Neural Comput & Applic (2015) 26:525–539
DOI 10.1007/s00521-014-1656-3

ADVANCES IN INTELLIGENT DATA PROCESSING AND ANALYSIS

Invasive weed classification

Roozbeh Razavi-Far · Vasile Palade · Enrico Zio

Received: 17 December 2013 / Accepted: 28 June 2014 / Published online: 30 July 2014

Abstract Invasive weed optimization (IWO) is a recently published heuristic optimization technique that resembles other evolutionary optimization methods. This paper proposes a new classification technique based on the IWO algorithm, called the invasive weed classification (IWC), to face the problem of pattern classification for multi-class datasets. The aim of the IWC is to find the set of the positions of the class centers that minimize the multi-objective function, i.e., the optimal positions of the class centers. The classification performance is computed as the percentage of misclassified patterns in the testing dataset achieved by the best plants in terms of fitness performance. The performance of the IWC algorithm, both in terms of classification accuracy and training time, is compared with other commonly used classification algorithms.

Keywords Pattern recognition · Invasive weed classification · Optimization

R. Razavi-Far · E. Zio
Department of Energy, Politecnico di Milano, Via Ponzio 34/3,
20133 Milan, Italy
e-mail: roozbeh.razavi@gmail.com; roozbeh@uwindsor.ca

R. Razavi-Far
Department of Electrical and Computer Engineering, University
of Windsor, Windsor, Canada

V. Palade (✉)
Faculty of Engineering and Computing, Coventry University,
Priory Street, Coventry, UK
e-mail: vasile.palade@coventry.ac.uk

E. Zio
Ecole Centrale Paris, Supelec, Paris,
92295 Chatenay-Malabry Cedex, France
e-mail: enrico.zio@ecp.fr; enrico.zio@supelec.fr;
enrico.zio@polimi.it

1 Introduction

In machine learning, pattern recognition is the task of assigning a label to each given input pattern. There exist two fundamental approaches for pattern recognition: unsupervised learning and supervised learning.

The first approach, usually called clustering, is to partition a given set of unlabelled patterns into groups containing similar patterns. In the latter, commonly called classification, a set of patterns (the so-called training set) with pre-assigned labels are used for learning. Therefore, classification maps a pattern from the feature space to a pre-defined class in the decision space. This mapping can then be used for the classification of upcoming new patterns.

Classification techniques vary in the learning mechanism and in the representation of the learned model. Early work on pattern classification was focused on constructing a set of discriminant functions [23], based on a set of training patterns, in order to classify among the pre-defined groups by the assigned classes.

Contemporary statistical approaches for pattern classification are mainly based on distances in the feature space, as is the case, for example, in the k-nearest neighbors method [18], or in the Bayesian classification, where one estimates the joint distribution of the features within each class [24], or in the classification tree-based analysis [12], where one tests one or more predictor variables.

Other popular pattern classification methods are rule-based learning methods [3] (including here fuzzy rule-based classification systems such as [41]), neural networks [11], support vector machines [33], neuro-fuzzy networks [46], evolutionary computing-based classifiers [6], instance-based learning systems [42], rough set-based systems [28], ensemble of classifiers [31, 35, 55, 56, 59, 60, 63].

Conversely, the problem of pattern classification can be simply considered as a multivariable optimization problem. Indeed, each class prototype corresponds to a centroid in a multi-dimensional feature space. Thus, the problem of pattern classification can be reformulated such that to find the optimal positions of all the class centroids, or more explicitly to determine optimal coordinates of each centroid.

There exist in the literature a few papers in which optimization techniques are used as a classification tool [6, 7, 13, 15, 21, 22, 48–53, 66–68].

The above-mentioned references indicate that optimization techniques are proper and effective techniques for implementing classification tasks and that they can be effectively applied to many problem domains. Yet there are no papers using invasive weed optimization as a classification technique.

Much work has been devoted to the invasive weed optimization (IWO) since the proposal of the algorithm in [44], but this mainly concerned function optimization [26, 27, 47, 61] and not pattern classification. More recently, a newer attempt that seeks to apply IWO to more diversified areas has also appeared [16].

This paper presents a novel method for pattern classification in multi-class datasets based on the analogy with the weed colonization process in nature.

The structure of the paper is as follows. Section 2 describes the IWO algorithm background and basic scheme, while Sect. 3 illustrates the proposed IWC version applied to a classification problem. The basic concepts of seeding, growth and competition that are analogous to what happens in a weed colony are utilized in this algorithm. Section 4 reports a performance study of the proposed classification approach, conducted on different challenging test datasets, and it discusses the results achieved and the comparison against well-known classification techniques. Finally, Sect. 5 presents some conclusions and ideas for future work.

2 Invasive weed optimization (IWO)

In this work, the IWO algorithm is used to find the optimal positions or coordinates of the class centroids with respect to a pre-defined multi-objective function. A comparative evaluation of the proposed IWC algorithm and other well-known classification algorithms is carried out in order to validate the appropriateness of the IWC as a pattern classifier.

2.1 Background and applications

The Invasive Weed Optimization (IWO) algorithm is a bio-inspired numerical optimization algorithm that simulates the behavior of weeds in nature when colonizing and finding a suitable place for growth and reproduction [44].

Since the earliest development for the optimization and tuning of a robust controller, IWO has been extensively used in a variety of practical applications, such as plant-wide performance optimization [26, 27, 61], development of a recommender system [54], optimal positioning of piezoelectric actuators [45], optimal tuning of thresholds for new class fault detection [62], cooperative identification and adaptive control of a surge tank [30], electromagnetics and antenna configuration [34], cooperative multiple task assignment of the Unmanned Aerial Vehicles [25], clustering [16], analysis of electricity markets dynamics [29, 47].

2.2 Invasive weed optimization algorithm

Before presenting the IWO algorithm in detail, we first define the key terms and concepts used in this algorithm [34, 44].

Seed each individual in the colony that includes a value for each variable in the optimization problem prior to fitness evaluation.

Fitness a value that represents the merit of the solution for each seed.

Weed/Plant each evaluated seed grows to a flowering plant or weed in the colony. Therefore, growing a seed to a plant corresponds to evaluating an individual's fitness.

Colony the set of all seeds.

Population size the number of plants in the colony.

Maximum weed population a pre-defined parameter that represents the maximum allowed number of weeds in the colony posterior to fitness evaluation.

The following steps are considered to imitate the colonizing behavior of weeds [34, 44]:

1. *Search space definition* Initially, the number of parameters that need to be optimized has to be defined, hereafter denoted by D. Next, for each parameter in the D-dimensional search space, a minimum and maximum values are assigned.
2. *Population initialization* A limited number P_0 of initial seeds, $W = \{w_1, w_2, \ldots, w_{P_0}\}^T$ are being randomly spread through the defined search space. Consequently, each seed catches a random position in the D-dimensional search space [44].
3. *Fitness estimation* A value is assigned to each initial seed by the fitness function J, which represents the goodness of the solution [34]. Here, initial seeds grow up to flowering plants.
4. *Ranking and reproduction* The flowering plants are firstly ranked based on their assigned fitness values relative to others. Subsequently, the flowering plants in the colony will produce a number of new seeds depending on their rank; i.e., the number of seeds

produced by each plant increases linearly from the minimum possible seeds production, $S_{\min}$, to its maximum, $S_{\max}$, based on its own fitness value, as well as the lowest and the highest fitness values in the colony.

Hence, the plants with higher fitness value, which are more adapted to the environment, can produce more seeds that will solve the problem better [34]. More precisely, the number of seeds to be created by each plant is computed as follows:

$$S_i = \left\lfloor \frac{F_i - F_{\min}}{F_{\max} - F_{\min}} (S_{\max} - S_{\min}) + S_{\min} \right\rfloor \tag{1}$$

where F_i is the fitness of the i-th plant. $F_{\min}$ and $F_{\max}$ stand for the lowest and highest fitness values in the weed population. Therefore, this step guarantees cooperation of every weed in the reproduction procedure. Note that $S_{\max}$ and $S_{\min}$ are pre-defined parameters of the algorithm and adjusted according to the structure of the problem. Further details about this strategy can be found in [44].

5. *Spatial dispersal* This part of the algorithm introduces some randomness in the search process and provides adaptation [44]. Here, the seeds are being randomly scattered through the search space by using normally distributed numbers with zero mean and adaptive standard deviations [44] as follows:

$$w_s[\kappa] = w[\kappa] + \mathcal{N}\left(0, \sigma_{\text{iter}}^2\right) \tag{2}$$

where, $w[\kappa]$ indicates the κ-th variable of a solution vector in the current iteration and $w_s[\kappa]$ shows the κ-th variable of its s-th seeds. The standard deviation σ_{iter} at the present time step can be computed adaptively according to the following equation [44]:

$$\sigma_{\text{iter}} = \frac{(\text{iter}_{\max} - \text{iter})^n}{(\text{iter}_{\max})^n} (\sigma_{\text{initial}} - \sigma_{\text{final}}) + \sigma_{\text{final}} \tag{3}$$

where σ_{initial} and σ_{final} denote the pre-defined initial and final standard deviations, respectively. $\text{iter}_{\max}$ indicates the maximum allowed number of iteration cycles and n is the nonlinear modulation index [44] assigned by the user. The σ_{iter} can be reduced from the σ_{initial} to the σ_{final} with different velocities in accordance with the chosen nonlinear modulation index, n.

Initially, the whole search space can be explored by the algorithm due to the high value of initial standard deviation σ_{initial}. Then, the standard deviation σ_{iter} is gradually reduced by increasing the number of iterations, to focus the search around the local minima or maxima to find the global optimum. This gradual reduction guarantees to collect only fitter plants and to discard plants with lower fitness. The produced seeds, along with their parents, are considered as the potential solutions for the next population.

6. *Competitive exclusion* After passing a number of iterations, the population size reaches its pre-defined maximum $(P_{\max})$ by fast reproduction and, consequently, a mechanism for discarding the plant with low fitness will be activated. To this end, the seeds and their parents are ranked together and those with higher fitness survive and subsequently reproduce new seeds in the next iteration.
7. *Termination condition* The plants that survived in the next generation will continue to produce new seeds depending on their fitness rank in the colony. The procedure is repeated at step 3 until either the maximum allowed number of iterations has been reached or the fitness criterion has been met [34].

A pseudocode version of the IWO algorithm is given in Fig. 1 [45].

3 Invasive weed classification

Let us assume a multi-class dataset with C classes, where N features are available for the classification. The classification problem is the task of finding the optimal positions of C centroids in an N-dimensional feature space, i.e., finding out the N coordinates of each centroid. Invasive weed optimization offers a potentially valid framework to handle such classification task.

The IWO algorithm (see Fig. 2) is the cornerstone of the IWC algorithm. Thus, to develop the IWC algorithm, the key parameters of the IWO algorithm are firstly set (see Table 2). The initial population is adopted and initialized according to the available data (see Sect. 3.1), and multiple objective functions are defined (see Sect. 3.2) to meet the goal of the classification task.

To perform the classification task, a multi-objective function has been defined, as described in this subsection. For each upcoming dataset, we use cross-validation, and thus, in each iteration of cross-validation, the original dataset is divided into train, validation and test sets:

$$\begin{aligned} D_{\text{train}} &= \left\{(x_1, y_1), \ldots, \left(x_p, y_p\right), \ldots, \left(x_{m_{\text{train}}}, y_{m_{\text{train}}}\right)\right\} \\ D_{\text{valid}} &= \left\{(x_1, y_1), \ldots, \left(x_{p'}, y_{p'}\right), \ldots, \left(x_{m_{\text{valid}}}, y_{m_{\text{valid}}}\right)\right\} \\ D_{\text{test}} &= \left\{(x_1, y_1), \ldots, \left(x_q, y_q\right), \ldots, \left(x_{m_{\text{test}}}, y_{m_{\text{test}}}\right)\right\} \end{aligned} \tag{4}$$

where $\left(x_p, x_{p'}, x_q \in X\right)$ are patterns along with their correct labels $\left(y_p, y_{p'}, y_q \in Y\right)$, $Y \subset \{\omega_1, \ldots, \omega_c, \ldots, \omega_C\}$, ω_c are classes, $p = 1, \ldots, m_{\text{train}}$, $p' = 1, \ldots, m_{\text{valid}}$, and $q = 1, \ldots, m_{\text{test}}$.

```
INPUTS:
    $P_0 \leftarrow 5$; is a limited number of initial seeds
    $S_{min} \leftarrow 0$ and $S_{max} \leftarrow 5$; are the minimum and maximum possible seeds production
    $iter_{max} \leftarrow 200$; indicates the maximum allowed number of iteration cycles
    $\sigma_{initial} \leftarrow 1$ and $\sigma_{final} \leftarrow 0.05$; denote the pre-defined initial and final standard deviations
    $n \leftarrow 3$; is the nonlinear modulation index
    $P_{max} \leftarrow 30$; is the maximum population size
DEFINITIONS:
    $F_i$ is the fitness of the $i-th$ plant
    $F_{min}$ and $F_{max}$ stand for the lowest and highest fitness values in the weed population
GENERATE a random population of $P_0$ individuals from a set of feasible solutions $W = \left\{w_1, w_2, \ldots, w_{P_0}\right\}^T$
for $iter = 1$ to $iter_{max}$ do
    EVALUATE the fitness function for each individual in $W$
    COMPUTE the maximum and minimum fitness in the colony $F_{max}$ and $F_{min}$
    for each individual $w_i$ do
        COMPUTE the number of seeds for $w_i$, $S_i = \lfloor (F_i - F_{min})(S_{max} - S_{min}) / (F_{max} - F_{min}) + S_{min} \rfloor$
        RANDOMLY distribute seeds over the search space with normal distribution $\mathcal{N}\left(0, \sigma_{iter}^2\right)$ around the parent plant $w$, with zero
        mean and an adaptive standard deviation:

            $\sigma_{iter} = \sigma_{final} + \left(\sigma_{initial} - \sigma_{final}\right)(iter_{max} - iter)^n / (iter_{max})^n$

        ADD the generated seeds to the solution set, $W$
    end
    if $(|W| = P) > P_{max}$ then
        SORT the population $W$ in descending order of their fitness
        TRUNCATE population of weeds with smaller fitness, so-called competitive exclusion, until $P = P_{max}$
    end
end
BEST solution is the plant $w_{best}$ with minimum fitness in the last population
```

Fig. 1 The pseudo-code for the IWO algorithm [45]

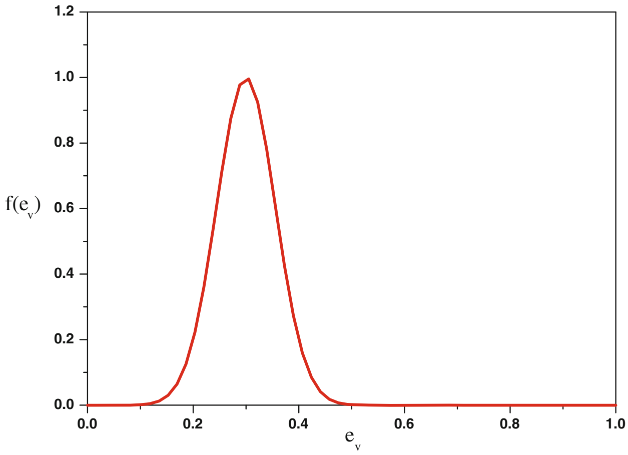

Fig. 2 The multi-criteria function $f(e_v)$

3.1 Initialization

At the beginning, the cluster centroids are some initial seeds, which are randomly spread within the N-dimensional feature space. The IWC evaluates and manipulates iteratively the seeds (the initial random cluster centroids) based on their fitness and finally converges to the optimal position of the cluster centroids. Thus, the optimal positions of the cluster centroids are those of the most fitted seeds, which minimize the multi-objective performance functions.

With these assumptions, a limited number P_0 of initial seeds $W = \{w_1, w_2, \ldots, w_{P_0}\}^T$ are being randomly spread across the defined N-dimensional feature space. Each individual seed w_i is encoded by C centroids as follow:

$$w_i = \left\{w_i^1, \ldots, w_i^C\right\} \tag{5}$$

Consequently, each seed catches a random position in the N-dimensional feature space, i.e., w_i^c for the c_{th} centroid comprises N real values. Each value corresponds to one coordinate in the N-dimensional feature space:

$$w_i^c = \left\{w_{i,1}^c, \ldots, w_{i,N}^c\right\} \tag{6}$$

Thus, each seed in the colony comprises $C \times N$ real values.

3.2 Multi-objective optimization

The task of multi-objective optimization is to optimize multiple conflicting objective functions concurrently, subject to a set of pre-assigned constraints. The problem can be reformulated as one where the goal is to optimize a given set J of θ objective functions $J_1, J_2, \ldots, J_\theta$ with respect to a set of μ variables $\alpha = [\alpha_1, \alpha_2, \ldots, \alpha_\mu]$:

$$\begin{aligned} &\underset{\alpha}{\text{minimize}} \quad (J_1(\alpha), J_2(\alpha), J_3(\alpha), \ldots, J_\theta(\alpha)) \\ &\text{subject to}: \; g_\xi(\alpha) \le 0 \\ &\qquad\qquad h_\zeta(\alpha) = 0 \\ &\qquad\qquad \alpha_{\text{lower}} \le \alpha \le \alpha_{\text{upper}} \end{aligned} \tag{7}$$

where $g_\xi(\alpha)$ and $h_\zeta(\alpha)$ stand for the ξ-th and ζ-th inequality and equality constraints, respectively. Besides each parameter can be constrained between a lower and an upper bound, α_{lower} and α_{upper}, respectively.

A set of possible solutions in the objective space is called Pareto optimal, if there does not exist another solution that dominates it. The vector α is a non-dominated solution if for any other possible objective vector, α^*:

$$J_\xi(\alpha) \le J_\xi(\alpha^*); \quad \forall \xi = 1, 2, 3, \ldots, \theta \tag{8}$$

and at least there is an objective such that:

$$J_\zeta(\alpha) \prec J_\zeta(\alpha^*); \quad \zeta \in 1, 2, 3, \ldots, \theta \tag{9}$$

Dominance can be indicated as $J(\alpha) \prec J(\alpha^*)$. The set of Pareto optimal solutions is often called the Pareto front.

3.2.1 *Class-based objective function*

To define the multi-objective function in our case, different optimization goals have been considered. The first objective function, J_c, is a class-oriented function. It is calculated based on the fitness of the individual seeds in the population, which are randomly spread through the feature space and each one corresponds to a set of positions of the C centroids, $c = 1, \ldots, C$. Any pattern in the training set is assigned to the class with the nearest possible centroid in the N-dimensional feature space.

A misclassified pattern $\vec{x}_p$ is simply one pattern assigned to a class $\omega_{\text{assigned}}(\vec{x}_p)$, different from the true class, $\omega_{\text{true}}(\vec{x}_p)$, stored in the training set. Then, the fitness of the i_{th} seed in the colony is calculated as:

$$J_c(i) = \frac{1}{m_{\text{train}}} \sum_{p=1}^{m_{\text{train}}} \lambda(\vec{x}_p) \tag{10}$$

where m_{train} in the above equation stands for the number of patterns in the training set and $\lambda(\vec{x}_p)$ is defined as follows:

$$\lambda(\vec{x}_p) = \begin{cases} 1 & \text{if} \quad \omega_{\text{assigned}}(\vec{x}_p) \ne \omega_{\text{true}}(\vec{x}_p) \\ 0 & \qquad \text{otherwise} \end{cases} \tag{11}$$

Note that J_c varies in the interval $[0, 1]$.

3.2.2 *Distance-based objective function*

Another objective function can be defined as a distance-based function, J_d. It is calculated as an average of the Euclidean distances:

$$J_d(i) = \frac{1}{m_{\text{train}}} \sum_{p=1}^{m_{\text{train}}} d\left(\vec{x}_p, w_i^{\omega_{\text{true}}(\vec{x}_p)}\right) \tag{12}$$

In the above equation, $\vec{x}_p$ stands for the p-th generic pattern, in the training set of size m_{train} that belongs to the class $\omega_{\text{true}}(\vec{x}_p)$ and $w_i^{\omega_{\text{true}}(\vec{x}_p)}$ is the current position of the related centroid of class $\omega_{\text{true}}(\vec{x}_p)$ in the training set that is taken by the i-th seed in the N-dimensional feature space. Both $\vec{x}_p$ and $w_i^{\omega_{\text{true}}(\vec{x}_p)}$ are N-dimensional vectors.

The Euclidean distance d in J_d is calculated in such a way that the components of $\vec{x}_p$ and $w_i^{\omega_{\text{true}}(\vec{x}_p)}$ are first normalized with respect to their maximum range and consequently the sum is divided by N.

The normalization helps to preserve the range of variation in the distance within the interval $[0, 1]$.

The class-based objective function J_c is defined so that it can vary in discrete increments of $1/m_{\text{train}}$ (generally as we move further from the cluster center), while the distance-based objective function J_d is more continuous. This implies that the J_c is not very sensitive to centroid positions, and small variations in the centroid positions might not lead to a change in the class membership. Consequently, the percentage of misclassified patterns remains constant. Conversely, the J_d is very sensitive to the centroid positions and can vary with respect to small variations in the centroid positions.

3.2.3 *The multi-objective function*

The two pre-described class and distance-based objective functions, J_c and J_d, are separately computed and linearly combined to form the multi-objective function as follows:

$$J(i) = \frac{1}{\theta} \sum_{\chi=1}^{\theta} J_\chi(i) = \frac{1}{2}[J_c(i) + J_d(i)] \tag{13}$$

where the two terms J_c and J_d are given equal weights, and J also varies in the range [0, 1].

3.2.4 *The overtraining problem*

The IWC has the potential to overfit the training set because its objective is to minimize the error on the training set. There is an increasing potential to overtrain the data containing higher rate of noise. This section focuses on alleviating the overfitting and on improving the performance by means of validation sets.

The basic idea to overcome the overfitting problem is to use a subset of data different from the training set, called validation set, and calculate the classification performance over the validation set.

The training focuses on learning a small hypothesis feature space, and the validation set includes a hypothesis from that feature space. Thus, in a large feature space, merely minimizing the training error leads to overtraining.

In particular, the so-called 'k-fold' cross-validation [19, 20, 36] is used to split the data into train, validation and test subsets as shown in Sect. 3. Here, the stratified k-fold cross-validation algorithm is used to prevent creating imbalanced data-subsets that can degrade the performance of the classifiers. The detailed explanation can be found in [65].

The IWC performance does not depend significantly on the selection of the parameter k: experiments not reported in the paper show that one gets slightly different results for different k values. In general, the choice of k, for instance $k = 10$ as used in this work, is somewhat arbitrary. k can be a divisor of the sample size or the size of the samples of the same groups that should be stratified. First of all, in order to lower the variance of the cross-validation result, one can repeat/iterate the cross-validation with new random splits. Ideally one can choose a very large k, but there is also a risk to be biased. Moreover, a very large k needs more computation time. The trade-off between the bias and the variance in cross-validation is studied in [64].

In this work, the k is set to 10 except for the very small size datasets (e.g., small size Gaussian dataset), which have less than 10 samples per class. In this case, a divisor of the sample size of the smallest group is used for the stratified cross-validation.

The original dataset is partitioned into k blocks of equal size. Cross-validation is performed by means of stratified sampling, where the class distribution in each fold is approximately the same as in the initial dataset [65]. Two of these blocks are used as validation and test subsets, and the remaining $k - 2$ blocks are combined together to form the training data subset. The cross-validation iteration is, then, repeated k times (the k-folds) using a different block as test set each time and consequently the IWC algorithm is repeated k times, each time with a different subsets.

A validation test set is devised to find out whether an optimal set of solution 'seeds' may cause overfitting. Each classifier trained by an individual set of optimal solutions 'seed' is then evaluated by the validation set. The performance evaluation on the validation set has been used as a factor to control the overtraining problem. Thus, the proposed term for the modification of the multi-objective function is an error-based objective function, J_e.

Here, we try to control the overtraining problem with respect to the validation error e_v such that the individual seed in the feature space receives the maximum penalty equal to 1, if the validation error takes a value outside of a closed interval of $[0.1, 0.5]$. For the centroids with the $e_v \in [0.1, 0.5]$, the $f(e_v)$ follows a Gaussian distribution defined as:

$$f(e_v) = 1 \times e^{\frac{-(e_v-0.3)^2}{2\times 0.4^2}} = e^{\frac{-(e_v-0.3)^2}{0.32}} \tag{14}$$

where $f(e_v)$ with $e_v \in [0.1, 0.5]$ follows a Gaussian 'bell curve' function that quickly falls toward zero. The height of the curve peak is 1, the center of the peak is located at 0.3, and the standard deviation that controls the width of the bell to be equal to 0.4 corresponds to $[0.1, 0.5]$. The domain of the $f(e_v)$ is between 0 and 1 since this is an error-based function, and the range of the function for the values e_v in the domain and outside of the closed interval $[0.1, 0.5]$ is equal to zero. The multi-criteria function $f(e_v)$ is defined as follows:

$$f(e_v) = \begin{cases} 0 & \text{if} \quad 0 \prec e_v \prec 0.1 \\ e^{-(e_v-0.3)^2/0.32} & \text{if} \quad 0.1 \leq e_v \leq 0.5 \\ 0 & \text{if} \quad 0.5 \prec e_v \prec 1 \end{cases} \tag{15}$$

The $f(e_v)$ is shown in Fig. 2.

Each pattern in the validation set D_{valid} is assigned to a class, that is the class with the nearest centroid in the N-dimensional feature space. Then, the validation error is calculated as a percentage of misclassified patterns with respect to the real label stored in D_{valid}, which is described as follows:

$$e_v(i) = \frac{1}{m_{\text{valid}}} \sum_{p'=1}^{m_{\text{valid}}} \lambda(\vec{x}_{p'}) \tag{16}$$

where m_{valid} in the above equation stands for the number of patterns in the validation set and $\lambda(\vec{x}_{p'})$ is a function of two criteria defined as follows:

$$\lambda(\vec{x}_{p'}) = \begin{cases} 1 & \text{if} \quad \omega_{\text{assigned}}(\vec{x}_{p'}) \neq \omega_{\text{true}}(\vec{x}_{p'}) \\ 0 & \text{otherwise} \end{cases} \tag{17}$$

Having $e_v(i)$, the function $f(e_v)$ is calculated based on the equation (15). Consequently, J_e is computed as follow:

$$J_e(i) = 1 - f(e_v) \tag{18}$$

To control the overtraining problem, a proper range of variation in the validation error $e_v \in [0.1, 0.5]$ has been devised to indicate whether a classifier is overtrained or weak-learned, and then, we control the fitness of the corresponding w_i representing C centroids in the population.

The parameters in equation 15 are selected properly to form the multi-criteria function, as displayed in Fig. 2. Having the defined multi-criteria function, the fitness $J_e(i)$ of the seeds (i.e., cluster centroids) that leads to an error e_v higher than 0.5 or less than 0.1 is increased. On the contrary, the fitness $J_e(i)$ of the seeds with $0.1 < e_v < 0.5$ (i.e., those that are not weak or not overtrained) is decreased exponentially, which minimizes the performance function. Indeed, other types/shapes of multi-criteria functions $f_{(e_v)}$ can be defined, such as triangular, etc.

The error-based objective function J_e increases the fitness of an overtrained or very weak-learned classifier; on the other hand, it decreases the fitness of a classifier that performs particularly well on the validation set and its validation error falls into the pre-devised range of $[0.1, 0.5]$. The modification term J_e is added to the multi-objective function, forming a modified function, called J' as follows:

$$J'(i) = \frac{1}{\theta}\sum_{\chi=1}^{\theta} J_{\chi}(i) = \frac{1}{3}[J_c(i) + J_d(i) + J_e(i)] \tag{19}$$

The constructing terms of the equations 13 and 19 (i.e., J_c, J_d, J_e) can vary in the interval [0,1]. All these terms are given equal weights in J and J'. In order to keep the same range of variation (i.e., [0,1]), the equal weights of $\frac{1}{2}$ and $\frac{1}{2}$ ($\frac{1}{3}$, $\frac{1}{3}$, $\frac{1}{3}$) are given to the J_c and J_d (J_c, J_d, J_e) in J (J'), respectively. However, one can increase or decrease the weights to give higher priority to the term of choice. The effect of unequal weights will be studied in future research.

As a result of the choice of these two different multi-objective functions J and J', two different invasive weed classifiers have been designed IWC$-J$ and IWC$-J'$, respectively. In the former, the focus is to optimize the position of the centroids in the N-dimensional feature space based on the training set D_t; in the latter, a mechanism has been also devised to control the overtraining.

The test performance either in IWC $-$ J and IWC $-$ J' is, then, computed as the percentage of misclassified patterns in the test set D_{test} by the best individual seed (i.e., the fittest seed). Resorting to cross-validation and test performance, the IWC $-$ J and IWC $-$ J' outcomes can be compared each other and against other well-known classification algorithms.

With the choice of the proposed multi-objective functions J and J', the classification task turns out to be a classical optimization problem.

4 Experimental results

In this section, different versions of invasive weed classifiers are firstly compared to each other and, then, compared to well-known classification techniques with respect to the classification of artificial datasets of different size and complexity, including the challenging artificial 3-spirals and other real-world datasets. A summary of the datasets characteristics (i.e., number of classes C, number of features N, number of patterns $m = m_{\text{train}} + m_{\text{valid}} + m_{\text{test}}$ and number of patterns per class m_c) is reported in Table 1.

The datasets have no missing values (dealing with missing values is not the focus of this work). The following datasets have been considered. Firstly, we focus on the size of the datasets, and four datasets with different number of patterns are generated from a Gaussian distribution: a small Gaussian dataset constituted by four classes and 30 patterns; two medium size Gaussian datasets (i.e., indexed with a and b in the second and third rows of the Tables 1, 3 and 4) formed by 60 and 120 patterns, respectively; a large Gaussian dataset constituted by 1,500 patterns. All these four Gaussian datasets are randomly sampled from a Gaussian distribution. This latter dataset deals with a large number of patterns, with the advantages of having only two features.

Table 1 Description of the datasets used in the experiments

	C	N	m	m_c
Small size Gaussian	4	2	30	15–5–5–5
Medium size Gaussian[a]	2	2	60	30–30
Medium size Gaussian[b]	2	2	120	60–60
Large size Gaussian	3	2	1,500	500–500–500
Iris	3	4	150	50–50–50
Wine	3	13	178	48–71–59
Glass	6	9	214	70–17–76–13–9–29
Statlog (heart)	2	13	270	150–120
Cone-torus	3	2	800	207–198–395
3-Spirals	3	2	1,200	400–400–400
Weaning	2	17	302	151–151
Laryngeal	3	16	353	53–218–82

Next, four well-known datasets in literature (i.e., Iris, Wine, Glass and Statlog Heart) taken from the UCI database repository [4] are used for classification. The number of features and classes are partially increased in these datasets.

Cone-torus dataset has 800 bi-dimensional patterns of three classes, generated from three differently shaped distributions with prior probabilities 0.25, 0.25 and 0.5, respectively [39]. This dataset is available as two different subsets for train and test, which are combined here to form a unique dataset for cross-validation.

A 3-spirals dataset contains 1,200 bi-dimensional patterns of three classes as shown in Fig. 3, exhibiting complex cluster shapes. This dataset is considered because of its complex decision boundaries.

Weaning is a real medical dataset [37], consisting of 302 patterns and 17 features representing two classes of patients distinguished as not ready/ready for weaning from mechanical ventilation.

Laryngeal is a real medical dataset [37], which contains 353 patterns of three classes representing normal/organic pathological/functional pathological voices. The set was originally used for diagnosing Laryngeal pathology in its early stages. The main difficulties with this dataset are to

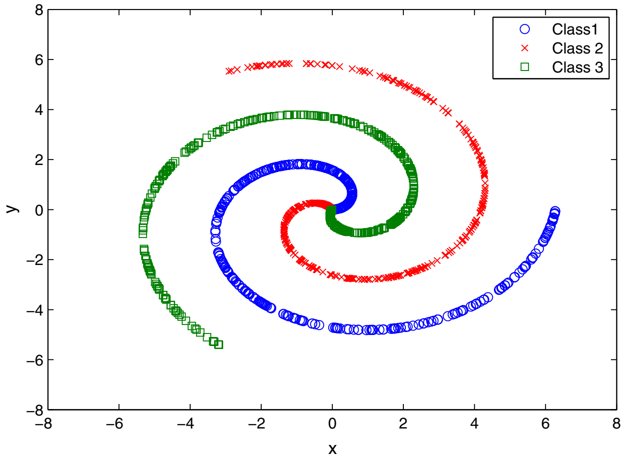

Fig. 3 3-Spirals dataset with three classes (400 patterns per class)

Table 2 The key parameters in the IWC algorithm

Parameters	Symbol	Value
Number of initial plants	P_0	5
Maximum number of iterations	$\text{iter}_{\max}$	200
Problem dimension	D	$C \times N$
Maximum number of plants	$P_{\max}$	30
Maximum number of seeds	$S_{\max}$	5
Minimum number of seeds	$S_{\min}$	0
Nonlinear modulation index	n	3
Initial value of standard deviation	σ_{initial}	1
Final value of standard deviation	σ_{final}	0.05

deal with the large number of features and imbalanced distribution of patterns between classes. However, this work does not focus on classification of class-imbalanced data.

The datasets are properly selected such that the number of patterns and features, number of classes, class distribution, noise level, correlation of features, decision boundaries and other characteristics vary to provide an adequately different and relatively complete classification benchmark.

The current version of the IWC just handles numerical data. It is possible for the IWC to classify categorical data, but the objective function should be modified. For instance, one can modify the distance-based objective function and define a distance measure based on co-occurrence of values instead of the Euclidean distance [2]. Albeit classification of the categorical datasets was not the aim of this paper, the distance-based objective function can be modified to handle these datasets as well, and we will address this in our future research.

4.1 IWC$-J$ versus IWC$-J'$

The IWC has some key parameters that impact on its performance. These parameters are tuned a priori, by rule of thumb, to guarantee good classification performance. The key parameters of concern and their proper values to perform the classification task are listed in Table 2.

The evolutions of the fittest individuals in the colony calculated by IWC $-\ J$ and IWC $-\ J'$ as a function of the number of iterations during an arbitrary cycle of cross-validation are shown in Fig. 4 for the case of large size Gaussian dataset.

The evolutionary profiles of the fittest individuals for IWC $-\ J$ and IWC $-\ J'$ show significant reductions in the first 60 iterations and then a linear decrease with much lower slopes that last until iteration 300. The value of the fittest individuals calculated by IWC $-\ J$ and IWC $-\ J'$ are 4.1 and 2.78, respectively, at the first iteration and reach to 0.52 and 0.49 in iteration 300. Albeit from iteration 60 to 300, the fittest individuals in IWC $-\ J$ and IWC $-\ J'$ have very close values, it is not expected that they converge to a same value, since IWC $-\ J'$ includes the complementary term of J_e.

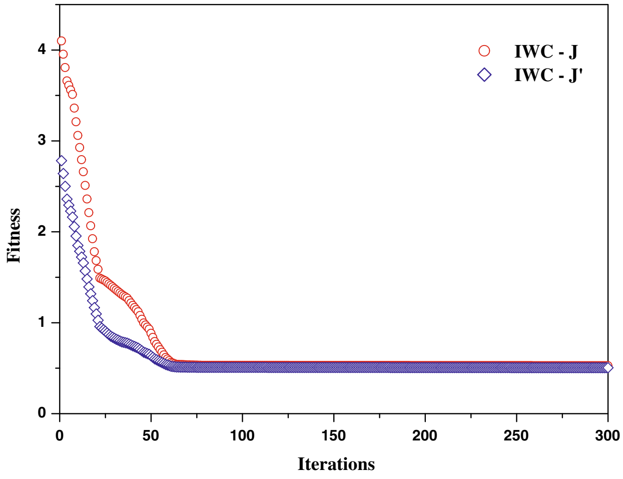

Fig. 4 The evolutionary profiles of the fittest individuals in the colony calculated by IWC $-\ J$ and IWC $-\ J'$ as a function of the number of iterations

This significant reduction during the first iterations can be seen in the classification of all datasets and is achieved through the proper tuning of the key parameters of the IWC algorithm.

Then, the performances of IWC $-\ J$ and IWC $-\ J'$ are compared with respect to the classification of the selected datasets. In this respect, their classification errors on test datasets are compared and reported in Table 3. The reported results are averaged over different iterations of cross-validation.

The IWC $-\ J'$ outperforms IWC $-\ J$ in terms of smaller error in most of the selected datasets, and its performance

Table 3 Comparison of averaged classification errors on test datasets

	$\overbrace{IWC - J}^{\bar{e}_{\text{test}}}$	$\overbrace{IWC - J'}^{\bar{e}_{\text{test}}}$
Small size Gaussian	3.1	2.5
Medium size Gaussian[a]	0	0
Medium size Gaussian[b]	0	0
Large size Gaussian	0.38	0.3
Iris	0.29	0.3
Wine	1.7	1.7
Glass	23.4	21.1
Statlog (heart)	8.9	9.3
Cone-torus	13.5	12.1
3-Spirals	18.4	17.9
Weaning	14.6	13.7
Laryngeal	10.5	9

is relatively close to IWC $-$ J in other cases, as Table 2 attests. Thus, it can be concluded that IWC $-$ J' on average outperforms IWC $-$ J and will be selected for further study in the rest of the paper. Hereafter, for the sake of simplicity, IWC is substituted for IWC $-$ J'.

4.2 IWC versus other classification methods

In this section, the performance of the IWC algorithm is compared to standard well-known pattern classifiers with respect to the classification of the selected datasets. This comparative analysis is done by resorting to widely used classifiers.

Firstly, among the Bayesian classifiers, the Naive Bayes (NB) has been selected [69]. Then, a powerful data mining tool, the decision tree (DT) method, is chosen [12]. Our choice among neural networks is the MultiLayer Perceptron (MLP) [11]. A supervised-FCM (S-FCM) is selected among fuzzy-based classifiers [8–10]. Linear discriminant analysis (LDA) and quadratic discriminant analysis (QDA) are selected among the statistical learning techniques [32, 43]. Among Kernel-based classifiers, the support vector machines (SVM) has been chosen [14, 17]. The radial basis function (RBF) has been used as a Kernel for the SVM. The k-nearest neighbor (KNN) algorithm is chosen as a well-known nonparametric technique from the computational geometry domain [5, 18]. Finally a Mamdani type neuro-fuzzy (M-NF) classifier has been selected among the rule-based techniques [57, 58].

The parameters of all classifiers, if any, are tuned properly to guarantee a fair comparison. The classification performances of different classifiers, at each iteration of cross-validation, are collected and illustrated in box-plot (see Fig. 5) for the Laryngeal dataset. The box-plot, in Fig. 5, presents the distributions of classification performances (percentage of number of correctly classified patterns/total number of patterns) for the patterns of the test subsets resulting from the Laryngeal dataset by means of the tenfold stratified cross-validation procedure.

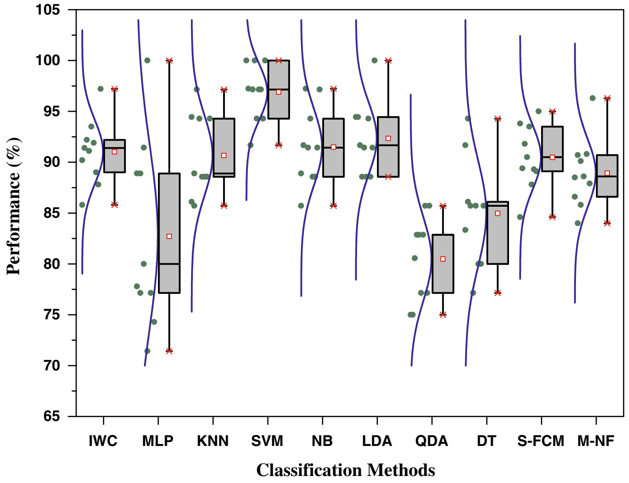

Fig. 5 *Box-plots* representing the distribution of classification performances (%) with respect to the patterns of the test subsets resulting from the Laryngeal dataset by means of the tenfold stratified cross-validation procedure. *Solid circles* represent the distribution of classification performances for each classification method. The *red dashes* stand for maximum and minimum performance values, and the *solid squares* denote the average performance for each classifier. The *red crosses* stand for 1 and 99 percentiles of the performance values (color figure online)

It is shown in box-plot (Fig. 5), that the average performance achieved by the support vector machine (SVM) is the highest and, conversely, the quadratic discriminant analysis (QDA) has the lowest average performance among all classifiers. MLP has the largest range of variation in the performance values among all classifiers with respect to the classification of the patterns of the Laryngeal dataset. The IWC has the minimum range of variation in the performance values and ranked fourth based on the average performance among all classifiers. For this dataset, the linear discriminant analysis and naive Bayes classifiers outperform compared to IWC with respect to the classification of the Laryngeal dataset and one ranked second and third, respectively.

Table 4 reports the average performance (%) obtained by different classifiers in a k-fold cross-validation scheme on each dataset. Classification performance is calculated with respect to the percentage of the correctly classified patterns on the test subsets at each iteration of cross-validation.

In the table, the entries highlighted in italic at each row denote the highest performance achieved among all classifiers on that dataset, while the bold entries stand for the lowest performance. A brief discussion on the achieved results is outlined below.

Table 4 Average performances in a *k*-fold cross-validation with respect to the percentage of the correctly classified patterns on each dataset

	IWC	MLP	KNN	SVM	NB	LDA	QDA	DT	S-FCM	M-NF
Small size Gaussian	*97.5*	85.7	96.7	90.0	90.0	86.7	86.7	**76.7**	93.4	91.7
Medium size Gaussian[a]	*100*	96.4	98.3	*100*	98.3	*100*	98.3	96.7	95.7	**94.8**
Medium size Gaussian[b]	*100*	*100*	**99.2**	**99.2**	98.3	98.3	98.3	98.3	**99.2**	*100*
Large size Gaussian	*99.7*	*99.7*	**99.6**	**99.6**	**99.6**	*99.7*	*99.7*	*99.7*	**99.6**	**99.6**
Iris	*99.7*	98.0	97.3	97.3	98.0	98.7	98.0	96.7	**94.3**	*99.7*
Wine	98.3	99.4	98.9	**91.5**	98.3	*100*	*100*	94.9	92.3	94.6
Glass	78.9	84.1	89.7	*90.1*	**76.7**	86.4	77.0	82.6	82.3	83.9
Statlog (heart)	90.7	91.7	92.2	*98.1*	93.3	92.9	93.3	89.3	**85.7**	94.5
Cone-torus	87.9	81.8	*95.0*	90.5	85.8	86.5	85.8	91.8	**80.1**	83.3
3-Spirals	82.1	84.7	*100*	*100*	73.0	67.3	71.8	*100*	**55.7**	86.5
Weaning	86.3	93.5	*98.3*	**75.4**	96.0	91.7	94.0	92.7	80.1	95.0
Laryngeal	91.0	82.7	90.7	*96.9*	91.5	92.3	**80.5**	85.0	90.5	88.9

It is shown in Table 4, that the IWC is on average relatively effective with respect to the classification of the selected datasets. Albeit invasive weed is an optimization technique, it is competitive against other methods particularly tailored for classification tasks. IWC has the maximum average performance in the classification of five datasets and most important is that the IWC has never been ranked worst, i.e., the minimum average performance. As seen in Table 4, the IWC along with SVM have the maximum number of highest ranks, due to maximum average performance on the five classified datasets.

The performance of the invasive weed algorithm as an optimization tool is highly tailored to its key parameters. The IWC parameters are properly tuned; however, the problem dimension $D = C \times N$ has a huge impact on the classification performance. It is assumed that the lower dimension may lead to the higher performance. However, this assumption is not fully supported by the attained results, as it is shown in Table 4. Albeit the average performance of the IWC is superior for the first five datasets, where the problem dimension is small, the IWC average performance is not the best for other two class problems, i.e., in cone-torus and 3-spirals datasets. Conversely, the IWC outperforms other techniques and is among the top 4 in the classification of the high dimensional Laryngeal dataset.

It can be seen that there does not exist any direct relation between the IWC performance and the size of dataset $N \times m$, since it shows the highest performance for the large size Gaussian dataset and is among the highest ranks in the classification of the Laryngeal dataset.

The IWC is ranked fourth and fifth with respect to classification of low dimensional problems of cone-torus and 3-spirals datasets, respectively, which is due to the fact that these two datasets contain complex classification decision boundaries. This limitation arises from the multi-objective function of the IWC, which is class oriented and merely constructed to optimize the positions of the centroids in the future space. This behavior can be seen in other centroid-oriented classifiers, such as S-FCM.

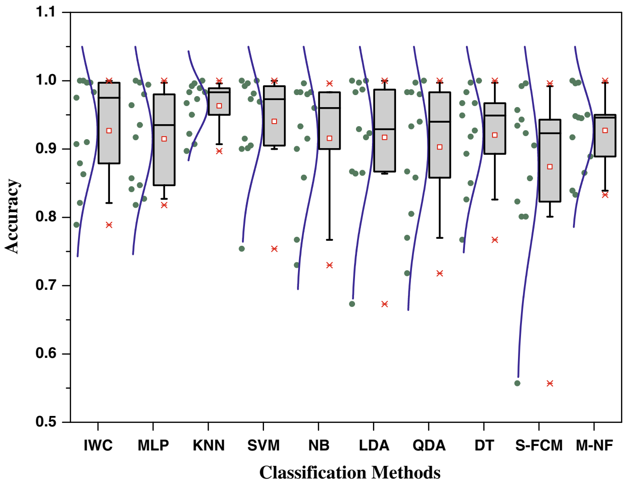

Fig. 6 *Box-plots* representing the distribution of average performances or accuracy on each dataset. *Solid circles* represent the distribution of average performances or accuracy on each dataset. The *red dashes* stand for maximum and minimum values, and the *solid squares* denote the average of average performances on all datasets. The *red crosses* stand for 1 and 99 percentiles (color figure online)

Indeed, the complex decision boundaries such as spirals and rings cannot be easily spanned by centroid-oriented decision functions since the class concept based on the distance from a centroid cannot separate the surrounding region correctly. More centroids or association of different centroids to subclasses might be necessary to handle such a situation. This work could also be extended to add another term to the multi-objective function, which looks to be a worthwhile direction for future research in order to overcome this limitation.

The average performances of different classifiers with respect to the classification of all datasets, as collected in Table 4, are illustrated in the box-plot in Fig. 6. The box-plot in Fig. 6 illustrates the distributions of the average performances of different classifiers over the *k*-fold cross-validation with respect to classification of each dataset.

There exist statistical differences between the distributions of the average performances or accuracies. The distribution of the average performances on different datasets shows the largest variation for the S-FCM. KNN has the minimum range of variation that implies the steadiest performance on all datasets among other classifiers. The highest median corresponds to the KNN classifier, which shows that the KNN classifier has a superior average performance on the majority of the datasets.

Finally, the classifiers are ranked based on the average over the average performances on each dataset, which are shown by solid squares in the box-plots of Fig. 6 for each classifier. These average values over the average performances are reported in the first row of Table 5.

The classifiers are finally ranked based on these values in order to present their efficiency in the classification of all selected datasets. The final rank is reported in the second row of Table 5. The KNN classifier, with the average value of 0.963, is ranked first in the classification of all selected datasets among all other classifiers, and is highlighted in italic in the Table. KNN is followed by SVM and M-NF as second and third ranks, respectively. IWC is ranked fourth, with a superior average performance of 0.926, over all selected datasets, which is pretty close to the average performance of the top three classifiers. The final rank is followed by other classifiers, such as DT, LDA, NB, MLP and QDA, with total accuracy over 0.9. The worst rank, S-FCM, is the only classifier with total accuracy less than 0.9 and is highlighted in bold in Table 5.

Albeit SVM achieves maximum performance with respect to the classification of the five datasets (Table 4), which is the highest record among all classifiers, it is ranked as the second best classifier after KNN (Table 5). The distribution of the values of the average over the average performances on all datasets, box-plots in Fig. 6, reveals a larger range of variation for SVM compared to KNN. This is due to the weak performance of SVM with respect to the classification of the Weaning dataset, that leads to a lower value of the final average for SVM comparing to KNN.

Even though methods specifically tailored for classification, such as KNN and SVM, show a superior performance in classification of all selected datasets on average, the IWC performance is still comparable with the top two classifiers and ranked fourth among all selected classifiers.

The time needed for these classifiers to be trained depends on the problem dimension and varies in a range from a few seconds up to a few minutes. However, the IWC has an acceptable rank in terms of the training time among all other classifiers.

4.3 Classification with noisy data

In this section, the performance of the IWC algorithm is compared to standard well-known pattern classifiers (i.e., those used in Sect. 4.2) with respect to the classification of noisy datasets. A summary of the noisy datasets characteristics is reported in Table 6.

Firstly, we focus on the datasets that contain some noisy features such as: Difficult Doughnut, Easy Doughnut and Four Gaussian [38, 40]. All these three datasets contain 10 noisy features N_n. Then, different types of 2-dimensional noisy datasets are selected. The easiest is the Noisy Lines dataset, which is available in [40].

The noisy datasets with more complex decision boundaries are the 2-Noisy Spirals and the Noisy Pinwheels datasets, with different number of classes and complexity. The 2-Noisy Spirals dataset contains 2,000 bi-dimensional patterns of two classes as shown in Fig. 7a. The Noisy Pinwheel datasets with two, three, four and five classes are shown in Fig. 7b–f, respectively. There were 1,000 data points for each class in each dataset (see Table 6). These pinwheels are merely Gaussian datasets that have been stretched and rotated [1]. Two different 3-Class Noisy Pinwheel datasets are used in this experiment. The '3-Class Noisy Pinwheel^T' stands for the one with more twisted arms.

Table 7 reports the average performance (%) obtained by different classifiers in a tenfold cross-validation scheme on each noisy dataset. In Table 7, the entries highlighted in italic at each row denote the highest performance achieved among all classifiers on that noisy dataset, while the bold entries stand for the lowest performance. A brief discussion on the obtained results on the noisy datasets is outlined below.

As shown in Table 7, the IWC is relatively effective with respect to the classification of the noisy datasets. The IWC is stable and ranked around middle among all other classifiers with respect to each noisy dataset (see each row of the Table 7). The IWC has never been ranked worst, i.e., the minimum average performance in this experiment.

As seen in Table 7, KNN has the maximum number of highest ranks, due to maximum average performance on the

Table 5 Final rank based on the average over the average performance values

	IWC	MLP	KNN	SVM	NB	LDA	QDA	DT	S-FCM	M-NF
Average	0.926	0.914	*0.963*	0.940	0.915	0.917	0.902	0.920	**0.874**	0.927
Rank	4	8	*1*	2	7	6	9	5	**10**	3

Table 6 Description of the noisy datasets used in the experiments

	C	N	m	m_c	N_n
Difficult doughnut	2	12	400	200–200	10
Easy doughnut	2	12	400	200–200	10
Four Gaussian	4	12	800	200–200–200–200	10
Noisy lines	2	2	200	100–100	–
2-Noisy spirals	2	2	2,000	1,000–1,000	–
2-Class noisy pinwheel	2	2	2,000	1,000–1,000	–
3-Class noisy pinwheel	3	2	3,000	1,000–1,000–1,000	–
3-Class noisy pinwheel^T	3	2	3,000	1,000–1,000–1,000	–
4-Class noisy pinwheel	4	2	4,000	1,000–1,000–1,000–1,000	–
5-Class noisy pinwheel	5	2	5,000	1,000–1,000–1,000–1,000–1,000	–

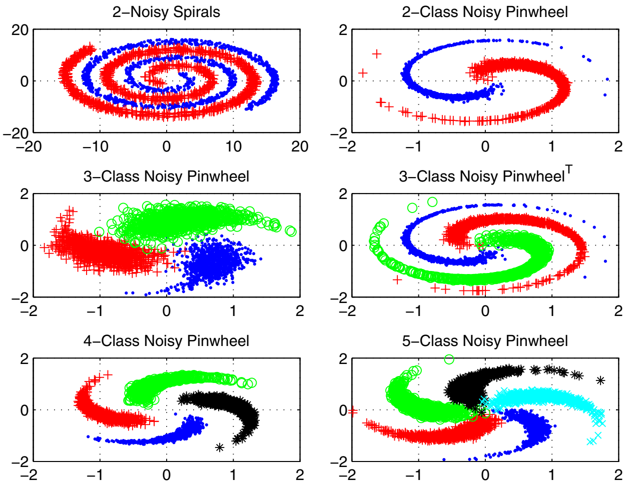

Fig. 7 2-Noisy spirals and noisy pinwheels datasets

seven noisy datasets. Conversely, S-FCM has the maximum number of lowest ranks on the last six noisy datasets.

Albeit SVM achieves reasonable performances with respect to the classification of the six noisy datasets (Table 7), and it is ranked as the ninth classifier (Table 8), due to its weak performance with respect to the datasets with noisy features, $N_n > 0$ (first three rows in Tables 6 and 7).

DT achieves a good performance in almost all noisy datasets. It ranked first among all other classifiers on the classification of the noisy datasets. DT even outperforms KNN that ranked first on the seven noisy datasets due to its stable performance on all noisy datasets, especially on the Four Gaussian dataset (third row in Table 7).

These average values over the average performances are reported in the first row of Table 8. The classifiers are finally ranked based on these values in order to present their performance in the classification of all noisy datasets. The final rank is reported in the second row of Table 8. The DT classifier, with the average value of 0.994, is ranked first in all noisy experiments among all other classifiers, and is highlighted in italic in the Table 8.

DT is followed by KNN and NB as second and third ranks, respectively. IWC is ranked fourth, with a superior average performance of 0.948 over all the noisy datasets. The final rank is followed by other classifiers, such as the QDA, MLP, M-NF and LDA, with a total accuracy over 0.9. The lowest ranks, the SVM and S-FCM are the only classifiers with total accuracy less than 0.9.

The average performances of different classifiers with respect to the classification of all datasets (i.e., the 13 datasets of the experiments in Sect. 4.2 and the 10 noisy datasets of the experiments in Sect. 4.3), as collected in Tables 4 and 7, are illustrated in the box-plot in Fig. 8. The box-plot in Fig. 8 shows the distributions of the average performances of different classifiers on the k-fold cross-validation with respect to the classification performance in all experiments.

In conclusion, the classifiers are ranked based on the average over the average performances on each dataset (all 22 datasets), which are shown by solid squares in the box-plots of Fig. 8 for each classification method. These average values over the all average performances are reported in the first row of Table 9. The final rank over all 22 datasets is reported in the second row of Table 9.

The KNN classifier, with the average value of 0.976, is ranked first in the classification of all the 22 datasets among all other classifiers. KNN is followed by DT as second rank. IWC is ranked third, with superior average performance (0.936) over all 22 datasets. The final rank over all the 22 datasets is followed by other classifiers, such as the NB, MLP, M-NF, QDA and LDA, with total accuracy over 0.9. The worst ranks, the SVM and S-FCM have a total accuracy less than 0.9.

The box-plot in Fig. 8 reveals a large range of variation for SVM and S-FCM compared to KNN. The IWC has an acceptable range of variation (0.211), which leads to a stable performance over all the datasets including the noisy experiments.

5 Discussion and conclusion

In this paper, an invasive weed optimization technique has been adapted to perform as a pattern classifier, called the invasive weed classification (IWC). The classification task was formulated as a multi-objective optimization problem, for which three objective functions were defined in this work. However, the proposed IWC is mainly a

Table 7 Average performances in a tenfold cross-validation with respect to the percentage of the correctly classified patterns on each noisy dataset

	IWC	MLP	KNN	SVM	NB	LDA	QDA	DT	S-FCM	M-NF
Difficult doughnut	94.0	97.8	98.5	**57.3**	*99.3*	74.9	97.5	99.0	90.0	91.5
Easy doughnut	97.8	99.0	*99.8*	**56.5**	99.0	77.2	98.3	99.3	91.7	93.3
Four Gaussian	99.5	96.1	94.4	**46.6**	*99.6*	*99.6*	*99.6*	99.4	96.7	96.1
Noisy lines	*100*	**99.5**	*100*	*100*	*100*	*100*	*100*	*100*	**99.5**	*100*
2-Noisy spirals	79.1	86.9	*100*	81.1	73.0	79.4	70.1	98.6	**53.1**	79.5
2-Class noisy pinwheel	94.5	94.7	*99.9*	99.7	96.3	96.9	96.2	99.6	**67.5**	94.3
3-Class noisy pinwheel	99.5	97.2	*99.8*	99.7	99.7	99.5	99.7	99.7	**96.1**	97.0
3-Class noisy pinwheel^T	90.0	92.2	99.5	*99.7*	88.1	89.4	87.5	99.4	**57.1**	81.07
4-Class noisy pinwheel	94.5	97.1	*100*	*100*	99.0	99.1	99.8	99.9	**61.8**	98.5
5-Class noisy pinwheel	99.4	86.1	*99.8*	*99.8*	99.4	99.0	99.6	99.7	**74.5**	95.9

Table 8 Final rank based on the average over the average performance values on the noisy datasets

	IWC	MLP	KNN	SVM	NB	LDA	QDA	DT	S-FCM	M-NF
Average	0.9483	0.9465	0.9915	0.8403	0.9532	0.9151	0.9482	*0.9945*	**0.7880**	0.9271
Rank	4	6	2	9	3	8	5	*1*	**10**	7

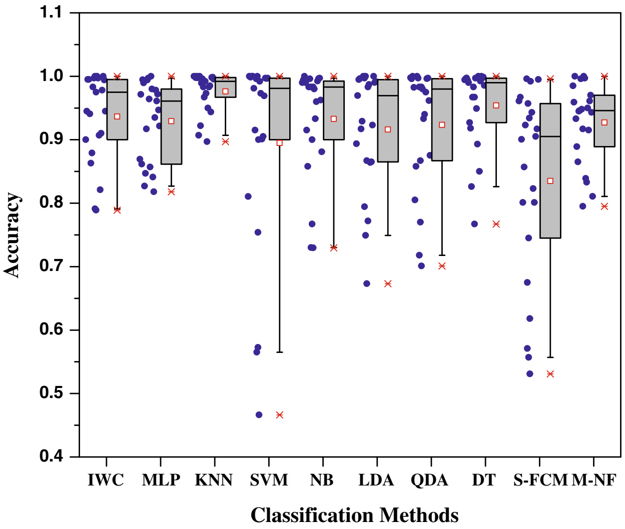

Fig. 8 *Box-plots* representing the distribution of average performances on each dataset (i.e., including the 12 datasets of the experiments in Sect. 4.2 and the 10 noisy datasets of the experiments in Sect. 4.3). *Solid circles* represent the distribution of average performances on each dataset. The *red dashes* stand for maximum and minimum values, and the *solid squares* denote the average of average performances on all datasets. The *red crosses* stand for 1 and 99 percentiles (color figure online)

centroid-oriented classifier: it mainly assigns the positions of the optimal plants to the class centroids. Initial seeds are being randomly spread onto the feature space and their fitness values are calculated with a distance-based function. Those plants with minimum fitness are selected as class centroids and become reproductive iteratively until the optimization goal has been met.

Another term has been devised in the multi-objective function to avoid overtraining.

Different versions of the IWC are firstly compared each other and then against other well-known classifiers. For this comparative analysis, 22 different datasets have been selected for the classification task, different in terms of size, complexity of their decision boundaries and noise level. The attained results and careful analysis show that the IWC is an efficient classifier, which performs well when compared against other well-known classifiers.

It is shown that the IWC outperforms other classifiers where the decision boundaries can be spanned based on the distance from the centroids, but its performance is slightly reduced when classifying complex decision boundaries.

To overcome this limitation, the multi-objective function could be extended by adding a non-centroid function, which looks to be a worthwhile direction for future research.

Table 9 Final rank based on the average over the all average performance values

	IWC	MLP	KNN	SVM	NB	LDA	QDA	DT	S-FCM	M-NF
Average	0.9365	0.9291	*0.9761*	0.8949	0.9327	0.9161	0.9234	0.9540	**0.8349**	0.9271
Rank	3	5	*1*	9	4	8	7	2	**10**	6

References

1. Adams RP, Ghahramani Z (2009) Archipelago: nonparametric bayesian semi-supervised learning. In: Danyluk AP, Bottou L, Littman ML (eds) Proceedings of the 26th international conference on machine learning ICML. ACM, Montreal, p 1
2. Ahmad A, Dey L (2007) A k-mean clustering algorithm for mixed numeric and categorical data. Data Knowl Eng 63(2):503–527
3. An A (2003) Learning classification rules from data. Comput Math Appl 45:737–748
4. Bache K, Lichman M (2013) UCI machine learning repository. Technical report, University of California, School of Information and Computer Science, Irvine, CA
5. Bailey T, Jain A (1978) A note on distance-weighted k-nearest neighbor rules. IEEE Trans Syst Man Cybern 8:311–313
6. Bandyopadhyay S, Murthy C, Pal S (1995) Pattern classification with genetic algorithms. Pattern Recogn Lett 16:801–808
7. Bandyopadhyay S, Murthy C, Pal SK (1998) Pattern classification using genetic algorithms: determination of H. Pattern Recogn Lett 19(13):1171–1181
8. Baraldi P, Razavi-Far R, Zio E (2010). A method for estimating the confidence in the identification of nuclear transients by a bagged ensemble of FCM classifiers. In: Proc. NPIC&HMIT, pp. 283–293. Las Vegas, NV
9. Baraldi P, Razavi-Far R, Zio E (2011) Bagged ensemble of fcm classifier for nuclear transient identification. Ann Nucl Energy 38(5):1161–1171
10. Baraldi P, Razavi-Far R, Zio E (2011) Classifier-ensemble incremental-learning procedure for nuclear transient identification at different operational conditions. Reliab Eng Syst Saf 96(4):480–488
11. Bishop C (1995) Neural networks for pattern recognition. Oxford University, Oxford
12. Breiman L, Friedman J, Olshen R, Stone C (1984) Classification and regression trees. Wadsworth, Belmont
13. Cervantes A, Galvan I, Isasi P (2009) Ampso: a new particle swarm method for nearest neighborhood classification. IEEE Trans Syst Man Cybern B Cybern 39(5):1082–1091
14. Chang CC, Lin CJ (2011) Libsvm: a library for support vector machines. ACM Trans Intell Syst Technol 2(3):27:1–27:27. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm
15. Chi-Yang T, Yeh SW (2008) A multiple objective particle swarm optimization approach for inventory classification. Int J Prod Econ 114:656–666
16. Chowdhury A, Bose B, Das S (2011) Automatic clustering based on invasive weed optimization algorithm. In: Proceedings of the second international conference on Swarm, Evolutionary, and Memetic Computing (SEMCCO'11), vol. Volume II, pp 105–112
17. Cortes C, Vapnik V (1995) Support-vector networks. Mach Learn 20:273–297
18. Cover T, Hart P (1967) Nearest neighbor pattern classification. IEEE Trans Inf Theory IT 13(1):21–27
19. Efron B (1983) Estimating the error rate of a prediction rule: improvement on cross-validation. J Am Stat Assoc 78(382):316–331
20. Efron B, Tibshirani R (1995) Improvements on cross-validation: the bootstrap method. J Am Stat Assoc 92(43):548–560
21. De Falco I, Della Cioppa A, Tarantino E (2006) Evaluation of particle swarm optimization effectiveness in classification. In: Bloch I, Petrosino A, Tettamanzi AGB (eds) Fuzzy logic and applications, Lecture notes in computer science, vol 3849. Springer, Berlin, Heidelberg, pp 164–171
22. Falco ID, Cioppa AD, Tarontino E (2007) Facing classification problems with particle swarm optimization. Appl Soft Comput 7:652–658
23. Fisher RA (1936) The use of multiple measurements in taxonomic problems. Ann Eugen 7(2):179–188
24. Friedman N, Geiger D, Goldszmidt M (1997) Bayesian network classifiers. Mach Learn 29:131–163
25. Ghalenoei M, Hajimirsadeghi H, Lucas C (2009). Discrete invasive weed optimization algorithm: application to cooperative multiple task assignment of uavs. In: Proceedings of the IEEE conference on decision and control, pp 1665–1670
26. Green T, Izadi-Zamanabadi R, Razavi-Far R, Niemann H (2014) Plant-wide dynamic and static optimisation of supermarket refrigeration systems. Int J Refrig 38:106–117
27. Green T, Razavi-Far R, Izadi-Zamanazabdi R, Niemann H (2012) Plant-wide performance optimization, the refrigeration system case. In: IEEE multi-conference on systems and control (MSC). Dubrovnik, Croatia, pp 208–213
28. Grzymala-Busse JW (1992) LERS-A system for learning from examples based on rough sets. In: Słowiński R (ed) Intelligent decision support: handbook of applications and advances of rough sets theory. Theory and Decision Library, vol 11. Springer, Netherlands, pp 3–18
29. Hajimirsadeghi H, Ghazanfari A, Rahimi-Kian A, Lucas C (2009) Cooperative coevolutionary invasive weed optimization and its application to nash equilibrium search in electricity markets. In: World congress on nature and biologically inspired computing, NABIC, pp 1532–1535
30. Hajimirsadeghi H, Lucas C (2009) A hybrid iwo/pso algorithm for fast and global optimization. In: IEEE Eurocon, pp 1964–1971
31. Hansen L, Salamon P (1990) Neural network ensembles. IEEE Trans Pattern Anal Mach Intell 12(10):993–1001
32. Hastie T, Tibshirani R, Friedman J (2008) The elements of statistical learning. Springer, Berlin, New York
33. He X, Wang Z, Jin C, Zheng Y, Xue X (2012) A simplified multi-class support vector machine with reduced dual optimization. Pattern Recogn Lett 33(1):71–82
34. Karimkashi S, Kishk AA (2010) Invasive weed optimization and its features in electromagnetics. IEEE Trans Antennas Propag 58(4):1269–1278
35. Kittler J, Hatef M, Duin R, Mates J (1998) On combining classifiers. IEEE Trans Pattern Anal Mach Intell 20(3):226–239
36. Kohavi R (1995) A study of cross-validation and bootstrap for accuracy estimation and model selection. In: Proceedings of the 14th international joint conference on artificial intelligence, vol 2, IJCAI 95Morgan Kaufmann, Montréal Québec, Canada, pp 1137–1145
37. Kuncheva L (2005) Real medical datasets. Technical report, School of Informatics. University of Wales, Bangor, UK
38. Kuncheva L, Hadjitodorov S (2004) Using diversity in cluster ensembles. In: IEEE international conference on systems, man and cybernetics, 2004, vol 2, pp 1214–1219
39. Kuncheva L, Whitaker C (2002) Using diversity with three variants of boosting: aggressive, conservative, and inverse. In: Roli F, Kittler J (eds) Multiple classifier systems, Lecture notes in computer science, vol 2364. Springer, Berlin, pp 81–90
40. Kuncheva LI, Vetrov DP (2006) Evaluation of stability of k-means cluster ensembles with respect to random initialization. IEEE Trans Pattern Anal Mach Intell 28(11):1798–1808
41. Li B, Liu PY, Hu RX, Mi SS, Fu JP (2012) Fuzzy lattice classifier and its application to bearing fault diagnosis. Appl Soft Comput 12(6):1708–1719
42. McGovern A, Jensen D (2008) Optimistic pruning for multiple instance learning. Pattern Recogn Lett 29(9):1252–1260
43. McLachlan G (2004) Discriminant analysis and statistical pattern recognition. Wiley Interscience, London
44. Mehrabian AR, Lucas C (2006) A novel numerical optimization algorithm inspired from weed colonization. Ecol Inform 1:355–366

45. Mehrabian AR, Yousefi-Koma A (2007) Optimal positioning of piezoelectric actuators on a smart fin using bio-inspired algorithms. Aerosp Sci Technol 11:174–182
46. Mitrakis N, Theocharis J, Petridis V (2008) A multilayered neuro-fuzzy classifier with self-organizing properties. Fuzzy Sets Syst 159(23):3132–3159
47. Nikoofard A, Hajimirsadeghi H, Rahimi-Kian A, Lucas C (2012) Multiobjective invasive weed optimization: application to analysis of pareto improvement models in electricity markets. Appl Soft Comput 12:100–112
48. Nouaouria N, Boukadoum M (2010) Particle swarm classification for high dimensional datasets. In: Proceedings of 22th international IEEE conference on tools with artificial intelligence IEEE-ICTAI, vol 1, Arras, France, pp 87–93
49. Nouaouria N, Boukadoum M (2011) A particle swarm optimization approach to mixed attribute data-set classification. In: Proceedings of IEEE symposiumon swarm intelligence. Paris, France, pp 44–51
50. Nouaouria N, Boukadoum M, Proulx R (2013) Particle swarm classification: a survey and positioning. Pattern Recogn 46(7): 2028–2044
51. Omran M, Al-Sharhan A (2007) Barebones particle swarm methods for unsupervised image classification. In: Proceedings of IEEE congress on evolutionary computation (CEC). Singapore, pp 3247–3252
52. ONeill M, Brabazon A (2006) Self-organizing swarm (SO-Swarm): a particle swarm algorithm for unsupervised learning. In: Proceedings of IEEE congress on evolutionary computation. Vancouver, BC, Canada, pp 634–639
53. Owechko Y, Medasani S, Srinivasa N (2004) Classifier swarms for human detection in infrared imagery. In: Proceedings of IEEE computer society conference on computer vision and pattern recognition workshops (CVPRW04)
54. Rad H, Lucas C (2007) A recommender system based on invasive weed optimization algorithm. In: IEEE congress on evolutionary computation, CEC, pp 4297–4304
55. Razavi-Far R, Baraldi P, Zio E (2012) Dynamic weighting ensembles for incremental learning and diagnosing new concept class faults in nuclear power systems. IEEE Trans Nucl Sci 59(5):2520–2530
56. Razavi-Far R, Baraldi P, Zio E (2012) Ensemble of neural networks for detection and classification of faults in nuclear power systems. In: Proceedings of 10th international FLINS conference on uncertainty modeling in knowledge engineering and decision making, world scientific proc. ser. on computer engineering and information science, vol 7. Istanbul, Turkey, pp 1202–1207
57. Razavi-Far R, Davilu H, Palade V, Lucas C (2009) Model-based fault detection and isolation of a steam generator using neuro-fuzzy networks. Neurocomput J 72(13–15):2939–2951
58. Razavi-Far R, Davilu H, Palade V, Lucas C (2009) Neuro-fuzzy based fault diagnosis of a steam generator. In: 7-th IFAC conference on fault detection, supervision and safety of technical processes, international federation of automatic control (IFAC). Barcelona, Spain, pp 1180–1185
59. Razavi-Far R, Kinnaert M (2012) Incremental design of a decision system for residual evaluation: a wind turbine application. In: 8-th IFAC conference on fault detection, supervision and safety of technical processes, international federation of automatic control. Mexico City, pp 343–348
60. Razavi-Far R, Kinnaert M (2013) A multiple observers and dynamic weighting ensembles scheme for diagnosing new class faults in wind turbines. Control Eng Pract 21(9):1165–1177
61. Razavi-Far R, Palade V, Sun J (2013) Optimizing performance of a refrigeration system using an invasive weed optimization algorithm. In: Combinations of intelligent methods and applications, Springer series book: smart innovation, systems and technologies. In: Proceedings of CIMA12—the 3rd international workshop on combinations of intelligent methods and applications, in conjunction with the 22nd European conference on artificial intelligence—ECAI, vol 23. Springer, Montpellier, France, pp 79–93
62. Razavi-Far R, Palade V, Zio E (2014) Optimal detection of new classes of faults by an invasive weed optimization method. In: Proceedings of the IEEE World Congress on Computational Intelligence - IEEE WCCI. Beijing, China
63. Razavi-Far R, Zio E, Palade V (2014) Efficient residuals preprocessing for diagnosing multi-class faults in a doubly fed induction generator, under missing data scenarios. Expert Syst Appl 41(14):6386–6399
64. Rodriguez J, Perez A, Lozano J (2010) Sensitivity analysis of k-fold cross validation in prediction error estimation. IEEE Trans Pattern Anal Mach Intell 32(3):569–575
65. Schaffe C (1993) Selecting a classification method by cross-validation. Mach Learn 13:135–143
66. Serapiao A, Mendes J (2009) Classification of petroleum well drilling operations with a hybrid particle swarm/ant colony algorithm. In: Chien B (ed) Proceedings of IEA/AIE, LNAI5579. Springer, Berlin/Heidelberg, pp 301–310
67. Sousa T, Silva A, Neves A (2004) Particle swarm based data mining algorithms for classification tasks. Parallel Comput 30:767–783
68. Tewolde G, Hanna D (2007) Particle swarm optimization for classification of breast cancer data using single and multi surface methods of data separation. In: Proceedings of IEEE IT, pp 443–446
69. Zhang H (2004) The optimality of naive bayes. In: Barr V, Markov Z (eds) FLAIRS Conference: Proceedings of the 7th international florida artificial intelligence research society conference. AAAI Press, Miami Beach, Florida, USA, pp 562–567