IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 6, 2020, accepted July 14, 2020, date of publication July 22, 2020, date of current version July 31, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3011153

Impact of Solution Representation in Nature-Inspired Algorithms for Feature Selection

UROŠ MLAKAR, (Member, IEEE), IZTOK FISTER, JR., (Member, IEEE), AND IZTOK FISTER, (Member, IEEE)

Faculty of Electrical Engineering and Computer Science, University of Maribor, 2000 Maribor, Slovenia

Corresponding author: Uroš Mlakar (uros.mlakar@um.si)

This work was supported by the Slovenian Research Agency under Grant J2-1731, Grant L7-9421, and Grant P2-0041.

ABSTRACT This paper investigates how does the solution representation in nature-inspired algorithms impact the performance of feature selection in classification problems. Four most suitable nature-inspired algorithms for feature selection were considered in the analysis, namely the Differential Evolution, Artificial Bee Colony, Particle Swarm Optimization, and Genetic Algorithm. The binary-coded and real-coded variants of the mentioned algorithms were compared for filter-based and wrapper-based feature selection methodologies on datasets commonly used by the research community. Additionally, the algorithms' performance on reducing the feature subset size regarding different solution representations was compared. Statistical tests were performed for discovering any significant differences in the algorithms' performances.

INDEX TERMS Classification, evolutionary algorithms, feature selection, solution representation.

I. INTRODUCTION

The volume of data needed for classification has been increasing daily. These data are said to be remarkable in both the number of data samples and the number of attributes/features within each sample, which represents a prominent problem for any learning algorithm either supervised or unsupervised [1]. To increase the performance of the classification algorithm, and, on the other hand, also decrease the training time, several methods have been proposed for dimensionality reduction (i.e., feature selection [2]). Indeed, the goal of feature selection is to select a subset of the most relevant features without incurring loss of information. This algorithm can be found in many application areas which are relevant to intelligent and expert systems, such as data mining [3] and machine learning [4], [5], image processing [6], bioinformatics [7] etc. It is normally treated as a data preprocessing step similar to training a model for classification previously.

Two methods for feature selection are found commonly in literature, namely wrapper-based [6] and filter-based [8]. The main difference between the listed methods is in evaluating the selected feature subsets. Wrapper-based methods rely on a learning algorithm to determine the fitness of each feature subset. Among the more used learning algorithms are Support Vector Machines (SVM), Decision Trees (DT), k-Nearest Neighbor classifier (kNN), etc. On the other hand, filter-based methods rely on the characteristics of data for ranking feature subsets (i.e. Joint Mutual Information [9], where the feature selection is regarded as a preprocessing step. The latter are computationally cheaper, and also independent of any learning algorithm. In this method, a learning algorithm is employed only in the validation stage, when the optimal subset of features is already known.

For a dataset with larger number of features the exhaustive search, enumerating all possible solutions, is not eligible to find an optimal subset of features due to a too large search space of 2^m, where m denotes the number of features. Therefore, computationally efficient methods are needed, capable of good global search ability (exploration). This ability is considered mostly within the nature-inspired algorithms, to which Evolutionary Algorithms (EAs) [10], and Swarm Intelligence (SI) based [11] algorithms belong. Many studies of applying these algorithms for feature selection in classification tasks have been made in the last couple of years [12], where the Genetic Algorithm (GA) [13], Particle Swarm Optimization (PSO) [14], Differential Evolution (DE) [15],

The associate editor coordinating the review of this manuscript and approving it for publication was Halil Yetgin.

and Artificial Bee Colony (ABC) [16] are distinguished as the suitable algorithms for solving this problem according to a recent survey [12]. As a result, these algorithms were used in our comparative study as well.

The following review of papers referring to this domain additionally justify the suitability of the mentioned algorithms for the performed study. Indeed, Ghosh *et al.* [17] applied a DE algorithm with adaptive control parameters to feature selection. Their results showed that their proposed algorithm outperformed the GA and regular DE algorithms in an image classification problem. Derrac *et al.* [18] proposed a coevolutionary GA algorithm for feature selection which used three populations. Their method is focused on both feature selection and instance selection in a single process, which reduced the computational time. Their approach is useful for datasets with a large number of features. Zhang *et al.* [19] implemented a hybrid PSO algorithm for unsupervised feature selection. They applied two filter-based strategies for improving the convergence of the algorithm. The first strategy is employed for removing the irrelevant features from the original dataset, while the latter removes the redundant features within the already selected feature subset. They test their method on typical classification datasets, and justify the superiority and effectiveness of their hybrid PSO.

Interestingly, there are also some drawbacks of these algorithms. Some papers reported that scalability is one of the key limitations of these algorithms [20], [21]. This comes to the fore, especially when we are faced with thousands or more features. Further, it is worth to mention that parameter settings of these algorithms have a big influence on the results of the optimization. To find an appropriate combination of parameter settings for a particular optimization problem may therefore be time consuming and may involve a lot of experiments [22].

In summary, we were focused on the nature-inspired algorithms suitable for feature selection. Although we could also include state-of-the-art algorithms like jSO [23] and lShade [24] into the comparative study, the No Free Lunch (NFL) theorem [25] usually prevents to improve the results of proven nature-inspired algorithms for solving this problem.

In classification, the goal of feature selection is to reduce the number of used features, and at the same time, maintaining the same, or achieving a better classifier accuracy. Indeed, the problem can be defined either as single or multi-objective. Because of simplicity, the majority of authors treat the problem as single-objective. In the single-objective feature selection, the optimization is performed with regard to one objective only. In wrapper-based methods, this objective is usually the classifier accuracy, while in filter-based, the information shared between the selected feature subset and class labels are maximized. On the other hand, when feature selection is treated as a Multi-Objective (MO) problem, then two or more objectives are taken into consideration. Again, for wrapper-based methods, the additional objective is normally the length of the feature subset, while for filter-based methods, it is the redundancy of information shared between features. Let us notice that many authors today misidentify the concept of MO optimization with a weighted sum of objectives, where they do not optimize according to two or more conflicting objectives, but actually transform the MO problem into single objective problem, and regulate the importance of each objective in the linear sum with size of corresponding weights.

The nature-inspired algorithms always require some thought on defining the appropriate solution representation of the problem being solved. In the case of feature selection in classification tasks, a solution can be represented either as binary-coded or real-coded values. Obviously, a genotype-phenotype mapping is needed for obtaining the solution in the original problem context (i.e., phenotype space) from the corresponding problem-solving space (i.e., genotype space), where it is encoded. When using the binary representation, a feature is present/absent, when the appropriate bit in the solution is set to 1/0. On the other hand, when using a real-coded solution representation, usually some threshold is selected which indicates the inclusion or absence of a feature. Although this mapping is different for both representations, the results represents the feasible solution in their original problem context in both cases.

According to the knowledge of the authors, no paper has ever investigated the impact of the solution representations in nature-inspired algorithms on the quality of results obtained by feature selection. Key novelties of this research paper are:

- As the first, we conducted the study on the importance of solution representations (i.e. binary and real-value) in nature-inspired algorithms for the feature selection problem.
- The study also investigates the impact of solution representations on the feature subsets reduction considering different feature selection methodologies (i.e. wrapper-based via filter-based).

The remainder of the paper is structured as follows. In Section II, the nature-inspired algorithms are presented, then in Section III the two most used feature selection methodologies are described in detail. Section IV is reserved for reporting the results of experimental work, while the paper is concluded in Section V, with possible directions for the future work.

II. NATURE-INSPIRED ALGORITHMS FOR FEATURE SELECTION

Nature-inspired algorithms maintain a population of solutions $\mathbf{x}_i = \{x_{i,j}\}$ for $i = 1, \ldots, Np$ and $j = 1, \ldots, D$. Here, Np denotes the population size (i.e., the number of different solutions), and D is reserved for the dimensionality of the problem to be solved (i.e., the number of elements in the solution). According to the problem for which the algorithm was designed, the solution representation is one the more important factors for the success of finding an optimal solution. For some problems, like numerical function

optimization [23] or constrained engineering optimization problems [26], solution coding and representation is an easy task, while some reflection is needed for problems like association rule mining [27]. In the first case, the best real-coded vector is searched for, while in the second, the solution is encoded into representation, and some kind of mapping the representation to a real-world solution is needed. In feature selection, the problem of solution representation is relatively straightforward, since the number of features in a dataset Z corresponds directly to the dimension of a solution vector $\mathbf{x}_i$. As stated in the last section, there are two possible representations of a solution in the feature selection problem. The former is the already mentioned binary-coded representation, where the algorithm needs to be adapted appropriately. The latter is the real-coded, where the presence of a feature is calculated by mapping the element in the solutions to some predefined interval. For example, if the search space is limited within the interval $[-1, 1]$, then the feature is selected if the value of the element is within the subinterval $[0, 1]$, while, in contrast, the feature is not selected, if the value is within the interval $[-1, 0)$. Obviously, an arbitrary interval may be defined with mapping that is appropriate to this new interval.

Since the motivation of this paper is to provide a comparison between several nature-inspired algorithms for feature selection according to the different representation of individuals, the remainder of this section is devoted to describing briefly the algorithms used in this study. At first, the real-coded variants are described, then the differences to their binary-coded counterparts are presented.

A. PARTICLE SWARM OPTIMIZATION

The Particle Swarm Optimization (PSO) is a computational algorithm belonging to the SI-based family. It was developed by Kennedy and Eberhart in 1995 [14]. This is a population-based algorithm that consists of a population (i.e., swarm) of solutions (i.e., particles). Each particle is associated with a velocity $\mathbf{v}_i^{(t)} = (v_{i,1}, \ldots, v_{i,n})^T$, which is responsible for moving the particles in the search space towards the more promising regions, where a better solution of the problem can be found. During this movement, the algorithm maintains the global best $\mathbf{x}_{best}$ and the local best solutions $\mathbf{p}_i^{(t)}$ for each particle that influence their next move. The moving is formulated mathematically as:

$$\mathbf{v}_i^{(t+1)} = \mathbf{v}_i^{(t)} + c_1 r_1 (\mathbf{p}_i^{(t)} - \mathbf{x}_i^{(t)}) + c_2 r_2 (\mathbf{x}_{best}^{(t)} - \mathbf{x}_i^{(t)}), \quad (1)$$

and

$$\mathbf{x}_i^{(t+1)} = \mathbf{x}_i + \mathbf{v}_i^{(t+1)}, \quad (2)$$

where c_1 and c_2 are acceleration constants used to scale the contribution of the cognitive (third term in Eq. (1)) and social components (second term in Eq. (1)), respectively, and r_1 and r_2 are random values drawn from uniform distribution in the interval $(0, 1)$.

The Eqs. (1) and (2), are used when dealing with real-coded solutions (rPSO), whereas for binary-coded PSO (bPSO), a modification of Eq. (2) is required, as follows:

$$x_{i,j}^{(t+1)} = \begin{cases} 1, & \text{if } rand() \geq S(v_{i,j}^{(t+1)}), \\ 0, & otherwise, \end{cases} \quad (3)$$

where

$$S(v_{i,j}^{(t+1)}) = \frac{1}{1 + e^{-v_{i,j}^{(t+1)}}}. \quad (4)$$

The $S(\cdot)$ represents a sigmoid function for transforming the velocity of the particle to the probability for changing the bit value in the solution vector $\mathbf{x}_i^{(t)}$.

B. GENETIC ALGORITHM

The Genetic Algorithm (GA) was introduced by Holland [13]. It is based on the principle of natural selection and genetics, and belongs to population-based algorithms as well. The original GA was proposed to work in the binary-coded space (bGA), whereas the real-coded variants (rGA) were introduced later. Each solution within the population undergoes acting the several operators.

The first operator is the parent selection, that is responsible for selecting those promising candidate solutions that can pass their own good characteristics on to their offspring, which constitute the so-called mating pool of newly developed individuals. There are several parent selection operators, from which the roulette-wheel and tournament selection are the most frequently used in the GA community [10]. The selected parent solutions then undergo acting the crossover operator, that mainly from two parents generates two offspring. There are many possible crossover operators, yet a one-point crossover was implemented in our study. In the one-point crossover, a crossover point is first drawn randomly. The point divides selected parent solutions into heads and tails. Then, the heads of both parents are copied into corresponding offspring, while the tails from the opposite parents are transferred into the same offspring. In the next step, the offspring solutions in the mating pool undergo acting the mutation operator (normally a bit flip by binary-coded solutions), according to probability p_m. Next, all offspring solutions are evaluated using a fitness function. The last step of this evolution cycle presents survivor selection (also replacement), where the offspring solutions compete with their parents for a place in the new generation. When the generational population model is selected, then the whole population is replaced with the best individuals from both populations (i.e., parent and mating pool). On the other hand, only the best percent of the best offspring replace their parents in the new generation, when the so-called steady-state population model is selected.

In real-coded implementation of the GAs, the crossover operator (e.g., one-point crossover) is replaced with a whole arithmetic crossover that combines two parents $x^{(1)}$ and $x^{(2)}$ to produce two offspring $o^{(1)}$ and $o^{(2)}$ linearly according to the expression, as follows:

$$o_i^{(1)} = \alpha x_i^{(1)} + (1 - \alpha) x_i^{(2)}, \quad and$$

$$o_i^{(2)} = \alpha x_i^{(1)} + (1-\alpha)x_i^{(2)}, \tag{5}$$

where α is used to balance the amount of information which is passed to candidate solutions from their parents. For instance, if $\alpha = 0.5$, two same offspring are generated in the mating pool. In this crossover, all elements of both parents are modified by transfer into their offspring, i.e., no original values are preserved.

Additionally, the mutation operator is also replaced in real-coded GAs, since the bit flip is no longer suitable. Typically, the real-coded mutation replaces the value of a randomly selected element with the value randomly drawn from the uniform distribution in the interval, determined with low and high boundary of the observed element.

C. DIFFERENTIAL EVOLUTION

Differential Evolution (DE) was introduced by Storn and Price in 1995 [15]. DE is a simple, yet effective nature-inspired algorithm for solving various complex problems. It bases on a simple mathematical model, which relies on vector differences. A population of solutions (i.e., vectors) are evolved through generations, where each vector undergoes acting a set of evolutionary operators. These operators are DE mutation, DE crossover, and DE selection. The DE mutation can be expressed mathematically as follows:

$$\mathbf{v}_i^{(t)} = \mathbf{x}_{r_1}^{(t)} + F(\mathbf{x}_{r_2}^{(t)} - \mathbf{x}_{r_3}^{(t)}), \tag{6}$$

where r_1, r_2, and r_3 present random and mutually different integers within the interval $[1, Np]$. Factor F is used for scaling the difference of vector and is usually within the range $[0, 1]$. Then, the newly created mutant vector $\mathbf{v}_i^{(t)}$ enters into the DE crossover operation with the corresponding vector $\mathbf{x}_i$ as:

$$u_{i,j}^{(t)} = \begin{cases} v_{i,j}^{(t)}, & \text{if } rand() \leq Cr or j = j_{rand}, \\ x_{i,j}^{(t)}, & otherwise. \end{cases} \tag{7}$$

The crossover rate Cr, normally defined within the interval $[0, 1]$, controls the probability of modifying an element of the trial vector $\mathbf{u}_i^{(t)}$. Additionally, the j_{rand} index ensures that at least one value from the mutant vector $\mathbf{v}_i^{(t)}$ is modified in the new trial vector. The DE selection is the final operator, which the trial vector must undergo. Thus, each trial vector $\mathbf{u}_i^{(t)}$ competes with its parent $\mathbf{x}_i^{(t)}$ for a place in the new population-based on their fitness value, and the fittest between trial and parent is preserved for surviving into the next generation. This process can be expressed mathematically as follows:

$$x_i^{(t+1)} = \begin{cases} u_i^{(t)}, & \text{if } f(u_i^{(t)}) \leq f(x_i^{(t)}), \\ x_i^{(t)}, & otherwise. \end{cases} \tag{8}$$

However, the described process is valid for the real-coded vector representation (rDE). To make the DE algorithm suitable to be able to work with binary-coded solutions (bDE), the work of [28] was thoroughly analyzed and applied in this study. The difference vector from Eq. 6 ($(\mathbf{x}_{r_2}^{(t)} - \mathbf{x}_{r_3}^{(t)})$) is calculated as the Hamming distance between the two vectors. Since the scaled difference must be interpreted as an integer value, the calculated scaled difference is rounded to its nearest integer. This mutation is denoted as the any-change mutation, which redefines Eq. 6 as:

$$D_h(\mathbf{v}_i^{(t)}, \mathbf{x}_{r_1}^{(t)}) = \begin{cases} (int)d' + 1, & \text{if } rand() \leq d' - (int)d', \\ (int)d', & otherwise. \end{cases} \tag{9}$$

In simple words, Eq. 9 selects one solution vector from among all the vectors at a distance D_h from the vector $\mathbf{x}_{r_1}^{(t)}$.

D. ARTIFICIAL BEE COLONY

The Artificial Bee Colony (ABC) is a SI-based nature-inspired algorithm proposed by Karaboga [16] that mimics the intelligent foraging behavior of a honey bee swarm. In the ABC algorithm, the population (i.e., colony) consists of three types of bees, which correspond to different search strategies within the search space. These bees are: employed, onlooker, and scout. Indeed, the employed and onlooker bees are devoted primarily for exploitation, while the scout bees primarily for exploration. In the first phase of the ABC algorithm, the onlooker and employed bees search for food (i.e., the solution of the problem) in the vicinity of their current location that can be expressed mathematically as:

$$v_i^{(t)} = x_i^{(t)} + \sigma(x_i^{(t)} - x_k^{(t)}), \tag{10}$$

where σ is a random value drawn from uniform distribution in the interval $[-1, 1]$. The onlooker bees then select food sources with regard to the probability p_i associated with the i-th food source:

$$p_i = \frac{f(\mathbf{x}_i)}{\sum_{j=0}^{Np} f(\mathbf{x}_j)}. \tag{11}$$

If the solution is not improved in a predetermined number of iterations, it is reinitialized randomly and, thus, tries to search for the better solutions in the new region of the search space. This phase is the so-called scout bee phase, in which exploration is promoted.

In order to use the ABC algorithm in the binary space (bABC), we implemented the algorithm presented in [29], which is based on the inverse of the Jaccard similarity coefficient. The idea is to rewrite Eq. 10 as:

$$v_i^{(t)} - x_i^{(t)} = \sigma(x_i^{(t)} - x_k^{(t)}), \tag{12}$$

which is then formed as:

$$Dissimilarity(v_i^{(t)}, x_i^{(t)}) \approx \sigma \times Dissimilarity(x_i^{(t)}, x_k^{(t)}), \tag{13}$$

where *Dissimilarity* is defined as: $1 - Similarity(x_i^{(t)}, x_k^{(t)})$. The *Similarity*() function represents the Jaccard coefficient.

In the binary-coded ABC the scaling factor σ is calculated according to Eq. 14.

$$\sigma = \sigma_{max} - (\frac{\sigma_{max} - \sigma_{min}}{MCN})iter \tag{14}$$

σ_{max} and σ_{min} are the upper and lower bounds of σ, MCN is the maximum number of generations (or cycles), while $iter$ is the current generation.

III. NATURE-INSPIRED ALGORITHMS FOR FEATURE SELECTION

A feature selection problem is defined as follows. Let us assume that a large set of features $Z = \{z_1, \ldots, z_m\}$ is defined. Then, the goal is to select a subset of relevant features $F = \{z_{\pi_1}, \ldots, z_{\pi_k}\}$ from the original set of features Z, such that $k < m$ and $F \subset Z$, where k is the number of elements in the feature set and m the number of all features in Z. Thus, it is preferable that the subset of relevant features F should not incur any loss of information compared to the whole set of features Z.

Nature-inspired algorithms are a suitable tool for solving this kind of problems. Three components of these algorithms need to be modified before using for feature selection:

- representation of solution,
- genotype-phenotype mapping,
- evaluation function.

Nature-inspired algorithms operate on the 'generate-and-test' principle, where the algorithm ensures for appropriate modifications of current solutions during the generate phase. In the test phase, the generated solutions undergo the evaluation, where their quality is determined. The evaluation may be performed either internally, like in the case of function optimization, or externally, where the quality of solutions is evaluated by an external process. In our case, the evaluation is performed by wrapper-based and filter-based methods, where the main difference between them is that the former applies the real classification method for evaluation of the generated solution, while the latter actually uses the so-called surrogate model for determining the optimal subset of features.

In the remainder of this section, the mentioned components of the algorithms are described in detail. Because the evaluation function is calculated differently for wrapper-based methods than for filter-based, each mode of calculation is dealt in its own subsection.

A. REPRESENTATION OF SOLUTIONS

The solutions are represented in problem-solving space the same for both encoding (i.e., binary and real-valued), i.e., as real-valued vectors:

$$\mathbf{x}_i^{(t)} = (x_{i,1}^{(t)}, \ldots, x_{i,D}^{(t)})^T, \quad \text{for } i = 1, \ldots, Np, \tag{15}$$

where D denotes a dimension of the problem, and Np is the number of individuals in the population.

B. GENOTYPE-PHENOTYPE MAPPING

The applied genotype-phenotype mapping is illustrated in Figure 1,

from which it can be seen that the binary vector is encoded from the vector in problem-solving space according to rules valid for the particular nature-inspired binary-coded algorithm, while the unchanged real-valued vector is taken by

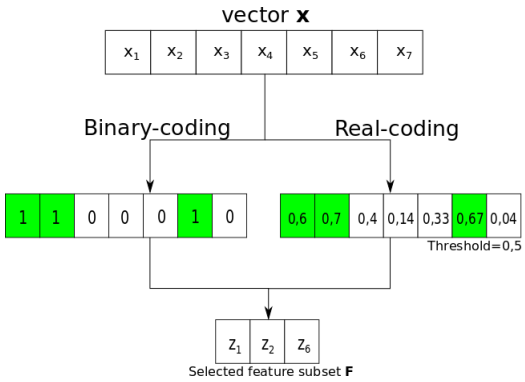


FIGURE 1. The genotype-phenotype mapping used for feature selection, where the vector x in problem-solving space is of dimension $D = 7$.

real-coding. In the former case, the binary value directly determines the presence of the j-th feature in subset F, when decoded value of $x_{i,j}$ is one. In the latter case, features for which hold the relation $x_{i,j} \geq Threshold$, where $Threshold = 0.5$, are taken into subset F.

The mapped subset of features F is then applied to classification of databases either to wrapper-based or filter-based methods, where the quality of the selected features is evaluated with the corresponding evaluation function. In the remainder of the section, both methods are presented in detail.

C. WRAPPER-BASED METHOD

Wrapper-based methods perform feature selection with the particular machine learning algorithm. All candidate feature subsets are evaluated by applying the selected learning algorithm, where the trained model is evaluated on the development set. The results of the classification accuracy correspond to the fitness of the candidate solution. The most common learning algorithm used is the kNN classifier [12]. The wrapper-based methods usually achieve good performance at a great computational cost. However, these methods can be prone to over-fitting, since the selected feature subsets may become too specialized for the development set. Also, there is usually no mechanism to investigate the redundancy within the selected feature subset.

The fitness function for a candidate solution in a wrapper-based method is defined as follows:

$$Fit_1 = Classify(C, F), \tag{16}$$

where C is the classifier, F is the selected feature subset, and $Classify$ stands for the applied cross-validation classification method. The result of a classification method is a confusion matrix, from which the classification accuracy statistical measure is calculated.

The idea of wrapper-based feature selection is depicted in Fig. 2, from which it can be seen that the best subset F^* found during the feature selection is validated on test dataset. The classification accuracy serves as a statistical measure for estimating the quality of the proposed method.

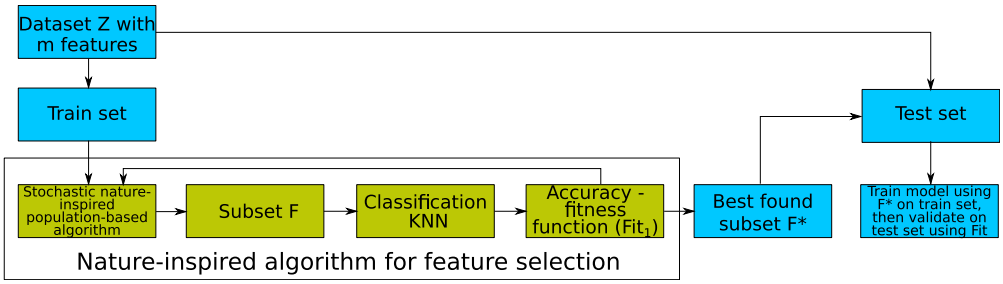


FIGURE 2. Wrapper-based feature selection.

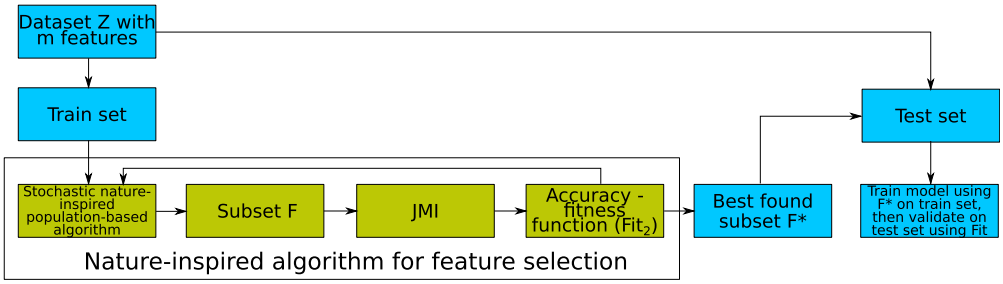


FIGURE 3. Filter-based feature selection.

D. FILTER-BASED METHOD

Filter-based methods use statistical characteristics of the data for determining the subset of features F. Therefore, the feature selection search process is independent of any classification algorithm for evaluating the selected feature subsets. It is also worth emphasizing that filter-based methods are computationally less expensive and more general than wrapper-based approaches, while wrappers are better than filters in terms of the classification quality [30]. One of the most frequently used measures for ranking the selected feature subset is Joint Mutual Information (JMI) [31]. JMI defines the amount of information shared between two random variables, and lies within the range [0, 1]. If these random variables are closely related/unrelated, then the value of JMI will be high/low. For discrete random variables, JMI is defined as:

$$I(A;B) = -\sum_{a\in A}\sum_{b\in B} p(a,b) log_2 \frac{p(a,b)}{p(a)p(b)}, \tag{17}$$

where $p(a, b)$ is the joint probability distribution function of A and B, while $p(a)$ and $p(b)$ are marginal probability distribution functions of A and B, respectively.

This idea can be applied easily to the feature selection problem, where the goal is to select those features whose JMI in relation to the class labels is the highest. Additionally, redundancy within the feature subset can be detected easily by measuring the JMI among all selected features, since a combination of k individually good features may not be the best combination of k [8].

Considering Eq. (17), the fitness function in a filter-based method for a candidate solution is defined as follows:

$$Fit_2 = Rel - Red, \tag{18}$$

where Rel denotes relevance, which is expressed as:

$$Rel = \sum_{z\in F} I(z, c), \tag{19}$$

and Red is redundancy expressed as:

$$Red = \sum_{z_i, z_j \in F} I(z_i, z_j). \tag{20}$$

In Eqs. (19) and (20), F represents the set of selected features, while c denotes the class labels. Fit_2 is a maximization function to maximize the relevance Rel and simultaneously minimise the redundancy $R1$ in the selected feature subset F. (max Eq. 18).

A diagram for filter-based feature selection is presented in Fig. 3, from which it can be seen that a machine learning algorithm is employed to test the performance of the best selected feature subset in the last stage of filter-based feature selection. Also in this case, the result of the algorithm is the classification Accuracy statistical measure calculated from the corresponding confusion matrix.

IV. RESULTS

The purpose of our experimental work was three-fold: (1) To find out how the representation of solutions influences the wrapper-based as well as filter-based feature selection methods, (2) to establish, which of the used representations of solutions reduce the feature set the best, and (3) finally, an analysis of convergence rates of algorithms for wrapper-based and filter-based methods for both solution representations. In line with this, four experiments were conducted, in which the influence of solution representation

within various nature-inspired algorithms is analyzed according to:

- Wrapper-based selection,
- Filter-based feature selection,
- The number of selected features, and
- Algorithms' convergence rates.

During the experimental work, the binary-coded and real-coded algorithm variants described in Section II were applied for solving the feature selection problem. The parameters of the algorithms used in the experiments were set as reported in their respective papers (or experimentally in case of rGA). Indeed, these values are also depicted in Table 1. To provide the comparison as fairly as possible, the termination condition for all algorithms was set to 100 generations, with a total of 30 independent runs per algorithm. The population size $Np = 30$ was kept the same for all algorithms. It is important to emphasize that all algorithm variants used/implemented in this study, have been used in a feature selection problem.

TABLE 1. Parameters of compared algorithms.

Algorithm	Parameter	Value
bDE [28]	Crossover probability	0.08
bABC [29],rABC [16]	Max trial	50
bPSO [32], rPSO [14]	Cognitive and social acceleration	2
bGA [13]	Tournament size	4
	Mutation probability	0.1
rDE [15]	Scaling Factor	0.5
	Crossover probability	0.9
rGA	Tournament size	4
	Mutation probability	0.05

Friedman tests [33] were conducted in order to estimate the results statistically. The Friedman test is a two-way analysis of variances by ranks, where the test statistic is calculated and converted to ranks in the first step. If no significant differences are found according to variances, the post-hoc tests are conducted using the calculated ranks in the second step. The lower the rank, the better the algorithm [34].

The Friedman statistical tests captured the results of classification accuracy according to the selected features on all evaluation datasets in validation phase. Eight classifiers, obtained for all four binary-coded and four real-coded nature-inspired algorithms, consisting of $15 \times 5 = 75$ statistical measures, where the first number in the term denotes the number of observed datasets and the second the five statistical measures, denoting: the best, the worst, the mean, the standard deviation, and median value of classification accuracy obtained over 30 independent runs.

The Friedman test is a safe and robust non-parametric test for comparing more algorithms over multiple classifiers, and, together with the corresponding Nemenyi post-hoc test, enables a good presentation of statistical results [35]. A drawback of the Friedman test is that it makes the whole multiple comparisons over data sets and is, consequently, unable to establish a proper comparison between the considered algorithms [36]. Therefore, a Wilcoxon two paired non-parametric test is applied as a post-hoc test after the control method (i.e., the algorithm with the lowest rank) is determined using the Friedman test. On the other hand, the Nemenyi test is applied as a second post-hoc test in our study. Although this test is very conservative, and may not find any difference in most experiments [37], it was used due to graphical presentation of the results. In contrast, the Wilcoxon test is more powerful [34]. Both tests were conducted using a significance level of 0.05 in this study.

In the remainder of the paper, evaluation databases are described in detail. The section is concluded with a presentation of results obtained in the mentioned experiments.

TABLE 2. Datasets used in the experimental work.

Dataset	# of observations	# of features	# of classes
Adult	48842	14	2
Gas	13910	128	6
Lymphography	148	18	4
Mushroom	8124	22	2
Optic	5620	64	10
Spect	267	22	2
Splice	3190	60	3
Vehicle	946	18	4
Credit-approval	653	15	2
Libra movement	480	90	15
Breast	699	9	2
Sonar	208	60	2
Iosnosphere	351	34	2
Semeion	4456	256	10
German	1996	20	2

A. EVALUATION DATASETS

The performance of all algorithm variants for solving the feature selection was carried out on 15 commonly used classification datasets, acquired from the UCI machine learning repository [38]. The used datasets are collated in Table 2. The features within the mentioned datasets have different characteristics, where they can be either binary, discrete and continuous. Continuous features present the biggest challenge for the algorithms, because JMI can be quite hard to calculate in this case. To simplify this problem, all features were discretized using the Weka software and the MDL method [39]. Each dataset was then split randomly into training and testing sets, with 70% of samples going to the training and 30% to the testing set. We payed closed attention that the classes were evenly distributed among the two sets.

Let us notice that the results are tracked according to test instances. A test instance is considered an average classification accuracy comparison of a binary- and a real-coded algorithm using the selected feature subset of a particular evaluation dataset for a distinct classifier. The classification accuracy is obtained by using the trained classifier

TABLE 3. Classification results for binary and real-value coded algorithms after applying filter-based feature selection, using the kNN classifier.

	ABC			DE			GA			PSO		
	Binary		Real	Binary		Real	Binary		Real	Binary		Real
Adult	59.36±0.0000	<	61.03±0.0511	64.95±0.0745	>	59.92±0.0306	59.82±0.0175	<	60.15±0.0327	59.36±0.0000	<	**67.10±0.0853**
Breast	**92.23±0.0000**	=	**92.23±0.0000**	**92.23±0.0000**	>	92.18±0.0027	**92.23±0.0000**	>	92.14±0.0037	**92.23±0.0000**	>	91.80±0.0068
Credit	**84.26±0.0000**	>	84.16±0.0039	83.94±0.0057	<	84.25±0.0009	**84.26±0.0000**	>	84.23±0.0013	**84.26±0.0000**	>	81.40±0.0918
Gas	59.72±0.0216	<	**80.39±0.0496**	76.78±0.0492	>	57.58±0.0614	56.70±0.0493	<	58.95±0.0244	65.57±0.0680	<	69.12±0.0563
German	70.07±0.0018	>	68.99±0.0209	68.80±0.0242	<	**70.10±0.0000**	69.80±0.0046	>	69.73±0.0049	69.53±0.0141	>	69.41±0.0225
Ionosphere	90.60±0.0017	>	87.83±0.0324	86.07±0.0327	<	**90.72±0.0036**	90.57±0.0000	<	90.69±0.0069	90.66±0.0176	>	87.74±0.0308
Libras	65.72±0.0466	<	71.58±0.0385	**73.33±0.0305**	>	65.39±0.0613	63.00±0.0435	<	66.44±0.0342	71.25±0.0408	>	70.44±0.0386
Sonar	**81.30±0.0390**	>	72.45±0.0775	74.58±0.0555	<	74.74±0.0468	79.11±0.0361	<	79.22±0.0488	76.15±0.0428	>	73.28±0.0779
Lymph	65.96±0.0000	<	68.30±0.0332	**68.65±0.0349**	>	65.96±0.0000	66.24±0.0155	<	67.09±0.0294	66.24±0.0155	<	67.87±0.0423
Mushroom	98.69±0.0002	<	98.77±0.0016	**98.85±0.0017**	>	98.69±0.0002	98.69±0.0000	<	98.70±0.0003	98.70±0.0004	>	97.42±0.0464
Optic	**67.01±0.0307**	>	58.01±0.0462	61.81±0.0388	>	57.34±0.0406	64.47±0.0355	<	65.47±0.0388	58.07±0.0492	>	56.28±0.0603
Semeion	76.64±0.0214	<	**87.65±0.0126**	86.87±0.0147	>	84.23±0.0195	82.49±0.0163	>	78.10±0.0244	85.70±0.0143	>	84.95±0.0162
Spect	**80.08±0.0043**	>	79.05±0.0323	78.31±0.0379	<	79.79±0.0061	79.59±0.0063	<	79.67±0.0063	79.79±0.0061	<	77.45±0.0430
Splice	75.22±0.0413	>	66.85±0.0428	66.38±0.0484	<	73.23±0.0389	73.71±0.0479	<	**76.26±0.0415**	72.57±0.0494	>	66.61±0.0592
Vehicle	**56.64±0.0000**	>	55.12±0.0220	55.49±0.0282	<	56.41±0.0089	**56.64±0.0000**	>	56.17±0.0122	56.52±0.0064	>	54.28±0.0348
Best	8		6	8		7	5		10	12		3

(which is trained using the best selected features F from the training set) on the test set of each experimental dataset. Actually, there is a total of 15 test instances for each algorithm, since 15 datasets were used.

B. INFLUENCE OF SOLUTION REPRESENTATION ON FILTER-BASED FEATURE SELECTION

The first experiment was intended to show how a particular solution representation impacts the filter-based feature selection in a classification problem. Thus, the most commonly used kNN classifier was utilized, while the k was set to 10. The results of filter-based feature selection for binary and real-coded algorithms are gathered in Table 3. The results in the mentioned Tables are averaged after 30 independent runs of each algorithm in the test.

To study the importance of solution representation, both variants of each algorithm are compared, based on the average classification accuracy of the trained models according to the selected feature subsets. To distinguish the performances of binary- and real-coded algorithm variants, appropriate relation operator signs are used to indicate which of the two algorithm variants is 'better than' (sign '>'), 'worse than' (sign '<', or 'are equal' (sign '='). These observations are summarized in Tables for each pair of ⟨algorithm's variant, dataset⟩. Additionally, the corresponding sums are presented in the last column of the table. Also the best average obtained classification accuracy is reported for each dataset, which is marked in bold font.

As can be seen from Table 3, the binary representation was best suited for the PSO, ABC, and DE algorithms, since better results were obtained in 33 test instances, while the real-value variants obtained the better results in just 26 test instances by considering the distinct sum of best rows for all datasets. Interestingly, only the real-coded GA outperformed its binary version, while obtaining better results in 10 test instances. Considering the best overall classification on each distinct dataset (marked in bold face, and not counting ties) the binary ABC and DE are clear winners, since both obtained the overall best results on 3 datasets. Results in Table 3 indicate that binary-coded algorithms perform best when used for filter-based feature selection.

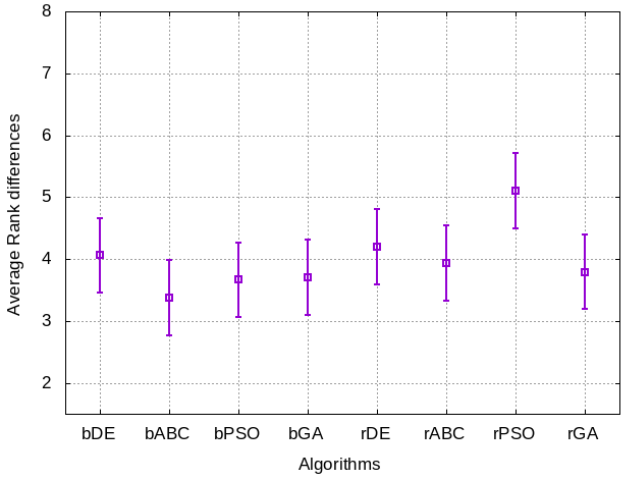
The results of the Friedman non-parametric test is presented in Figure 4, which is divided into a Table (tagged with (a)) presenting the results of the Wilxocon post-hoc test numerically, and a diagram (tagged with (b)) illustrating the results of the Nemenyi post-hoc test graphically. The best algorithm found by the Friedman test is used as a control method with which all the other algorithms are compared using the Wilcoxon two paired non-parametric test. The results of the Wilcoxon test are illustrated by corresponding p-values, where a significant difference between two algorithms appears if $p < 0.05$. The best algorithm in the Nemenyi post-hoc test, as well as the control method in Wilcoxon test, is denoted with the ‡ symbol in the Table, while the significant difference between the control method and corresponding algorithm is depicted by the † symbol.

The results of the Nemenyi post-hoc test are presented as intervals of critical differences. Additionally, these results of the Nemenyi post-hoc tests are presented in diagrams as squares representing average ranks, while the lines are used for delimiting the confidence intervals. The lower the rank value, the better the algorithm. On the other hand, two algorithms are significantly different, if their confidence intervals do not overlap.

The Friedman statistical test confirmed that the binary-coded ABC is the best suited algorithm for filter-based feature selection, since it obtained the lowest rank. It performed significantly better than then real-coded PSO, whereas no statistical differences were found when compared to all others variants. Considering just the real-coded algorithms, the GA obtained the lowest rank by the Friedman statistical test. For a more detailed analysis, the Wilcoxon post-hoc test revealed (see Table 4(a)), that the binary PSO is also significantly better than its real-coded variant when using any classifier. Indeed, confirmed by using the Friedman statistical test and the Wilcoxon test, it seems that filter-based feature selection favors binary-coded algorithms, since, according to obtained results, better feature subsets

Alg	Fri.	Nemenyi		Wilcoxon	
		CD	S.	p-value	S.
bDE	4.06	[3.45, 4.66]		0.64	
bABC	3.38	[2.77, 3.98]	‡	∞	‡
bPSO	3.66	[3.06, 4.27]		0.74	
bGA	3.70	[3.10, 4.31]		0.17	
rDE	4.19	[3.58, 4.79]		$\ll 0.05$	†
rABC	3.94	[3.33, 4.54]		0.56	
rPSO	5.10	[4.49, 5.70]	†	0.12	
rGA	3.79	[3.18, 4.39]		0.21	

(a) Numerical results obtained by kNN.



(b) Graphical results obtained by kNN.

FIGURE 4. Nemenyi and Wilcoxon post-hoc test results for filter-based feature selection with the kNN classifier.

TABLE 4. Results of Wilcox test for all algorithms in filter-based feature selection.

	bDE	bABC	bPSO	bGA	rDE	rABC	rPSO	rGA
bDE	∞	0.64	0.92	0.66	0.49	0.47	†	0.41
bABC	0.64	∞	0.74	0.17	†	0.56	0.12	0.21
bPSO	0.92	0.74	∞	0.99	†	0.55	†	0.91
bGA	0.66	0.17	0.99	∞	0.18	0.55	0.11	0.60
rDE	0.49	†	†	0.18	∞	0.76	0.23	0.17
rABC	0.47	0.56	0.55	0.55	0.76	∞	†	0.29
rPSO	†	0.12	†	0.11	0.23	†	∞	0.07
rGA	0.41	0.21	0.91	0.60	0.17	0.29	0.07	∞

selection was achieved. The mentioned findings are also justified by the results of the Nemenyi post-hoc test.

Table 4 provides a thorough Wilcox test, where each of the comparing algorithms (binary-coded and real-coded) was used as the control method. The analysis of results in Table 4 indicates, that both variants of the PSO algorithm and the real-coded DE show the most significant differences, when compared to other methods.

C. INFLUENCE OF SOLUTION REPRESENTATION ON WRAPPER-BASED FEATURE SELECTION

This experiment was designed to show how a particular solution representation impacts the wrapper-based feature selection in classification. Again, the same classifier as in the last experiment was used to train a model according to the selected features.

The results of the wrapper-based feature selection for binary and real-valued algorithms are gathered in Table 5. Considering the average classification accuracy, the binary-coded ABC and PSO algorithms outperformed the results of their real-coded variants. On the other hand, the opposite effect is visible with the DE and GA algorithms. Considering the best overall classification on each distinct dataset, the binary variants, with the exception of the DE, were able to find the best results on 7 datasets.

The Friedman statistical tests were applied again to find, if there are any significant differences among the algorithms and the used solution representations. The results of these tests are collated in Figure 5.

According to the Friedman statistical test, the best rank was obtained by the real-coded DE algorithm. The binary-coded DE and real-coded ABC were found to be significantly worse than the best algorithm according to the Friedman test.

The Wilcoxon test was considered for the more powerful and accurate analysis of the results in Table 5(a). Indeed, it confirmed the results of the Friedman test and additionally recognized both variants of PSO significantly worse than the best performing algorithm according to Friedman.

A thorough Wilcoxon test was also performed for wrapper-based feature selection. The results are depicted in Table 6.

It seems that wrapper-based feature selection is better suited for binary-coded algorithms, since, according to the results, lower ranks were obtained.

D. INFLUENCE OF SOLUTION REPRESENTATION ON REDUCING THE NUMBER OF FEATURES

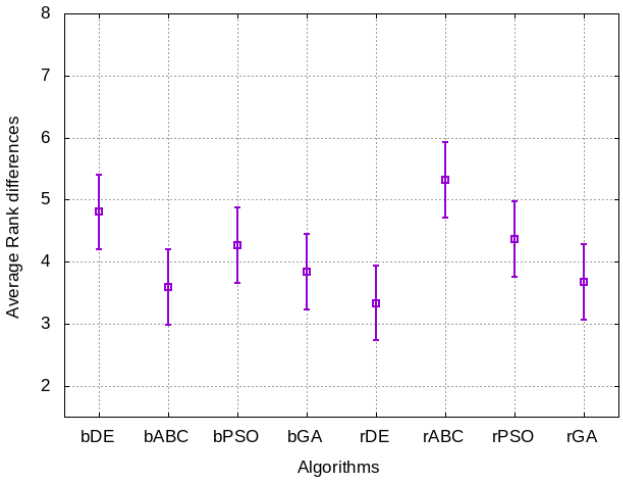
The goal of this experiment was to discover if the solution representation within comparing algorithms has any impact on the size of the selected feature subset. Since the selected

TABLE 5. Classification results for binary and real-value coded algorithms after applying wrapper-based feature selection, using the kNN classifier.

	ABC			DE			GA			PSO		
	Binary		Real	Binary		Real	Binary		Real	Binary		Real
Adult	**81.27±0.0000**	>	81.04±0.0036	81.00±0.0056	<	81.24±0.0012	81.06±0.0042	>	81.03±0.0046	81.10±0.0042	>	81.01±0.0063
Breast	95.36±0.0044	<	95.40±0.0055	95.23±0.0054	<	95.47±0.0047	**95.49±0.0044**	>	95.42±0.0044	95.39±0.0049	<	95.47±0.0045
Credit	82.22±0.0009	<	**82.35±0.0037**	82.34±0.0034	>	82.22±0.0025	80.74±0.0501	<	80.78±0.0503	82.17±0.0029	>	79.53±0.0617
Gas	95.09±0.0015	>	93.32±0.0074	94.23±0.0042	<	95.12±0.0021	**95.14±0.0021**	>	95.03±0.0017	94.87±0.0031	>	94.41±0.0056
German	73.22±0.0124	>	72.59±0.0132	72.83±0.0100	>	72.77±0.0006	72.97±0.0106	<	73.05±0.0161	72.82±0.0116	<	**73.27±0.0150**
Ionosphere	86.82±0.0252	<	87.17±0.0231	87.67±0.0170	>	87.39±0.0160	87.26±0.0247	<	**87.92±0.0212**	**87.92±0.0217**	>	87.36±0.0205
Libras	70.81±0.0296	<	71.72±0.0336	72.17±0.0271	>	71.36±0.0251	70.39±0.0232	<	71.67±0.0239	**72.47±0.0301**	>	71.72±0.0248
Sonar	80.89±0.0456	>	80.16±0.0367	79.95±0.0367	<	**81.56±0.0363**	80.21±0.0381	<	81.04±0.0359	81.04±0.0430	>	80.57±0.0375
Lymph	79.01±0.0675	>	78.51±0.0563	79.65±0.0480	<	**80.71±0.0321**	77.45±0.0611	<	77.59±0.0641	78.58±0.0570	<	78.79±0.0466
Mushroom	**100.00±0.0000**	=	**100.00±0.0000**	**100.00±0.0000**	=	**100.00±0.0000**	**100.00±0.0000**	=	**100.00±0.0000**	**100.00±0.0000**	=	**100.00±0.0000**
Optic	84.75±0.0038	>	81.83±0.0096	82.82±0.0090	<	**84.81±0.0046**	84.45±0.0056	<	84.55±0.0063	84.12±0.0070	>	83.41±0.0080
Semeion	**91.64±0.0066**	>	89.78±0.0076	90.55±0.0075	<	90.62±0.0098	90.97±0.0086	>	90.90±0.0098	90.20±0.0087	<	90.68±0.0066
Spect	82.96±0.0187	>	80.91±0.0376	81.11±0.0336	<	82.26±0.0157	83.42±0.0202	>	82.51±0.0268	**83.46±0.0257**	>	81.11±0.0430
Splice	81.53±0.0129	>	78.11±0.0132	78.85±0.0120	<	81.70±0.0124	82.07±0.0155	<	**82.09±0.0119**	80.16±0.0111	>	79.67±0.0101
Vehicle	**68.09±0.0205**	>	67.72±0.0175	67.07±0.0199	>	66.16±0.0206	67.30±0.0191	<	67.75±0.0221	66.91±0.0207	<	67.37±0.0176
Best	10		4	5		9	5		9	9		5

Alg	Fri.	Nemenyi		Wilcoxon	
		CD	S.	p-value	S.
bDE	4.80	[4.19, 5.40]	†	$\ll 0.05$	†
bABC	3.59	[2.98, 4.19]		0.35	
bPSO	4.26	[3.65, 4.86]		$\ll 0.05$	†
bGA	3.84	[3.23, 4.44]		0.06	
rDE	3.33	[2.72, 3.93]	‡	∞	‡
rABC	5.31	[4.70, 5.91]	†	$\ll 0.05$	†
rPSO	4.36	[3.75, 4.96]		$\ll 0.05$	†
rGA	3.67	[3.06, 4.27]		0.47	

(a) Numerical results obtined by kNN.



(b) Graphical results obtained by kNN.

FIGURE 5. Nemenyi and Wilcoxon post-hoc test results for wrapper-based feature selection with the kNN classifier.

TABLE 6. Results of Wilcox test for all algorithms in wrapper-based feature selection.

	bDE	bABC	bPSO	bGA	rDE	rABC	rPSO	rGA
bDE	∞	†	†	0.06	†	†	0.17	†
bABC	†	∞	0.14	0.28	0.35	†	†	0.79
bPSO	†	0.14	∞	0.32	†	†	0.07	0.56
bGA	0.06	0.28	0.32	∞	0.06	†	0.08	0.33
rDE	†	0.35	†	0.06	∞	†	†	0.47
rABC	†	†	†	†	†	∞	†	†
rPSO	0.17	†	0.07	0.08	†	†	∞	†
rGA	†	0.79	0.56	0.33	0.47	†	†	∞

feature subsets influence the classification results directly, both the feature subset size and the classification accuracy were considered in this experiment. In line with this, both encoding variants of each nature-inspired algorithm were considered for all evaluation datasets. The algorithm variant was promoted as better when the obtained feature size was smaller, while the classification result was the same or better. If the feature subset size was equal, and, at the same time, equal classification accuracy was achieved, the algorithm variants were denoted as equal.

The results of the conducted experiments are depicted in Tables 7 to 8, where the relations between binary- and real-coded nature-inspired algorithms are marked with signs '<', '>', and '=' using the same meaning as in previous subsections. The comparison of algorithm variants being unfit for comparison was marked with sign '!'. The algorithm variant

TABLE 7. Comparison of feature subset reduction for binary and real-coded algorithms for filter-based feature selection.

	ABC			DE			GA			PSO		
	Binary		Real	Binary		Real	Binary		Real	Binary		Real
Adult	5.00±0.00	<	5.00±0.00	5.17±0.38	!	5.00±0.00	5.00±0.00	<	5.00±0.00	5.00±0.00	!	5.10±0.40
Breast	1.00±0.00	=	1.00±0.00	1.00±0.00	>	1.00±0.00	1.00±0.00	>	1.00±0.00	1.00±0.00	>	1.00±0.00
Credit	3.33±0.48	>	3.57±0.57	3.70±0.70	<	3.50±0.57	3.50±0.51	>	3.60±0.50	3.37±0.49	>	3.97±0.72
Gas	24.20±2.89	!	44.10±6.05	44.63±3.23	!	32.33±3.58	29.27±3.00	<	26.13±3.07	35.50±2.70	!	37.30±4.48
German	4.97±0.18	!	4.57±0.57	4.33±0.80	!	5.00±0.00	4.70±0.47	!	4.63±0.49	4.70±0.47	!	4.43±0.77
Ionosphere	5.30±0.47	>	5.73±0.98	6.33±1.09	<	5.57±0.57	5.50±0.51	!	5.77±0.43	6.03±0.67	!	5.53±1.01
Libras	6.83±0.38	!	21.10±5.61	23.60±1.94	!	9.07±1.68	6.90±0.84	<	6.77±0.43	15.07±1.51	>	15.80±2.98
Sonar	7.10±0.31	>	10.13±1.57	13.70±2.20	<	7.10±1.16	7.27±0.58	<	7.20±0.41	8.83±1.64	>	9.40±1.69
Lymph	4.00±0.00	<	3.37±0.72	3.27±0.87	>	4.00±0.00	3.97±0.18	<	3.87±0.35	3.97±0.18	<	3.17±0.59
Mushroom	2.27±0.45	!	2.90±0.71	3.20±1.00	!	2.67±0.48	2.43±0.50	!	2.50±0.57	2.60±0.67	>	3.17±1.02
Optic	6.67±0.92	>	11.57±2.28	15.40±2.33	!	9.13±1.87	6.73±1.28	<	6.23±0.90	10.27±2.21	>	11.47±2.32
Semeion	26.80±1.19	!	103.10±3.19	93.13±3.54	!	65.90±3.67	51.10±3.19	!	37.27±3.60	76.67±3.58	!	75.30±4.28
Spect	7.00±0.00	!	6.37±0.56	6.20±0.55	!	7.00±0.00	6.57±0.50	!	6.87±0.35	6.77±0.43	!	6.27±0.52
Splice	3.37±0.49	>	9.63±2.20	13.00±1.91	<	4.13±1.01	3.23±0.50	!	3.47±0.51	7.30±1.21	>	7.90±1.77
Vehicle	5.00±0.00	!	4.73±0.52	4.17±0.70	!	5.00±0.00	5.00±0.00	>	5.00±0.00	5.00±0.00	!	4.60±0.50

TABLE 8. Comparison of feature subset reduction for binary and real-coded algorithms for wrapper-based feature selection.

	ABC			DE			GA			PSO		
	Binary		Real	Binary		Real	Binary		Real	Binary		Real
Adult	8.00±0.00	!	7.80±1.06	7.67±1.47	!	7.87±0.51	7.70±1.44	>	8.73±2.53	7.67±1.18	>	8.07±1.87
Breast	6.37±1.00	!	6.37±1.03	6.00±0.98	!	6.50±0.97	6.53±0.90	>	6.77±1.01	6.60±1.04	<	6.53±0.82
Credit	7.23±0.73	<	6.33±1.32	6.27±1.39	>	6.70±1.47	6.40±1.40	!	6.57±1.30	6.50±1.81	!	5.90±1.60
Gas	100.50±5.10	!	71.67±4.51	72.53±5.72	!	73.63±5.23	73.03±5.65	>	74.10±4.51	76.93±4.74	!	74.60±3.92
German	12.43±1.79	!	11.23±1.65	11.87±1.98	!	11.03±1.07	11.70±2.07	<	11.47±2.21	11.20±2.06	!	12.17±1.86
Ionosphere	11.33±2.87	!	12.67±2.26	11.93±2.32	!	11.10±2.54	10.80±2.71	!	11.30±2.44	10.80±2.55	>	13.60±3.09
Libras	42.80±4.76	<	42.73±4.20	41.43±4.58	!	40.93±5.20	40.17±3.82	!	40.50±5.30	40.40±5.51	>	43.43±4.37
Sonar	28.37±4.01	>	29.60±3.67	27.67±3.56	<	26.43±3.22	26.20±3.99	!	26.77±2.90	26.97±3.74	>	29.17±3.82
Lymph	5.87±1.41	>	6.07±1.17	7.47±1.31	<	6.00±0.74	6.57±1.81	!	6.97±1.99	6.73±2.00	!	7.67±2.11
Mushroom	7.90±1.75	>	11.97±2.01	11.53±2.05	<	11.23±2.08	11.53±2.16	>	12.43±2.62	11.60±2.40	<	11.17±1.84
Optic	55.87±2.06	!	43.07±2.96	43.43±3.39	!	48.70±2.79	48.23±3.06	!	48.63±3.00	46.60±2.34	!	45.43±3.23
Semeion	200.80±7.82	!	132.00±8.49	137.10±7.84	!	140.20±7.40	138.70±8.33	>	140.10±6.18	136.10±7.47	!	138.40±8.56
Spect	11.30±1.62	!	11.07±2.23	11.80±1.85	<	11.27±1.31	11.20±1.90	>	11.57±1.43	11.00±1.88	!	10.90±2.31
Splice	21.83±3.51	>	30.07±3.67	29.17±3.87	<	20.60±2.28	20.07±4.26	<	18.77±3.20	23.80±3.10	>	27.17±3.83
Vehicle	10.83±1.29	!	10.77±1.07	10.37±1.40	!	9.60±0.89	10.40±1.54	!	10.47±1.28	9.97±1.43	!	10.93±1.51

is unfit for comparison if the better feature subset size reduction and best average classification accuracy is not achieved with the same algorithm variant. Thus, the obtained average feature subset length is reported for binary- and real-coded algorithms by feature selection on each observed dataset.

The results in Table 7 show that binary-coded algorithms perform better in filter-based feature selection. In several instances, they achieved the better results compared with real-coded algorithms in reducing the number of features, while maintaining the same or better classification accuracy. The same is true for the wrapper-based methods presented in Table 8, where the binary-coded algorithms achieved a similar performance. The only difference is in the higher average number of selected features in the case of wrapper-based methods. The reason for that lies in the calculation of redundancy among the selected features in filter-based feature selection methodologies that is not present in wrapper-based methods.

E. INFLUENCE OF FEATURE SELECTION METHODOLOGY ON BINARY- AND REAL-CODED ALGORITHMS

The goal of this experiment was to check how do both the feature selection methodologies influence the results of binary- and real-coded algorithms. The comparison is provided in Figure 6 in terms of Friedman ranks and critical differences. The left four methods in each graph belong to filter feature selection methodology, while the right are wrapper-based. The results show that wrapper-based feature selection provides better classification accuracy in real- and binary-coded algorithms by significant margins.

F. ANALYSIS OF CONVERGENCES RATES OF COMPARING ALGORITHMS

The last test is reserved for comparing the convergence rates of binary- and real-coded algorithms for filter- and wrapper-based feature selection. The convergence is analyzed in terms of objective function and also the number of features. The reported results are averages of all runs, and for all datasets for each algorithm. The results of this test is depicted in Figures 7 and 8.

Considering the filter-based feature selection, almost all algorithms, with the exception of binary-coded DE and real-coded ABC converge similarly in terms of the objective function. In terms of lowering the number of features, the binary-coded ABC algorithm is the most successful, where the difference with second best algorithm (real-coded GA) is enormous. We find the reason in good exploratory capabilities of the binary-coded ABC.

The analysis of convergence graphs for wrapper-based feature selection shows similar performances as in filter-based methodology for the objective function. Considering the number of features, the number of features grows through the

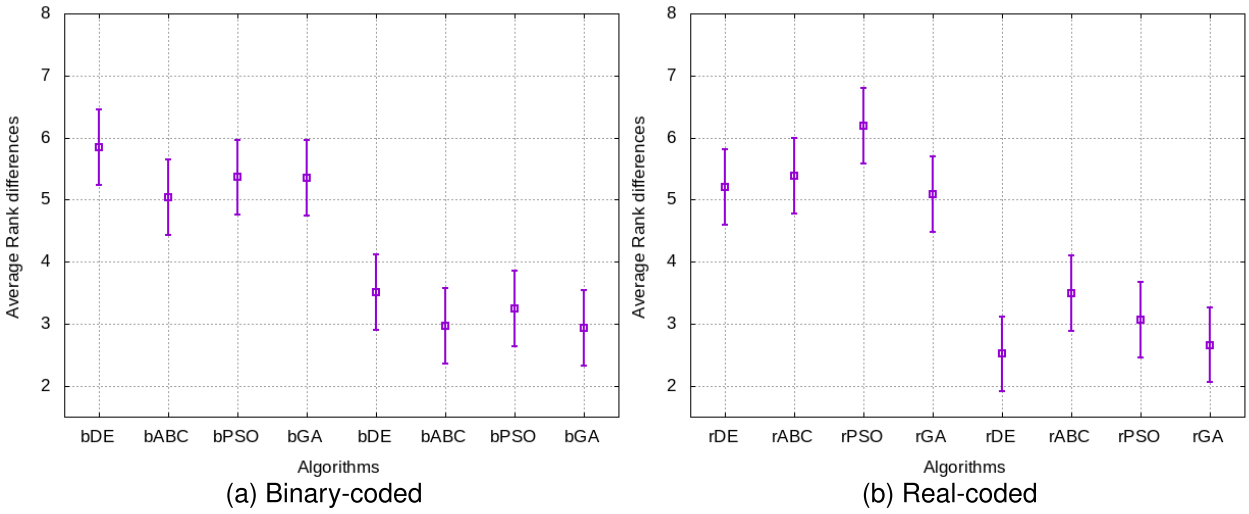


FIGURE 6. Comparison of filter and wrapper feature selection methodologies for binary- and real-coded algorithms.

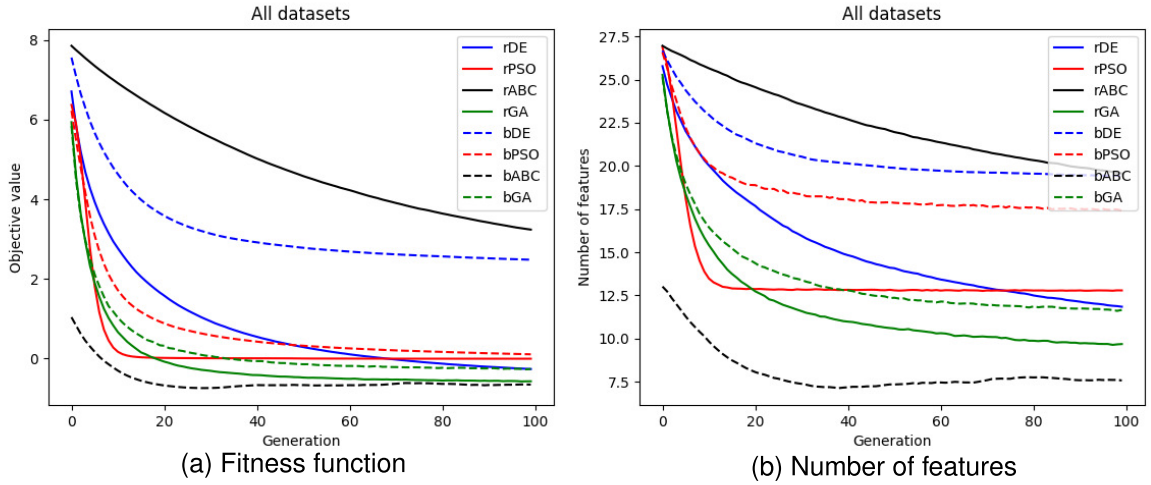


FIGURE 7. Converge rates for filter-based feature selection. Subfigure (a) reports the objective function convergence rates, while subfigure (b) is reserved for reducing the number of features through generations of the evolutionary process.

evolutionary process. This is normal, since the objective function in wrapper-based feature selection methodology only takes into account the classification accuracy of the current subset.

Results of this experiment show that filter-based feature selection methodology is more successful in reducing the feature subset size, although achieving a worse classification result. (see Tables 7 and 8).

G. DISCUSSION

After a careful analysis of results in Tables 3 and 5, we can conclude that the bABC outperformed the results of the rABC using both classification methods, since better classification results were achieved in more test instances. Specifically, the filter-based bABC obtained the best result in 8 test instances, while the wrapper-based bABC in 10. This can be contributed to the neighborhood search mechanism, which greatly improves the global search ability in the feature selection problem space. On the other hand, the rABC most likely use more function evaluations to find the optimal subset of features. Considering the DE algorithm, both variants achieved similar results. Since the 'DE/rand/1/bin' mutation strategy was utilized for the rDE variant, good exploration was achieved, which can compete with the advanced any-change mutation used in the bDE. The filter-based rGA outperformed the results of the filter-based bGA, since it achieved best results in 10 test instances (i.e., 5 more than the bGA). For the wrapper-based variant, it achieved the best results in 9 test instances, compared to just 5 for the wrapper-based bGA. We speculate that this fact can be mainly ascribed to a better crossover mechanism incorporated within this variant of the rGA. Consequently, the better and faster the exploration, the better the feature subsets. A similar phenomenon can be observed, when the PSO algorithm

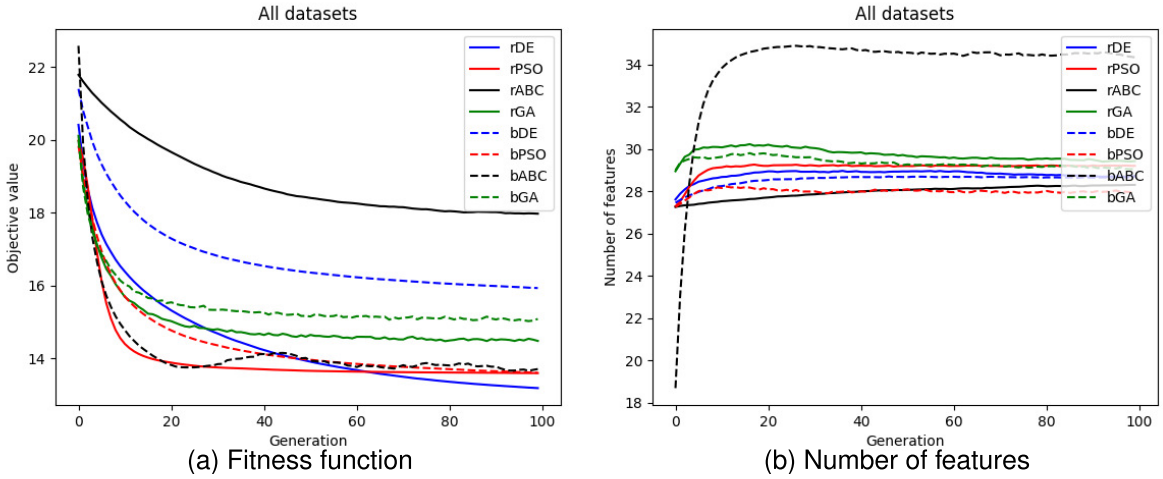


FIGURE 8. Converge rates for wrapper-based feature selection. Subfigure (a) reports the objective function convergence rates, while subfigure (b) is reserved for reducing the number of features through generations of the evolutionary process.

results were taken into consideration, but in the favor of the binary-coded variant. According to the results, the bPSO was able to find better feature subsets in filter-based and wrapper-based methods. In general, this complies with the well known fact that the binary-coded PSO variants achieve good results in solving discrete problems [2].

Considering the reduction of the number of features reported in Tables 7 and 8, the bABC outperformed the results of the rABC regardless of which feature selection method was used. Again, this can be ascribed to the superior neighborhood search when creating the mutant vector. Interestingly, the filter-based approach produced less incomparable test instances (marked with '!' in the corresponding tables), meaning that better feature subsets are obtained, since feature relations are included in the fitness function. The DE algorithm was able to produce smaller feature subsets when using the real-coded solution representation by using both classification methods. This is due to the rDE being more focused on the smooth and slow refinement of the feature subsets, whereas the bDE variant is focused more on exploration in all stages of the evolutionary process. The rGA algorithm showed better results in reducing the number of features with the filter-based method. On the other hand, the bGA found smaller feature subset using the wrapper-based approach.

The binary-coded PSO achieved the best results by using both feature selection methods. It seems that the sigmoid function, which is used for mapping the particle velocity to a solution, has the strong impact on the creation of mutant particles during the evolutionary process. This mapping offers better and faster exploration of the search space (i.e. all possible feature subsets).

Regarding the convergence rates in filter-based feature selection, all algorithms show similar performance, except the rABC and bDE algorithms. The most probable cause for this seems to be a fast convergence towards a local optimum on some of the datasets. The convergence rates of algorithms using the wrapper-based method is very similar to those using the filter-based, except the wrapper-based rABC, bDE, rGA, and bGA that converge much slower than all other algorithms in tests.

V. CONCLUSION

This paper investigates the importance of solution representations in nature-inspired algorithms when applied for feature selection in classification problems. Today, these problems have emerged in many application domains, referring especially to expert and intelligent systems. Simultaneously, more and more developers have begun to solve the problems using various methods. Therefore, the purpose of the paper is to help these upcoming developers with the findings of the study, i.e., primarily, how to select the appropriate solution representation, which algorithm to select, and which classification method is the more suitable for their needs.

According to a recent survey [12], the four most often used nature-inspired algorithms for feature selection have been exposed, like DE, ABC, PSO, and GA, which are also considered in our study. Binary-coded and real-coded algorithm variants were compared using the filter-based and wrapper-based feature selection methods on nine regularly used datasets from the UCI machine learning repository. An additional experiment was performed, where selected feature subset sizes were analyzed according to different solution representations. Statistical tests were performed to check for any significant differences between the results of various nature-inspired algorithms using binary-coded and real-coded representation.

The results showed that binary-coded solution representation is preferred to filter-based and wrapper-based methods. Interestingly, the binary-coded ABC achieved the best results in filter-based feature selection based on the classification accuracy. On the other hand, the binary variants of ABC and PSO provided the best results overall for the wrapper-based

feature selection. Regarding the feature subset reduction, the binary-coded variants of the observed nature-inspired algorithms lowered the number of selected features on average in both feature selection approaches. Filter-based methods had even lower average sizes of the selected feature subsets compared to those obtained with the wrapper-based method.

For the future work we would like to investigate the impact of solution representations in different applications areas of feature selection, while also extending the study to a multi-objective feature selection problem formulation.

REFERENCES

[1] M. Bennasar, Y. Hicks, and R. Setchi, "Feature selection using joint mutual information maximisation," *Expert Syst. Appl.*, vol. 42, no. 22, pp. 8520–8532, Dec. 2015.

[2] Q. Al-Tashi, S. J. A. Kadir, H. M. Rais, S. Mirjalili, and H. Alhussian, "Binary optimization using hybrid grey wolf optimization for feature selection," *IEEE Access*, vol. 7, pp. 39496–39508, 2019.

[3] M. A. T. Mohammed, W. M. W. Mohd, R. A. Arshah, M. Mungad, E. Sutoyo, and H. Chiroma, "Hybrid filter for attributes reduction in soft set," in *Proc. Int. Conf. Data Eng. (DaEng)*. Singapore: Springer, 2019, pp. 245–256.

[4] A. Zarshenas and K. Suzuki, "Binary coordinate ascent: An efficient optimization technique for feature subset selection for machine learning," *Knowl.-Based Syst.*, vol. 110, pp. 191–201, Oct. 2016.

[5] P. Zhou, X. Hu, P. Li, and X. Wu, "Online feature selection for high-dimensional class-imbalanced data," *Knowl.-Based Syst.*, vol. 136, pp. 187–199, Nov. 2017.

[6] U. Mlakar, I. Fister, J. Brest, and B. Potočnik, "Multi-objective differential evolution for feature selection in facial expression recognition systems," *Expert Syst. Appl.*, vol. 89, pp. 129–137, Dec. 2017.

[7] Y. Saeys, I. Inza, and P. Larranaga, "A review of feature selection techniques in bioinformatics," *Bioinformatics*, vol. 23, no. 19, pp. 2507–2517, Oct. 2007.

[8] L. Cervante, B. Xue, M. Zhang, and L. Shang, "Binary particle swarm optimisation for feature selection: A filter based approach," in *Proc. IEEE Congr. Evol. Comput.*, Jun. 2012, pp. 1–8.

[9] C. Pascoal, M. R. Oliveira, A. Pacheco, and R. Valadas, "Theoretical evaluation of feature selection methods based on mutual information," *Neurocomputing*, vol. 226, pp. 168–181, Feb. 2017.

[10] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing*. Berlin, Germany: Springer-Verlag, 2003.

[11] C. Blum and D. Merkle, *Swarm Intelligence: Introduction and Applications*. Berlin, Germany: Springer-Verlag, 2008.

[12] B. Xue, M. Zhang, W. N. Browne, and X. Yao, "A survey on evolutionary computation approaches to feature selection," *IEEE Trans. Evol. Comput.*, vol. 20, no. 4, pp. 606–626, Aug. 2016.

[13] J. H. Holland, "Genetic algorithms," *Sci. Amer.*, vol. 267, no. 1, pp. 66–72, Jul. 1992.

[14] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. 6th Int. Symp. Micro Mach. Hum. Sci. (MHS)*, 1995, pp. 39–43.

[15] R. Storn and K. Price, "Differential evolution–A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, 1997.

[16] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Dept. Eng. Fac., Comput. Eng., Erciyes Univ., Kayseri, Turkey, Tech. Rep.-tr06, 2005.

[17] A. Ghosh, A. Datta, and S. Ghosh, "Self-adaptive differential evolution for feature selection in hyperspectral image data," *Appl. Soft Comput.*, vol. 13, no. 4, pp. 1969–1977, Apr. 2013.

[18] J. Derrac, S. García, and F. Herrera, "A first study on the use of coevolutionary algorithms for instance and feature selection," in *Proc. Int. Conf. Hybrid Artif. Intell. Syst.* Berlin, Germany: Springer, 2009, pp. 557–564.

[19] Y. Zhang, H.-G. Li, Q. Wang, and C. Peng, "A filter-based bare-bone particle swarm optimization algorithm for unsupervised feature selection," *Int. J. Speech Technol.*, vol. 49, no. 8, pp. 2889–2898, Aug. 2019.

[20] L. Brezočnik, I. Fister, and V. Podgorelec, "Swarm intelligence algorithms for feature selection: A review," *Appl. Sci.*, vol. 8, no. 9, p. 1521, Sep. 2018.

[21] S. Kar, K. Das Sharma, and M. Maitra, "Gene selection from microarray gene expression data for classification of cancer subgroups employing PSO and adaptive K-nearest neighborhood technique," *Expert Syst. Appl.*, vol. 42, no. 1, pp. 612–627, Jan. 2015.

[22] J. Del Ser, E. Osaba, D. Molina, X.-S. Yang, S. Salcedo-Sanz, D. Camacho, S. Das, P. N. Suganthan, C. A. Coello Coello, and F. Herrera, "Bio-inspired computation: Where we stand and what's next," *Swarm Evol. Comput.*, vol. 48, pp. 220–250, Aug. 2019.

[23] J. Brest, M. S. Maučec, and B. Bošković, "Single objective real-parameter optimization: Algorithm jSO," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jun. 2017, pp. 1311–1318.

[24] R. Tanabe and A. S. Fukunaga, "Improving the search performance of SHADE using linear population size reduction," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Beijing, China, Jul. 2014, pp. 1658–1665.

[25] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, Apr. 1997.

[26] U. Mlakar, "Hybrid cuckoo search for constraint engineering design optimization problems," in *Proc. StuCoSReC*, Oct. 2016, pp. 57–60.

[27] U. Mlakar, M. Zorman, I. Fister, and I. Fister, "Modified binary cuckoo search for association rule mining," *J. Intell. Fuzzy Syst.*, vol. 32, no. 6, pp. 4319–4330, May 2017.

[28] J. Apolloni, G. Leguizamón, and E. Alba, "Two hybrid wrapper-filter feature selection algorithms applied to high-dimensional microarray experiments," *Appl. Soft Comput.*, vol. 38, pp. 922–932, Jan. 2016.

[29] E. Hancer, B. Xue, D. Karaboga, and M. Zhang, "A binary ABC algorithm based on advanced similarity scheme for feature selection," *Appl. Soft Comput.*, vol. 36, pp. 334–348, Nov. 2015.

[30] B. Xue, "Particle Swarm Optimisation for Feature Selection," Ph.D. dissertation, School Eng. Comput. Sci., Victoria Univ. Wellington, Wellington, New Zealand, 2014.

[31] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. Hoboken, NJ, USA: Wiley, 2012.

[32] Y. Zhang, D. Gong, Y. Hu, and W. Zhang, "Feature selection algorithm based on bare bones particle swarm optimization," *Neurocomputing*, vol. 148, pp. 150–157, Jan. 2015.

[33] M. Friedman, "A comparison of alternative tests of significance for the problem of *m* rankings," *Ann. Math. Statist.*, vol. 11, no. 1, pp. 86–92, Mar. 1940.

[34] U. Mlakar, I. Fister, and I. Fister, "Hybrid self-adaptive cuckoo search for global optimization," *Swarm Evol. Comput.*, vol. 29, pp. 47–72, Aug. 2016.

[35] P. Nemenyi, "Distribution-free multiple comparisons (doctoral dissertation, princeton university, 1963)," *Diss. Abstr. Int.*, vol. 25, no. 2, p. 1233, 1963.

[36] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 3–18, Mar. 2011.

[37] S. Garcia and F. Herrera, "An extension on 'statistical comparisons of classifiers over multiple data sets' for all pairwise comparisons," *J. Mach. Learn. Res.*, vol. 9, pp. 2677–2694, Dec. 2008.

[38] M. Lichman, "UCI machine learning repository," School Inf. Comput. Sci., Univ. California Irvine, Irvine, CA, USA, Tech. Rep., 2013.

[39] U. Fayyad and K. Irani, "Multi-interval discretization of continuous-valued attributes for classification learning," in *Proc. IJCAI*, 1993, pp. 1022–1029.

UROŠ MLAKAR (Member, IEEE) received the B.Sc., M.Sc., and Ph.D. degrees in computer science from the University of Maribor, in 2010, 2014, and 2019, respectively. His area of interests include evolutionary computation and image processing. He has participated in the development of 28 scientific articles, and is also a reviewer in several international journals.

IZTOK FISTER, JR. (Member, IEEE) received the B.Sc., M.Sc., and Ph.D. degrees in computer science from the University of Maribor, Slovenia. He is currently Assistant Professor with the University of Maribor, Slovenia. He has published more than 120 research articles in refereed journals, conferences, and book chapters. Furthermore, he is a member of the editorial boards of five different international journals, and he has acted as program committee in more than 30 international conferences. His areas of interests include data mining, pervasive computing, optimization, and sport science.

IZTOK FISTER (Member, IEEE) received the degree in computer science from the University of Ljubljana, in 1983, and the Ph.D. degree from the Faculty of Electrical Engineering and Computer Science, University of Maribor, in 2007. Since 2010, he was a Teaching Assistant with the Computer Architecture and Languages Laboratory, Faculty of Electrical Engineering and Computer Science, University of Maribor. His research interests include computer architectures, programming languages, operational researches, artificial intelligence, and evolutionary algorithms. In his research areas, he has published 35 original scientific articles, eight review articles, 11 book chapters, edited one book, and contributed to more than 50 international scientific conferences. He is also one of the presidents in the international student conference in computer science. In many journals with impact factor, he has been promoted as the outstanding reviewer.