data: $POP_{size}, p_{ga}, p_c, p_m, L_r, p_{up}, T_{max}$
Result: $Best\ individual\ found$
$GA_{size} \leftarrow \| POP_{size} \cdot p_{ga} \|$
$CE_{size} \leftarrow POP_{size} - GA_{size}$
$n_{up} \leftarrow \| CE_{size} \cdot p_{up} \|$
$t \leftarrow 0$
$POP_0, \leftarrow$ Initialize($POP_{size}$)
$\overline{M} \leftarrow$ Initialize Means vector
$S \leftarrow$ Initialize Standard Deviation vector
Evaluate $POP_0$
while $t < T_{max}$ do
    $GA_{pop} \leftarrow$ SelectionOperator($POP_t, GA_{size}$)
    $CE_{pop} \leftarrow$ SelectBestSamples($POP_t, CE_{size}$)
    Offspring$_{GA}$ $\leftarrow$ Crossover($GA_{pop}, p_c$)
    Offspring$_{GA}$ $\leftarrow$ Mutation(Offspring$_{GA}$, $p_m$)
    Offspring$_{CE}$ $\leftarrow$ Generate($CE_{pop}, CE_{size}, \overline{M}, S$)
    $\overline{M} \leftarrow UpdateMeans(L_r, \overline{M}$, Offspring$_{CE}$, $n_{up})$
    $S \leftarrow UpdateDeviation(L_r, S$, Offspring$_{CE}$, $n_{up})$
    $POP_{t+1} \leftarrow$ Offspring$_{GA} \bigcup$ Offspring$_{CE}$
    Evaluate $POP_{t+1}$
    Add the best individual found to $POP_{t+1}$ if it is
    not in the population
    $t \leftarrow t + 1$
end
```

In the next part of this subsection, we will explain the codification used for the solutions, the initialization of the population, and the specific crossover and mutation operators employed.

3.2.1 Codification of the solution

In a formal way, one individual in the population is composed of two different parts: one of them codifies the centroids of the partitions (C), and the other one contains the weights used in the discriminant functions of the ensemble classifiers (W). Figure 1 shows an individual with the described structure.

The part codifying the centroids C is represented as an array of H elements, where H is the number of areas or partitions defined by the user. Each centroid c_h, where $h \in [1 \cdots H]$ is the index of the partition, is a vector with the same number of elements as the dimension of the dataset d.

As mentioned in previous sections, there are H matrices contained in W. Each of these matrices, W_h, $h \in [1 \cdots H]$, has $k \times m$ entries, where k is the number of classifiers used in the ensemble and m is the number of classes in the dataset, as mentioned before. Each entry is the weight

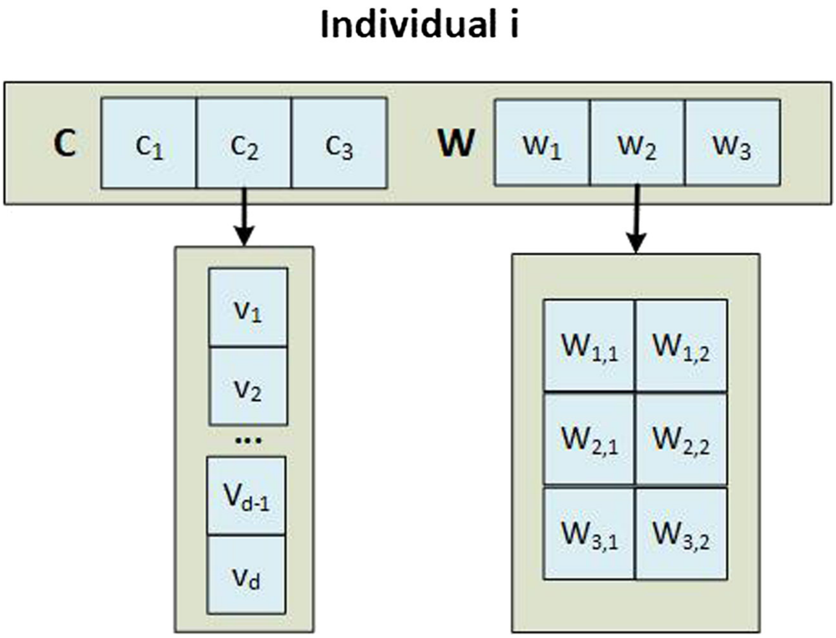

Fig. 1 Structure of an individual in the population with $H = 3$ areas, $k = 3$ classifiers and $m = 2$ classes

in discriminant function of the classifier l to determine the class i of an instance assigned to area h. For example, $w_3[1, 2]$ is the weight for the first classifier to determine the second class in the third partition. Each of the possible solutions is represented as shown in (10).

$$Ind(C, W) = \begin{cases} C = (C_1, C_2, \ldots C_H) \\ W = (W_1, W_2, \ldots W_H) \end{cases} \tag{10}$$

Here, $c_h = (v_1, v_2, \ldots v_d)$ and $W_h = \{W_h[1, 1], \ldots W_h[k, m]\}$. Then, GACE is applied to achieve the following goals:

1. Tuning the position of the different centroids C in the feature space.
2. Adjust the values of the weight matrices W, for the different classifiers and classes.

3.2.2 Initialization of the population

For the initial population, each value in C_h is initialized with a random value in the interval $[\min_r, \max_r]$, where $\max_r$ and $\min_r$ are the upper and lower bounds of the r-th dimension of the feature space.

For the weights, each value of the matrices is initialized randomly in the interval [0, 1] and then normalized to ensure that, for each class, they sum to one. As mentioned, each C_h has a size equal to the number of the variables in the feature space, and each W_h has a size of k classifiers per m classes.

3.2.3 Operators of the sub-populations

Different operators are applied to each sub-population. Selection, crossover, and mutation operators are used in the case of GA_{pop}. As selection operator, tournament selection [20] has been adopted. This operator chooses two random individuals in the population and selects the best one according to their fitness. A total of GA_{size} individuals are chosen by this operator to form the GA_{pop} sub-population. The crossover operator chosen was BLX-α [16]. Given two parents $X = (x_1 \ldots x_z)$ and $Y = (y_1 \ldots y_z)$, for each element i, BLX-α crossover creates two offspring by generating random values in the interval shown in (11), with $\alpha \in [0, 1]$. The choice of this crossover is justified due to its good synergy between exploration and exploitation of the individual [23].

$$[min(x_r, y_r) - \alpha|x_r - y_r|,\ \max(x_r, y_r) + \alpha|x_r - y_r|] \tag{11}$$

Gaussian mutation [4] is taken as the mutation operator. Each element x_i of an individual is updated according to (12):

$$a_r = \mathcal{N}(x_r, \frac{\max_r - \min_r}{10}) \tag{12}$$

where $\mathcal{N}$ is a normal distribution with mean x_r and standard deviation $(max_r$ - $min_r)$.

4 Experimentation

This section presents the results of the experimentation carried out. The objectives of this experimentation are listed below:

- To validate the performance of the proposed ensemble classification approach based on AdaSS and GACE on complex imbalanced datasets and real-world problems.
- To analyse the influence of the algorithm used to generate the base classifiers and the sizes of the sub-populations of GACE in the performance of the proposal.
- To compare the approach with different state-of-the-art algorithms in imbalanced classification.

This section is structured as follows. In Section 4.1, the different datasets and their main characteristics are presented. The parameter settings and base classifiers are presented in Section 4.2. The analysis of the results and the comparison versus the state-of-the-art are presented in Section 4.3.

4.1 Datasets

A total of 30 imbalanced datasets of different degrees of complexity have been extracted from the KEEL repository[1] in order to test the performance of the proposal in different kinds of scenarios. The datasets chosen have been used extensively in the literature. Table 1 shows the characteristics of each dataset: name, number of instances, features, classes, and imbalance ratio (IR) [69, 70], which is the ratio between the number of instances from the majority and minority classes. The larger the ratio is, the more imbalanced the dataset. The number of classes in these imbalanced datasets is two, which means that we are dealing with imbalanced binary classification. These classes are defined as positive (minority class) and negative (majority class).

In addition, real-world datasets have been used in order to apply the proposal to traffic congestion forecasting in a road; the data collected comes from Lisbon highway A5 and was used in EU FP7 project ICSI.[2] This highway is a 25-km long motorway in Portugal that connects Lisbon with Cascais. Data from a total of 10 sensors in the road have been transformed into datasets, and the proposal has been applied to forecast the congestion in each one. Each dataset contains 9 variables: day of the week, hour of the day, number of motorbikes, number of cars, number of trucks, number of buses, number of other types of vehicles, total number of vehicles, and a class called *nextlevel*. This class contains the value of congestion that appears in the next hour at a certain point and can take as values $\{LOW, MED, HIG\}$. The level of congestion is defined to be LOW if the total number of vehicles counted is below the 15th percentile, MED (Medium) if it is above the 15th percentile but below the 30th, and HIG (High), otherwise. In the present paper, HIG is the positive class (minority class), and LOW and MED will form the negative class (majority class). The three first weeks of the month were used as the training set and the last week as the test set. This group of datasets will be referred to as A5-Traffic in the following sections.

Table 1 Details of imbalanced datasets used in the experimentation

No.	Name	Objects	Features	IR
1	Ecoli1	220	7	3.36
2	Ecoli3	336	7	8.6
3	Glass1	214	9	1.82
4	Glass6	214	9	6.28
5	Iris0	150	4	2
6	Page-blocks0	5472	10	8.79
7	Pima	768	8	1.87
8	Vehicle1	846	18	2.9
9	Yeast1	1484	8	2.46
10	Yeast3	1484	8	8.1
11	Glass016vs2	192	9	10.29
12	Ecoli4	336	7	14.3
13	Glass016v5	184	9	19.44
14	Glass5	214	9	22.78
15	Dermatology6	358	34	16.9
16	Shuttle6	230	9	22
17	Poker9	244	10	29.5
18	Yeast28	482	8	23.1
19	Yeast4	1484	8	28.1
20	Led7digit	443	7	10.97
21	Ecoli0137	281	7	39.14
22	WineRed8	656	11	35.44
23	WineWhite9	168	11	32.6
24	Yeast6	1484	8	41.4
25	Poker896	1485	10	58.4
26	WineWhite395	1482	11	58.28
27	Shuttle25	3316	9	66.67
28	WineRed35	691	11	68.1
29	Poker895	2075	10	82
30	Poker86	1477	10	85.88

4.2 Parameter settings

This section presents the parameter settings for the experimentation and the definition of the base classifiers.

[1] http://sci2s.ugr.es/keel/datasets.php

[2] http://www.ict-icsi.eu/

Three different algorithms have been used for creating the baseline classifiers:

- Minimal distance classifier, which applies the 3-nearest neighbours (3-NN) algorithm [10].
- A neural network (NN) method [27], trained with back-propagation algorithm. The number of neurons depends on the dataset used: the size of the input layer is equal to the number of features. The size of the output layer is equal to the number of classes. The number of neurons in the hidden layer is equal to one-half of the sum of the numbers of neurons in the input and output layers. In this case, the total number of iterations was set to 2000 in order to have a fair comparison and not take so much time for the bigger datasets.
- Support vector machine (SVM) classifier [7], using the sequential minimal optimization procedure with a polynomial kernel.

A homogeneous pool was used in this experimentation, that is, all the base classifiers in the ensemble are built with the same algorithm. To induce diversification, each classifier is trained with a subset of $1/k$-th of the instances from the training set, where $k = 3$ is the number of classifiers in the pool. Each subset is mutually exclusive from each other and contains the same distribution of examples as the training set. In this way, each ensemble in the experimentation created with our proposal will have three base classifiers generated with the same algorithm (3-NN, NN or SVM) each one trained with 1/3 of the instances of the dataset. Focusing on the parameters of the algorithm used in the experimentation, the population size (POP_{size}) has been set to 50, and the size of the GA sub-population to 40 or 45 individuals, $GA_{\text{size}} = \{40, 45\}$. The reason for setting POP_{size} to this value is because of the good performance shown in other classification and optimization tasks [42]. Besides, in those papers, a population with a higher value of GA_{size} than the size of the CE sub-population (CE_{size}) tended to show better results [43]. As for the GA part parameters, the crossover probability p_c was set to 0.85 and the mutation probability p_m to 0.1. Regarding the CE parameters, the learning rate value L_r is usually recommended to be set within the interval [0.7, 0.9]. In this case, $L_r = 0.7$ was chosen. The parameter n_{up}, i.e. the number of individuals that is used to update the CE means and standard deviations, was set to $n_{up} = 0.4 \times CE_{\text{size}}$. The number of partitions H is the same as in the present authors' previous paper, and was set to $H = 3$. The number k of classifiers in each pool was also set to 3. The stop condition was designed as follows: in the first place, it checks if the best solution has not changed for 20 generations, and if so, the execution is stopped. Otherwise, it checks if a maximum number of generations $T_{max} = 200$ is fulfilled, stopping the execution in that case. A summary of the parameter settings is presented in Table 2.

4.3 Results

This section presents the results obtained by the proposal using the different configurations mentioned before. A broad comparison of these results with those obtained by state-of-the-art techniques from the literature is made. The following classification techniques from the literature have been used for this comparison:

- RUSBoost (RUS) [52] removes instances from the majority class by randomly undersampling the dataset in each iteration. After training a classifier, the weights of the original dataset instances are updated, and then another sampling phase is applied.
- UnderBagging to OverBagging (UOBag) [56] makes use of both oversampling and undersampling. One of the key points of this algorithm is that the diversity is boosted using a resampling rate in each iteration. This rate defines the number of instances taken from each class. Hence, the first classifiers are trained with a smaller number of instances than the last ones.
- Class and Prototype weighted classifier (CPW) [48] is a method of extracting weights associated to prototypes and classes, with the aim of enhancing the classification accuracy of the 1-NN rule.
- Adaboost [31] is a boosting algorithm which repeatedly invokes a learning algorithm to successively generate a committee of simple, low-quality classifiers.
- FARCHD [1] is a three-stage fuzzy association rule-based classification model which aims to obtain an accurate and compact fuzzy rule-based classifier with a low computational cost.
- Multilayer perceptron for cost-sensitive classification problems (NNCS) [64] uses a multilayer perceptron to classify, with minimal cost, a dataset of examples.

Table 2 Values of the parameters used in the experimentation

Parameter	Symbol	Values
Population size	POP_{size}	50
GA pop. size	GA_{size}	{40, 45}
CE pop. size	CE_{size}	{10, 5}
Crossover probability	p_c	0.85
Mutation probability	p_m	0.1
Learning rate	l_r	0.7
No. updates	n_{up}	0.4 CE_{size}
No. of areas	H	3
No. of classifiers	k	3
No. of iterations	T_{max}	200

Table 3 AUC obtained by the techniques on imbalanced datasets with IR less than 10

	Ecoli1	Ecoli3	Glass1	Glass6	Iris0	Page-blocks0	Pima	Vehicle1	Yeast1	Yeast3
$AdaSSGACE_{KNN.40}$	**0.890**	**0.864**	0.750	0.889	**0.999**	0.931	0.703	0.673	0.700	**0.897**
$AdaSSGACE_{NN.40}$	0.833	0.734	0.647	**0.903**	0.970	0.880	**0.738**	**0.778**	**0.711**	0.890
$AdaSSGACE_{SVM.40}$	0.872	0.785	0.639	0.641	0.841	0.751	0.589	0.651	0.688	0.891
$AdaSSGACE_{KNN.45}$	0.223	0.202	0.468	0.425	0.140	0.930	0.699	0.684	0.701	0.897
$AdaSSGACE_{NN.45}$	0.799	0.739	0.649	0.905	0.963	0.883	0.733	0.766	0.710	0.889
$AdaSSGACE_{SVM.45}$	0.875	0.761	0.635	0.647	0.910	0.757	0.577	0.652	0.681	0.892
RUS	**0.884**	0.840	**0.780**	**0.921**	0.990	**0.956**	0.725	**0.786**	0.701	**0.919**
UOBag	0.876	**0.886**	0.739	0.901	0.970	**0.953**	0.730	0.745	**0.720**	**0.919**
CPW	0.812	0.747	**0.775**	0.871	**1.000**	0.871	0.664	0.628	0.664	0.827
Adaboost	0.843	0.799	0.613	0.880	**1.000**	0.840	**0.742**	0.702	0.599	0.805
FARCHD	0.857	0.753	0.718	0.894	**1.000**	0.754	0.704	0.624	0.671	0.854
NNCS	0.854	0.856	0.607	0.851	**1.000**	0.736	0.727	0.660	0.677	0.743

It is important to note that some of the compared techniques use a pre-processing algorithm to modify the data before its execution. RUS uses SMOTE, and UOBag applies resampling to the data before the application of C4.5 as a base classifier. The techniques mentioned above have been included to determine whether the performance of the proposal of this paper, which we will refer to as AdaSSGACE, reaches or exceeds that of state-of-the-art techniques that do use pre-processing techniques.

KEEL [2] has been used for running the state-of-the-art techniques, and MATLAB r2017 using PRTools Toolbox[3] for the execution of AdaSSGACE. The experiments were carried out on an Intel Xeon E5 2.30 GHz computer with 32 GB of RAM. For validation, 5-fold cross-validation was used. The number of repetitions made for each method was set to 10. The performance metric that we used to set the function $\hat{Q}(\Psi, TS)$ defined in (9) is the Area Under the ROC Curve (AUC), which is calculated as in (13):

$$AUC = \frac{1 + TP_{rate} - FP_{rate}}{2} \tag{13}$$

where TP_{rate} and FP_{rate} correspond to the true positive ratio and the false positive ratio, respectively. This metric is used in order to compare imbalanced datasets in a fair way. It indicates the central tendency of the results obtained by each method. The configurations of the proposed technique are denoted by $AdaSSGACE_{BC,GA_{\text{size}}}$, where BC (either K-NN, NN, or SVM) is the algorithm used to generate the three base classifiers of the ensemble, and GA_{size} is the size of the GA sub-population. In this way, $AdaSSGACE_{KNN.40}$ indicates that this ensemble has three base classifiers generated with KNN and the GA_{size} parameter was set to 40.

Table 3 shows the results obtained by the techniques in the imbalanced datasets with IR less than 10. Bold values represent the two best AUC values obtained on the corresponding dataset. The most remarkable configurations of the proposal are those formed by the couple (K-NN, 40) and by SVM with both population sizes. In the case of the techniques from the state-of-the-art, RUS and UOBag are the two techniques with the best results, with similar values which, in turn, are similar to those obtained by $AdaSSGACE_{KNN.40}$.

Table 4 shows the results obtained for those datasets with IR between 10 and 30. As in the previous table, bold values represent the two best AUC values obtained on each dataset. In this case, those configurations that used K-NN to generate the base classifiers obtain, on average, better results than the rest of AdaSSGACE configurations, although the NN configurations also get similar results. As for the state-of-the-art techniques, RUS and UOBag continue obtaining good performance on almost every dataset. Also remarkable is the performance obtained by the FARCHD technique, which improves the values obtained when it was applied to those datasets with IR less than 10.

Lastly, Table 5 contains the results obtained by the techniques in those datasets with IR greater than 30. As in previous cases, K-NN configurations lead to better performance. Following the results obtained in the previous datasets, the RUS technique obtains the best results among the state-of-the-art techniques. In this case, FARCHD improves the results obtained by UOBAG, placing itself as the second best state-of-the-art technique in this case. While the results of the proposal configurations are close, equal to, or better than the results obtained by the best techniques in most cases, in some datasets, such as Poker896, they are far from the best results. This may be due to the fact that in datasets with high IR, a bagging or boosting method significantly improves the obtained results.

The AUC obtained by the techniques on the A5-Traffic datasets are collected in Table 6. A total of 10 real-data

[3] prtools.org/prtools/prtools-overview/

Table 4 AUC obtained by the techniques on imbalanced datasets with IR in the interval [10,30]

	Dermatology	Ecoli4	Glass016vs2	Glass016v5	Glass5	Led7digit	Poker9	Shuttle6	Yeast28	Yeast4
$AdaSSGACE_{KNN.40}$	0.938	**0.933**	**0.708**	0.867	0.804	0.851	0.740	0.960	**0.796**	0.798
$AdaSSGACE_{NN.40}$	**0.966**	0.760	0.630	0.796	0.718	0.856	0.563	0.920	0.720	0.739
$AdaSSGACE_{SVM.40}$	0.749	0.907	0.611	0.733	0.675	0.785	0.636	0.843	0.737	0.509
$AdaSSGACE_{KNN.45}$	0.946	**0.940**	0.678	0.863	0.785	0.845	0.705	**0.965**	**0.801**	**0.799**
$AdaSSGACE_{NN.45}$	0.979	0.805	0.594	0.832	0.764	0.839	0.586	0.894	0.711	0.709
$AdaSSGACE_{SVM.45}$	0.768	0.917	0.592	0.697	0.724	0.785	0.628	0.833	0.742	0.514
RUS	**0.966**	0.896	**0.700**	**0.954**	0.949	**0.894**	0.590	0.902	0.747	**0.827**
UOBag	0.938	0.867	0.629	**0.963**	**0.988**	0.881	0.556	0.948	0.778	0.763
CPW	0.500	0.870	0.577	0.836	0.893	0.500	**0.950**	0.900	0.769	0.677
Adaboost	0.500	0.842	0.494	0.891	**0.995**	**0.910**	0.572	0.900	0.770	0.548
FARCHD	**0.949**	0.872	0.491	0.789	0.745	0.883	**0.848**	**1**	0.700	0.565
NNCS	0.893	0.660	0.471	0.880	0.995	0.647	0.604	0.813	0.652	0.543

Table 5 AUC obtained by the techniques on imbalanced datasets with IR greater than 30

	Ecoli0137	Poker895	Poker896	Poker6	Shuttle25	WineRed35	WineRed8	WineWhite395	WineWhite9	Yeast6
$AdaSSGACE_{KNN.40}$	0.790	**0.631**	0.618	0.528	**0.986**	0.590	0.589	0.535	0.608	**0.875**
$AdaSSGACE_{NN.40}$	0.848	0.507	0.572	0.576	0.874	0.559	0.588	0.558	0.598	0.772
$AdaSSGACE_{SVM.40}$	0.682	0.588	0.557	0.469	0.672	0.580	0.528	0.531	0.576	0.515
$AdaSSGACE_{KNN.45}$	0.814	**0.629**	0.623	0.526	0.982	0.608	0.578	0.535	0.645	**0.868**
$AdaSSGACE_{NN.45}$	0.848	0.507	0.586	0.551	0.912	0.579	0.588	0.561	0.580	0.757
$AdaSSGACE_{SVM.45}$	0.685	0.596	0.483	0.494	0.642	0.597	0.535	0.560	0.566	0.524
RUS	**0.896**	0.547	**0.915**	0.631	**1**	**0.644**	**0.815**	**0.674**	**0.893**	0.851
UOBag	0.867	0.618	0.534	0.584	**1**	**0.615**	**0.700**	**0.576**	0.714	0.814
CPW	0.870	0.517	0.504	0.504	**1**	0.494	0.541	0.537	**0.894**	0.734
Adaboost	0.842	0.495	0.735	**0.864**	0.929	0.547	0.528	0.599	0.685	0.598
FARCHD	**0.872**	0.500	**0.960**	**0.900**	**1**	0.499	0.498	0.500	0.788	0.599
NNCS	0.660	0.529	0.503	0.451	0.590	0.583	0.664	0.444	0.418	0.686

Table 6 AUC obtained by the techniques on A5-Traffic datasets

	CL_{400}	CL_{600}	CL_{1505}	CL_{1980}	CL_{3600}	CL_{4000}	CL_{6800}	CL_{7100}	CL_{8050}	CL_{9400}
$AdaSSGACE_{KNN.40}$	0.913	0.869	0.945	**0.956**	0.881	0.945	0.810	0.922	0.961	0.845
$AdaSSGACE_{NN.40}$	0.925	0.846	**0.966**	**0.956**	0.887	0.956	0.809	0.946	0.961	0.905
$AdaSSGACE_{SVM.40}$	0.916	0.785	0.927	0.953	0.707	0.952	0.817	0.908	**0.963**	0.697
$AdaSSGACE_{KNN.45}$	0.909	0.868	0.949	0.954	0.873	0.947	0.806	0.923	0.960	0.850
$AdaSSGACE_{NN.45}$	0.915	0.828	**0.969**	0.953	0.891	0.954	0.828	0.940	0.959	0.886
$AdaSSGACE_{SVM.45}$	0.916	0.803	0.931	**0.957**	0.644	0.953	0.812	0.860	0.961	0.741
RUS	**0.975**	**0.978**	0.952	0.949	**0.940**	**0.961**	**0.889**	0.950	0.951	0.663
UOBag	**0.997**	**0.966**	**0.966**	0.949	0.922	**0.958**	**0.889**	0.937	**0.965**	0.825
CPW	0.967	0.868	0.965	0.500	0.852	0.500	**0.864**	**0.975**	0.500	**0.967**
Adaboost	0.950	0.894	**0.966**	0.938	**0.927**	0.938	0.839	0.958	0.948	**0.971**
FARCHD	0.905	0.841	0.984	0.945	0.873	0.935	0.864	**0.962**	0.957	0.929
NNCS	0.863	0.876	0.945	0.887	0.829	0.933	0.818	0.914	0.933	0.850

Table 7 Results of Friedman test for all the proposal configurations

	< 10	[10, 30]	> 30	A5-Traffic	Global
$AdaSSGACE_{KNN.40}$	2.05	1.8	2.25	3.75	2.46
$AdaSSGACE_{NN.40}$	2.8	3.8	3.35	2.15	3.02
$AdaSSGACE_{SVM.40}$	4.5	4.85	5.3	4.2	4.71
$AdaSSGACE_{KNN.45}$	4.35	1.8	2.45	4.1	3.17
$AdaSSGACE_{NN.45}$	3	4	3.15	2.9	3.26
$AdaSSGACE_{SVM.45}$	4.3	4.75	4.5	3.9	4.36

datasets are used, and each execution was repeated 10 times. The name of each column corresponds to the name of the dataset. For this part of the experimentation, the results obtained by all the techniques are similar. The techniques mentioned in the previous experiments maintain good performance, while others with poorer results, such as CPW or Adaboost, improve their performance on these datasets.

To assess whether the differences in performance observed in the previous tables are significant or not, it is necessary to perform statistical tests. For this reason, in this article we follow the guidelines proposed in [12], where non-parametric statistical testing is suggested in situations like the one faced in this study (several datasets, algorithms and configurations).

First, the Friedman test [12] has been used for multiple comparisons to check if significant differences exist among the set of algorithms. Besides this, the average rank return by this test allows sorting the algorithms in terms of performance. Each column of Tables 7 and 8 shows the mean ranking provided by this non-parametric test for each group of datasets (imbalanced with different IRs and A5-Traffic), and globally over all datasets, for all the configurations of the proposal, and between the comparative methods and the best proposed configuration obtained respectively. In case of the configurations, the best values are obtained by (K-NN, 40), followed by the (NN,40) configuration. As best configuration, $AdaSSGACE_{KNN,40}$ will be used in Table 8 as the reference of the proposal. Then, in the mentioned table, the best global rank is obtained by RUS on all the datasets, followed by UOBag. The proposed configuration obtains the 3rd best rank. Looking at each group of datasets, we can see that RUS gets the best average ranking in the imbalanced datasets with IR lower than 10, between 10 and 30, and greater than 30, while UOBag obtains the better rank in A5-Traffic. The best rank obtained by the proposal is in the datasets with IR between 10 and 30, where it gets the second best rank.

To assess if the performance of the best technique in the experimentation is significantly different from the other techniques from the state-of-the-art, we applied Holm's [26] and Finner's [18] post-hoc tests. Table 9 shows the results returned by the Holm's and Finner's post-hoc tests using RUS as the control method for all the datasets, except in A5-traffic datasets, where UOBAG is used as the control method due to the rank obtained in the previous Friedman tests. These tests were applied for each group of datasets, and globally over all datasets. The differences are considered significant when the p-value returned by the test is lower than 0.05. The values are rounded up to a maximum of three decimal places for the sake of the visualization.

These tests show that the results obtained by the proposed technique have no significant differences from those obtained by the state-of-the art techniques with KNN as base classifier and GA_{size} set to 40, according to both statistical tests, except for $IR > 30$ datasets and in a global way. Finally, in the case of state-of-the-art techniques, RUSBoost obtains significantly better results than all the techniques except UOBAG in all the datasets, and Adaboost in $IR \in [10, 30]$ datasets. For A5-Traffic datasets, where UOBAG is the reference method, it only obtains significantly better results against NNCS.

Table 8 Results of Friedman test between state-of-the-art algorithms and the proposal

	< 10	[10, 30]	> 30	A5-Traffic	Global
$AdaSSGACE_{KNN.40}$	3.4	2.95	4	4.45	3.7
RUS	2.35	2.8	1.85	2.9	2.475
UOBAG	2.55	3.45	3.15	2.65	2.95
CWP	5.25	4.7	4.35	4.65	4.73
Adaboost	4.65	4.35	4.8	3.45	4.31
FARCHD	4.75	4.2	4.05	3.95	4.23
NNCS	5.05	5.55	5.8	5.95	5.58

Table 9 Results of Holm and Finner tests for the experimental techniques

	<10		[10, 30]		>30		A5-Traffic		Global	
	Holm	Finner	Holm	Finner	Holm	Finner	Holm	Finner	Holm	Finner
$AdaSSGACE_{KNN.40}$	0.55	0.32	1.002	0.87	0.068	0.033	0.25	0.12	0.022	0.013
UOBAG	0.83	0.83	1.002	0.56	0.178	0.178	–	–	0.325	0.325
CPW	0.01	0.01	0.24	0.14	0.038	0.019	0.19	0.11	0	0
Adaboost	0.05	0.025	0.434	0.205	0.011	0.006	0.815	0.466	0	0
FARCHD	0.05	0.025	0.441	0.212	0.068	0.033	0.535	0.255	0	0
NNCS	0.025	0.01	0.026	0.026	0	0	0.003	0.003	0	0
RUS	–	–	–	–	–	–	0.815	0.795	–	–

5 Conclusions

In this paper, we have presented a new ensemble classification approach for imbalanced data based on feature space partitioning and hybrid metaheuristics, and concretely, on the Adaptive Splitting and Selection Strategy and the GACE metaheuristic, respectively. The main objective of this new method was to deal with imbalanced data without the use of data-level methods, something that usually entails an extra cost in terms of the time required for preprocessing the data.

The developed technique has been applied to a total of 40 datasets of different types: datasets with different imbalance ratios, and real imbalanced datasets with traffic information. Furthermore, the proposal has been compared with state-of-the-art classification techniques in the literature, such as RUSBoost, FARCHD, CPW, Adaboost, and NNCS. The performance obtained by the proposed method in most cases is similar to or better than the results obtained by the compared techniques, regardless of whether or not they used data-level methods. The best results so far have been obtained with configurations with KNN and NN as the algorithm to generate the base classifiers, with different sizes for the genetic populations. Statistical tests have been applied in order to corroborate the results obtained.

As future lines of research, it would be interesting to use a heterogeneous pool of classifiers instead of a homogeneous one. Also, another type of algorithms to be used as base classifiers could be explored. Configurations with various different sizes of the GA sub-population could be applied in order to show a more dedicated analysis to the optimization method in this theme. Besides, more techniques from the literature could be used for the comparison. In this paper, the experimentation has been focused on the performance of the presented proposal in binary imbalanced classification. In future work, OVO and OVA-based ensembles will be used with multi-class imbalanced datasets, and a study of the time consumed and the diversity of different population configurations will be made. Finally, in this paper, a hybrid method was formed by joining a GA and the CE technique. Other methods, such as PSO, ACO or BAT algorithm, could be used to replace either of the two components in order to compare the performance against the proposal in ensemble classification.

Acknowledgments This work has been supported by the research projects TEC2013-45585-C2-2-R and TIN2014-56042-JIN from the Spanish Ministry of Economy and Competitiveness, the TIMON project, which received funding from the European Union Horizon 2020 research and innovation programme under grant agreement No. 636220, and the LOGISTAR project, which received funding from European Union's Horizon 2020 research and innovation programme under grant agreement No. 769142.

References

1. Alcala-Fdez J, Alcala R, Herrera F (2011) A fuzzy association rule-based classification model for high-dimensional problems with genetic rule selection and lateral tuning. IEEE Trans Fuzzy Syst 19(5):857–872
2. 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 Multiple-Valued Logic Soft Comput 17(2–3):255–287
3. Amami R, Ben Ayed D, Ellouze N (2013) Adaboost with SVM using GMM supervector for imbalanced phoneme data. In: 2013 The 6th international conference on human system interaction (HSI), pp 328–333
4. Bäck T, Schwefel H (1993) An overview of evolutionary algorithms for parameter optimization. Evol Comput 1(1):1–23
5. Bi Y, Guan J, Bell D (2008) The combination of multiple classifiers using an evidential reasoning approach. Artif Intell 172(15):1731–1751
6. Bian J, Peng XG, Wang Y, Zhang H (2016) An efficient cost-sensitive feature selection using chaos genetic algorithm for class imbalance problem. Math Probl Eng, 2016
7. Burges C (1998) A tutorial on support vector machines for pattern recognition. Data Min Knowl Disc 2(2):121–167

8. Cervantes J, Huang DS, García-Lamont F, Chau A (2014) A hybrid algorithm to improve the accuracy of support vector machines on skewed data-sets. In: International conference on intelligent computing, pp 782–788
9. Chawla NV, Lazarevic A, Hall LO, Bowyer KW (2003) SMOTEBoost: improving prediction of the minority class in boosting. In: European conference on principles of data mining and knowledge discovery. Springer, pp 107–119
10. Cover T, Hart P (1967) Nearest neighbor pattern classification. IEEE Trans Inf Theory 13(1):21–27
11. Danesh A, Moshiri B, Fatemi O (2007) Improve text classification accuracy based on classifier fusion methods. In: 10th International conference on information fusion, pp 1–6
12. Derrac J, García S, Molina D, Herrera F (2011) A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evol Comput 1(1):3–18
13. Díez-Pastor JF, Rodríguez GOCJ, Kuncheva LIJ (2015) Random balance: ensembles of variable priors classifiers for imbalanced data. Knowl-Based Syst 85:96–111
14. Dorigo M, Gambardella LM (1997) Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Trans Evol Comput 1(1):53–66
15. Duin RP (2002) The combining classifier: to train or not to train? In: Proceedings 16th international conference patter recognition, vol 2. IEEE, pp 765–770
16. Eshelman LJ, Schaffer JD (1992) Real-coded genetic algorithms and interval-schemata. Found Gen Algor 2:187–202
17. Fattahi S, Othman Z, Othman Z (2015) New approach with ensemble method to address class imbalance problem. J Theor Appl Inf Technol 72:1
18. Finner H (1993) On a monotonicity problem in step-down multiple test procedures. J Am Stat Assoc 88(423):920–923
19. Giacinto G, Roli F (2001) Dynamic classifier selection based on multiple classifier behaviour. Pattern Recogn 34(9):1879–1881
20. Goldberg DE, Deb K (1991) A comparative analysis of selection schemes used in genetic algorithms. Found Gen Algor 1:69–93
21. Haixiang G, Xiuwu L, Kejun Z, Chang D, Yanhui G (2011) Optimizing reservoir features in oil exploration management based on fusion of soft computing. Appl Soft Comput 11(1):1144–1155
22. Hashem S (1997) Optimal linear combinations of neural networks. Neural Netw 10(4):599–614
23. Herrera F, Lozano M, Verdegay JL (1998) Tackling real-coded genetic algorithms: operators and tools for behavioural analysis. Artif Intell Rev 12(4):265–319
24. Ho D, Drake T, Bentley R, Valea F, Wax A (2015) Evaluation of hybrid algorithm for analysis of scattered light using ex vivo nuclear morphology measurements of cervical epithelium. Biom Opt Express 6(8):2755–2765
25. Holland JH (1992) Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. MIT Press
26. Holm S (1979) A simple sequentially rejective multiple test procedure. Scand J Stat 6:65–70
27. Hopfield J (1982) Neural networks and physical systems with emergent collective computational abilities. Proc Natl Acad Sci USA 79(8):2554–2558
28. Jackowski K, Wozniak M (2009) Algorithm of designing compound recognition system on the basis of combining classifiers with simultaneous splitting feature space into competence areas. Pattern Anal Applic 12(4):415–425
29. Jackowski K, Krawczyk B, Woźniak M (2014) Improved adaptive splitting and selection: the hybrid training method of a classifier based on a feature space partitioning. Int J Neural Syst 24(03):1430007
30. Jackowski K (2015) Adaptive splitting and selection algorithm for regression. N Gener Comput 33(4):425–448
31. del Jesus M, Hoffmann F, Junco L, Sánchez L (2004) Induction of fuzzy-rule-based classifiers with evolutionary boosting algorithms. IEEE Trans Fuzzy Syst 12(3):296–308
32. Jurek A, Bi Y, Wu S, Nugent C (2011) Classification by cluster analysis: a new meta-learning based approach. Multiple Classif Syst, 259–268
33. Jurek A, Bi Y, Wu S, Nugent C (2014) A survey of commonly used ensemble-based classification techniques. Knowl Eng Rev 29(5):551–581
34. Kennedy J (2011) Particle swarm optimization. Encyclopedia of machine learning. Springer, pp 760–766
35. Krawczyk B, Cyganek B (2017) Selecting locally specialised classifiers for one-class classification ensembles. Pattern Anal Appl 20(2):427–439
36. Krawczyk B, McInnes BT (2018) Local ensemble learning from imbalanced and noisy data for word sense disambiguation. Pattern Recogn 78:103–119
37. Kuncheva LI (2004) Combining pattern classifiers: methods and algorithms. Wiley
38. Kuncheva LI, Jain LC (2000) Designing classifier fusion systems by genetic algorithms. IEEE Trans Evol Comput 4(4):327–336
39. Kuncheva LI, Whitaker CJ, Shipp CA, Duin RP (2003) Limits on the majority vote accuracy in classifier fusion. Pattern Anal Appl 6(1):22–31
40. Lavanya S, Palaniswami S, Divyabharathi M (2015) Resampling ensemble algorithm for class imbalance problem using optimization algorithm. Int J Appl Eng Res 10(13):11520–11526
41. Liu X, Lin J, Deng K (2011) Scheduling optimization in re-entrant lines based on a GA and PSO hybrid algorithm. Tongji Daxue Xuebao/J Tongji Univ 39:726–729
42. Lopez-Garcia P, Onieva E, Osaba E, Masegosa A, Perallos A (2016) Gace: a meta-heuristic based in the hybridization of genetic algorithms and cross entropy methods for continuous optimization. Expert Syst Appl 55:508–519
43. Lopez-Garcia P, Onieva E, Osaba E, Masegosa AD, Perallos A (2016) A hybrid method for short-term traffic congestion forecasting using genetic algorithms and cross entropy. IEEE Trans Intell Transp Syst 17(2):557–569
44. Lopez-Garcia P, Woźniak M, Onieva E, Perallos A (2016c) Hybrid optimization method applied to adaptive splitting and selection algorithm. Lecture notes in computer science, vol 9648. Springer, pp 742–750
45. Mauša G, Galinac Grbac T (2017) Co-evolutionary multi-population genetic programming for classification in software defect prediction: an empirical case study. Appl Soft Comput J 55:331–351
46. Mokeddem D, Belbachir H (2009) A survey of distributed classification based ensemble data mining methods. J Appl Sci 9(20):3739–3745
47. Opitz DW, Maclin R (1999) Popular ensemble methods: an empirical study. J Artif Intell Res 11:169–198
48. Paredes R, Vidal E (2006) Learning weighted metrics to minimize nearest-neighbor classification error. IEEE Trans Pattern Anal Mach Intell 28(7):1100–1110
49. Qian Y, Liang Y, Li M, Feng G, Shi X (2014) A resampling ensemble algorithm for classification of imbalance problems. Neurocomputing 143:57–67
50. Rokach L (2010) Ensemble-based classifiers. Artif Intell Rev 33(1):1–39
51. Ruta D, Gabrys B (2005) Classifier selection for majority voting. Inform Fus 6(1):63–81

52. Seiffert C, Khoshgoftaar TM, Van Hulse J, Napolitano A (2010) RUSBoost: a hybrid approach to alleviating class imbalance. IEEE Trans Syst Man Cybern-Part A: Syst Humans 40(1):185–197
53. Sentinella M, Casalino L (2009) Cooperative evolutionary algorithm for space trajectory optimization. Celest Mech Dyn Astron 105(1-3):211
54. Stanciu S, Tranca D, Stanciu G, Hristu R, Bueno J (2016) Perspectives on combining nonlinear laser scanning microscopy and bag-of-features data classification strategies for automated disease diagnostics. Opt Quant Electron 48(6):320
55. Vorraboot P, Rasmequan S, Chinnasarn K, Lursinsap C (2015) Improving classification rate constrained to imbalanced data between overlapped and non-overlapped regions by hybrid algorithms. Neurocomputing 152:429–443
56. Wang S, Yao X (2009) Diversity analysis on imbalanced data sets by using ensemble models. In: Proceedings of IEEE symposium in computational intelligence and data mining, 2009, CIDM'09, pp 324–331
57. Wang S, Yao X (2012) Multiclass imbalance problems: analysis and potential solutions. IEEE Trans Syst Man Cybern Part B (Cybern) 42(4):1119–1130
58. Wang S, Minku L, Yao X (2015) Resampling-based ensemble methods for online class imbalance learning. IEEE Trans Knowl Data Eng 27(5):1356–1368
59. Xu L, Krzyzak A, Suen CY (1992) Methods of combining multiple classifiers and their applications to handwriting recognition. IEEE Trans Syst Man Cybern 22(3):418–435
60. Yang J, Ji Z, Xie W, Zhu Z (2016) Model selection based on particle swarm optimization for omics data classification. Shenzhen Daxue Xuebao (Ligong Ban)/J Shenzhen Univ Sci Eng 33(3):264–271
61. Yang P, Xu L, Zhou B, Zhang Z, Zomaya A (2009) A particle swarm based hybrid system for imbalanced medical data sampling. BMC Genomics 10:Suppl. 3. https://doi.org/10.1186/1471-2164-10-S3-S34
62. Yang XS (2010) A new metaheuristic bat-inspired algorithm. Stud Comput Intell 284:65–74
63. Yu H, Ni J, Zhao J (2013) ACOSampling: an ant colony optimization-based undersampling method for classifying imbalanced DNA microarray data. Neurocomputing 101:309–318
64. Zhou ZH, Liu XY (2006) Training cost-sensitive neural networks with methods addressing the class imbalance problem. IEEE Trans Knowl Data Eng 18(1):63–77
65. Krawczyk B (2016) Learning from imbalanced data: open challenges and future directions. Progress Artif Intell 5(4):221–232
66. Cano A, Zafra A, Ventura S (2013) Weighted data gravitation classification for standard and imbalanced data. IEEE Trans Cybern 43(6):1672–1687
67. Mahdizadehaghdam S, Dai L, Krim H, Skau E, Wang H (2017) Image classification: a hierarchical dictionary learning approach. In: IEEE International conference in acoustics, speech and signal processing (ICASSP), 2017, pp 2597–2601
68. Khari M, Kumar P, Burgos D, Crespo RG (2017) Optimized test suites for automated testing using different optimization techniques. Soft Comput, 1–12
69. Fernández A, García S, Herrera F (2011) Addressing the classification with imbalanced data: open problems and new challenges on class distribution. Hybrid Artif Intell Syst, 1–10
70. Sun Y, Wong AK, Kamel MS (2009) Classification of imbalanced data: a review. Int J Pattern Recognit Artif Intell 23(04):687–719
71. Krawczyk B, Cano A, Woźniak M (2018) Selecting local ensembles for multi-class imbalanced data classification, In: 2018 International joint conference on neural networks (IJCNN) 1–8
72. Fernandez A, Garcia S, Galar M, Prati RC, Krawczyk B, Herrera F (2018) Learning from imbalanced data sets. Springer

Dr. Pedro Lopez-Garcia obtained his Technical Engineer in Computer Systems from University of Almeria in 2012. He received the title of Master Degree in Soft Computing and Intelligent Systems in 2013 from University of Granada. In 2014, He has obtained a Master Degree in Advanced Artificial Intelligent from National University of Distance Education while he started working in Mobility Unit at DeustoTech research center. In 2016, He obtained his PhD in Engineering from the University of Deusto with Cum Laude qualification and International Mention. He authored more than 15 scientific articles, and has participated in national and international projects. His main research interests are focused on Intelligent Systems, Imbalanced Classification, Hybrid Metaheuristics, Optimization, Fuzzy Systems, Ensembles Classification, and Deep Learning.

Antonio D. Masegosa took his University degree in Computer Engineering in 2005 and his PhD in Computer Sciences in 2010, both from the University of Granada, Spain. From June 2010 to November 2014 he was a post-doc researcher at the Research Center for ICT of the University of Granada. In 2014 he received an IKERBASQUE Research Fellowship to work in the Mobility Unit of the Deusto Institute of Technology, in Bilbao, Spain. He has published four books, seventeen JCR papers and more than 20 papers in both international and national conferences. His main research interests are Artificial Intelligence, Intelligent Systems, Soft Computing, Hybrid Metaheuristics, Machine Learning, Deep Learning, Intelligent Transportation Systems, Logistic Networks, Travel Demand Estimation and Traffic Forecasting.

Eneko Osaba works at TECNALIA as researcher in the ICT/OPTIMA area. He obtained his Ph.D. degree on Artificial Intelligence (Cum Laude) in 2015 in the University of Deusto. He has participated in the proposal, development and justification of more than 15 research projects. He has contributed in the development of more than 85 papers, including more than 15 Q1. He has performed several stays in universities of United Kingdom, Italy and Malta. He served as a member of the program and/or organizing committee in more than 25 international conferences. At HAIS 2015 and IDC 2018, Eneko was also member of the organizing committee, and he organized several special sessions in conferences such CEC 2017, IDC 2018, IDEAL 2018, DCAI 2018, PAAMS 2019, GECCO 2019. Besides this, he is member of the editorial board of International Journal of Artificial Intelligence and Journal of Advanced Transportation. Furthermore, he has acted as guess editor in journals such as Journal of Computational Science, Neurocomputing, Logic Journal of IGPL and IEEE ITS Magazine.

Enrique Onieva is professor in the area of artificial intelligence, machine learning and Big Data at the University of Deusto, as well as Researcher of Intelligent Transportation Systems applications related with data processing, mobility and Smart city solutions in the DeustoTech-Mobility research Unit. Director of the Big Data Executive Program and the PhD program in Engineering for the Information Society and Sustainable Development of the Faculty of Engineering.

He has participated in more than 25 research projects. Among them, CYBERCARS-2 (FP6), ICSI (FP7) and PostLowCit (CEF-Transport). Research responsible for the Artificial Intelligence Work Package of the Project TIMON (H2020) and Project Coordinator of the LOGISTAR Project (H2020). He has authored more than 100 scientific articles. From them, more than 40 are published in journals of the highest level. His research has been recognized and awarded several times in international conferences, and his publications get an H-index of 17 (scopus.com).

Currently, he is one of the most prolific researchers in his area. His research interest is based on the application of Artificial Intelligence to Intelligent Transportation Systems, including fuzzy-logic based decision, evolutionary optimization and machine learning.

Dr. Asier Perallos PhD, MSc and BSc in Computer Engineering from the University of Deusto. More than 15 years of experience as a lecturer in the Faculty of Engineering at the University of Deusto. His teaching focuses on software design and distributed systems. Associate Professor at the University of Deusto, where he currently serves as Dean of the Faculty of Engineering. Principal Researcher of "Deusto Smart Mobility", a Basque Government's official research group that promotes the application of ICT to address smarter transport and mobility. In particular, his research background is focused on telematic systems, vehicular communication middleware and intelligent transportation systems. Over a decade of experience in R&D management, with tens of projects and technology transfer actions led.