Int. J. Appl. Math. Comput. Sci., 2014, Vol. 24, No. 1, 111–122
DOI: 10.2478/amcs-2014-0009

amcs

A DIFFERENTIAL EVOLUTION APPROACH TO DIMENSIONALITY REDUCTION FOR CLASSIFICATION NEEDS

GORAN MARTINOVIĆ, DRAŽEN BAJER, BRUNO ZORIĆ

Faculty of Electrical Engineering
J.J. Strossmayer University of Osijek, Kneza Trpimira 2b, 31000 Osijek, Croatia
e-mail: goran.martinovic@etfos.hr

The feature selection problem often occurs in pattern recognition and, more specifically, classification. Although these patterns could contain a large number of features, some of them could prove to be irrelevant, redundant or even detrimental to classification accuracy. Thus, it is important to remove these kinds of features, which in turn leads to problem dimensionality reduction and could eventually improve the classification accuracy. In this paper an approach to dimensionality reduction based on differential evolution which represents a wrapper and explores the solution space is presented. The solutions, subsets of the whole feature set, are evaluated using the k-nearest neighbour algorithm. High quality solutions found during execution of the differential evolution fill the archive. A final solution is obtained by conducting k-fold cross-validation on the archive solutions and selecting the best one. Experimental analysis is conducted on several standard test sets. The classification accuracy of the k-nearest neighbour algorithm using the full feature set and the accuracy of the same algorithm using only the subset provided by the proposed approach and some other optimization algorithms which were used as wrappers are compared. The analysis shows that the proposed approach successfully determines good feature subsets which may increase the classification accuracy.

Keywords: classification, differential evolution, feature subset selection, k-nearest neighbour algorithm, wrapper method.

1. Introduction

The problem of selecting the features to use during the classification of patterns is very common. When a problem originates from the real world, it is often very hard to tell the difference between the necessary and differentiating features from the ones that are not. At a first glance, it may seem that a higher number of features guarantees a better classification accuracy, but that is often not the case. Through the use of all available features, some redundant, unnecessary or even detrimental ones could be included, which would eventually cause poor classification results and reduce the classification accuracy. It is not always possible to know up-front which features should be included in the classification process and which should be left out. For the stated reason, various methods have been developed which evaluate the features. These approaches can be divided into Ranked Selection (RS) methods (e.g., Wu *et al.*, 2009; Balakrishnan *et al.*, 2008), and into Feature Subset Selection (FSS) methods that use either filters (Ferreira and Figueiredo, 2012) or wrappers, where search methods such as Differential Evolution (DE) (Khushaba *et al.*, 2008), Genetic Algorithms (GAs) (Yusof *et al.*, 2012), ant colony optimization (Kubir *et al.*, 2012) and particle swarm optimization (Chuang *et al.*, 2011) are applied.

This paper proposes a wrapper based FSS approach. The classifier used is the k-Nearest Neighbour (k-NN) algorithm and the DE algorithm adapted for binary space operation is used as a wrapper. The proposed approach includes an archive which stores a selected number of solutions obtained through the process of optimization, i.e., good quality solutions found during the DE run. A final solution is then obtained from the archive by performing a k-fold cross-validation of its solutions and selecting the best one. This process is expected to yield a more refined subset selection. The archive alongside with the k-fold cross-validation attempts to ensure the selection of the most salient features from the set.

The remainder of the paper is organized as follows. In Section 2 the classification problem is defined, the method of feature selection is described and a brief overview of related work is given. It also describes the k-NN algorithm, which is used as a classifier due to its

simplicity and good performance. A brief description of DE, as well as a couple of approaches to its adaptation to binary optimization problems, is given in Section 3. The proposed approach, which combines DE, adapted for operation in the binary space, and the k-NN algorithm, is described in Section 4. Section 5 presents the experiment set-up and the results of the experimental analysis conducted on several standard datasets.

2. Classification and feature selection

Classification belongs to the area of pattern recognition and it is its typical representative. The classification problem can be defined as follows. If a pattern set $\mathcal{S} = \{(\mathbf{X}_1, L_1), \ldots, (\mathbf{X}_m, L_m)\}$ is given, where $m = |S|$, $\mathbf{X}_i$ is the sample of the i-th set element, and L_i is a designation, i.e., the class of the i-th set element, then it is necessary to determine the class for an incoming sample of an unknown class based on its features and based on a known set $\mathcal{S}$ (Duda *et al.*, 2001). Usually the samples are represented with feature vectors containing n elements, as in

$$\mathbf{X} = [x_1, \ldots, x_n]^T. \tag{1}$$

Methods used for classification vary from Bayes classifiers and k-NN algorithms (Jain *et al.*, 2000), Artificial Neural Networks (ANNs) (Debska and Guzowska-Swider, 2011; Gocławski *et al.*, 2012) and Support Vector Machines (SVMs) (Hsu and Lin, 2002; Jeleń *et al.*, 2008) to classifier ensembles (Woźniak and Krawczyk, 2012),

The classification accuracy depends greatly on the method used, but also on the underlying problem, i.e., the characteristics of the data on which the classification method is applied. Since each feature vector is represented by n values, $x_i \in \mathbf{X}$ for $i = 1, \ldots, n$, where every value represents one of the features of the input data, we can observe feature vectors as points in an n-dimensional space. Features can be of different types, which could, for example, be integral (picture width in pixels), real (width in centimetres in measurements) or categorical ("Africa" or "Asia" for a continent, "0" or "AB" for blood type). As the number of features increases, the dimensionality of the problem proportionally expands and with it the calculation complexity of the relation between feature vectors.

An increased number of features does not always guarantee a better classification quality nor does it improve differentiation between the classes. Additional features are often introduced in order to better describe the problem, and one of the consequences could be the introduction of irrelevant, redundant and even features that are detrimental to classification accuracy (Dash and Liu, 1997). To avoid this problem, an attempt is made to reduce the number of features used in classification and this process is called feature selection (Javed *et al.*, 2012).

The selection is performed through four formal steps: generating a subset, subset evaluation, the break criterion and result evaluation (Dash and Liu, 1997), and two ways of conducting this process are commonly used. The first one, RS, is the process of selection where weights are given to all of the features. After that, they are ranked and a predetermined number of the most highly ranked features is used. The second one, FSS, is the process of selection where various methods are used to determine the best subset of the full feature set. In this paper, an FSS approach is used, and the goal is to determine one subset of the feature set that maximizes the result of the fitness function. An attempt is made to find a binary vector o^* which is used as a mask on the full feature vector to yield a reduced feature set that will be used for the classification. The yielded reduced feature set must achieve the highest possible classification accuracy as described by an objective function f, which is formally shown in

$$o^* \in O = \arg \max_{\mathbf{b} \in \{0,1\}^n} f(\mathcal{S}, \mathcal{I}, \mathbf{b}) \,. \tag{2}$$

Here O represents the set of binary vectors that yield a reduced feature vector for which the objective function $f(S, I, b)$ reaches a global maximum, where I is the set of input patterns to be classified. It is clear that this problem can also be treated as a minimization problem if the classification error is observed. Additionally, it is important to mention that FSS methods are commonly divided into filter and wrapper ones. Filter methods are somewhat faster and represent a form of preprocessing that incorporates subset or individual feature evaluation without the inclusion of a classifier. Ferreira and Figueiredo (2012) gave an example of filter usage where they showed filter effectiveness on feature vector sets of up to 10^5 features. They used two different filters. The first one is based on the idea that the importance of a particular feature is proportional to its dispersion. The second one uses a redundancy measure between the features.

In wrapper methods the algorithm that selects the features uses a classification algorithm for evaluation. Accordingly, wrapper methods are more precise but computationally more complex (Wang *et al.*, 2011), and they also heavily depend on the data selected for classifier development. Since these data guide the selection, they can lead to over-fitting (Loughrey and Cunningham, 2004). A broad spectrum of various wrappers is used in today's approaches. For example, the forward and the backward floating search and their combinations are commonly used, where one feature is added or reduced at a time, depending on the classification accuracy, but also some advanced methods based on the aforementioned ones which take into account feature dependencies (Michalak and Kwaśnicka, 2006).

Evolutionary algorithms are also used. For instance, Raymer *et al.* (2000) employed a GA as a wrapper and a k-NN algorithm as a classifier. The dimensionality reduction problem was approached by these authors by using weight factors and one or more bit masks for feature selection. Kubir *et al.* (2011) presented a hybrid GA for feature selection. To enable a more refined search space exploration, they built a local search algorithm based on the difference and informative nature of features calculated based on correlation information in the GA. The classifier used was an ANN. Currently, hybrid approaches combining filter and wrapper methods or even RS and FSS are becoming more and more popular. One such method was presented by Hsu *et al.* (2011). Firstly they combined the F-Score filter, which calculates discriminatory possibilities of each feature, and the information gain filter, which selects features that contain more information. Through this filter combination they select the candidates and create a new, reduced feature set. After that a wrapper method is used for FSS. Sequential floating search is used as a wrapper and an SVM with a radial basis function kernel as the classifier.

The k-nearest neighbour algorithm is a classifier that works based on the classes of k nearest vectors from $\mathcal{S}$, which are nearest to the input vector based on some distance measure (Zhua *et al.*, 2007). Distance measures used to determine the relation between two samples can be distance functions such as Euclidean, Mahalanobis, Minkowski, Manhattan and others. In this paper the Euclidean distance is used as the distance measure. The algorithm parameter k determines the number of neighbours from $\mathcal{S}$ to be used when the label of the input vector is chosen. As stated by Garcia *et al.* (2010), the most common way to determine the value of k is through the process of cross-validation. Most often, k is selected as an odd number between 1 (where the classifier is reduced to the Nearest Neighbour (NN) classifier) and several tens. The k-NN algorithm is one of simpler classifiers, but due to its simplicity and good performance it is often used in classification problems. In time it has been improved and enhanced according to the needs of various problems. An overview of different versions is given by, e.g., Bhatia and Vandana (2010) or Jiang *et al.* (2007).

3. Differential evolution

Differential Evolution, or briefly DE (Storn and Price, 1997; Price *et al.*, 2005; Xinjie and Mitsuo, 2010), is a simple but effective search method for continuous optimization problems. According to Xinjie and Mitsuo (2010), DE represents a direction based search that maintains a vector population of candidate solutions. Like other usual Evolutionary Algorithms (EAs), it uses mutation, crossover and selection. The key part of DE, which differentiates it from standard EAs, is the mutation operator that perturbs the selected vector according to the scaled difference of the other two members of the population. The operation of DE is shown with pseudo-code as Algorithm 1.

Algorithm 1. DE in pseudo-code.

```
Initialization and parameter setting
while termination condition not met do
  for all population member—vector v_i do
    create mutant vector u_i
    crossover v_i and u_i to create trial vector t_i
  end for
  for all population member—vector v_i do
    if f(t_i) ≤ f(v_i) then
      v_i ← t_i
    end if
  end for
end while
```

The population of size NP contains vectors and each vector $\mathbf{v}_i$, of dimensionality D, consists of real-valued parameters, $\mathbf{v}_i = (v_i^1, \ldots, v_i^D) \in \mathbb{R}^D$, for $i = 1, \ldots, NP$. Usually the population is initialized with vectors of values obtained randomly in the interval $[v_{lb}, v_{ub})$, where v_{lb} and v_{ub} represent the lower and upper bound, respectively. In each generation, a new population is created through mutation and crossover. This new population is composed of the so-called trial vectors $\mathbf{t}_i$. For each member of the current population, $\mathbf{v}_i$ (called the target vector), a new corresponding mutant vector $\mathbf{u}_i$ is formed using mutation. The mutation is conducted according to

$$\mathbf{u}_i = \mathbf{v}_{r1} + F \cdot (\mathbf{v}_{r2} - \mathbf{v}_{r3}) \,. \tag{3}$$

Here $\mathbf{u}_i$ is a mutant while $\mathbf{v}_{r1}$, $\mathbf{v}_{r2}$ and $\mathbf{v}_{r3}$ are population vectors selected randomly with the condition $i \neq r1 \neq r2 \neq r3$, and $F \in [0, \infty)$ is the scale factor which represents a parameter of the algorithm. After the mutation, crossover occurs between the target vector $\mathbf{v}_i$ and the corresponding mutant $\mathbf{u}_i$ creating a trial vector $\mathbf{t}_i$. The crossover is done as follows:

$$\mathbf{t}_i^j = \begin{cases} \mathbf{u}_i^j & \text{if } U[0,1) \leq \mathrm{CR} \text{ or } j = r_j, \\ \mathbf{v}_i^j & \text{otherwise} \end{cases} \tag{4}$$

for $j = 1, \ldots, D$. Here $\mathbf{t}_i$ is a trial vector obtained through crossover, $U[0,1)$ is a variable with its value randomly selected from the interval $[0,1)$ with uniform distribution, r_j is a random variable with the value from the set $\{1, \ldots, D\}$, while $\mathrm{CR} \in [0,1)$ is the crossover rate and represents a parameter of the algorithm. The described crossover is called the binomial crossover.

Once the trial vector population has been created, vectors that transfer over to the next generation, i.e., which

will constitute the new population, are selected. A given trial vector $\mathbf{t}_i$ replaces the corresponding target vector $\mathbf{v}_i$ if it is of equal or lesser cost, according to the given objective/fitness function.

Due to its simplicity, DE is a very popular search method that has been successfully applied to various problems. Here, the classic DE is described that is commonly denoted by DE/rand/1/bin (Storn and Price, 1997). However, various other variants/strategies are presented in the literature. A very comprehensive overview of different DE strategies, as well as application areas, is given by Das and Suganthan (2011).

3.1. Differential evolution within the binary space. Differential evolution was originally proposed for solving continuous optimization problems. Primarily, because of the nature of the mutation operator, DE is not directly applicable to discrete optimization problems. Still, it is possible to use it for discrete optimization, and the literature (e.g., Lichtblau, 2012; Vegh *et al.*, 2011; Zhang *et al.*, 2008) proposes different ways of achieving the aforementioned. Several different approaches for applying DE to binary optimization problems have been proposed (Engelbrecht and Pampara, 2007).

Angle Modulated DE (AMDE), proposed by Pampara *et al.* (2006), boils down the basic problem (in the binary space) to a simpler one in the continuous space. AMDE optimizes the coefficients of a function $h(x)$, given in (5), in the continuous space which will be used for the creation of a binary vector. AMDE reduces the D-dimensional problem in the binary space to a 4-dimensional problem in continuous space,

$$h(x) = \sin(2\pi(x-a)\cdot b\cdot\cos(2\pi(x-a)\cdot c)) + d. \quad (5)$$

Here x is an independent variable while a, b, c, and d are the function coefficients that determine its shape (usually constrained to the range $[-1, 1]$). The binary vector is obtained by sampling the given function in equal intervals. If the function has a positive value in the given interval, a 1 is inserted, and a 0 otherwise.

An alternative, simple and intuitive approach is to use vectors of D dimensions, i.e., vectors of the same size as the problem in the binary space. The real-valued parameters are constrained to the range $[0, 1]$ and a binary vector is then derived from it by setting a 0 if the corresponding real-valued parameter is less than 0.5, or, otherwise, by setting a 1. We adopt this approach in the proposed algorithm since it proved, in our preliminary analysis, to yield better solutions than AMDE. Also, it resulted in a more stable algorithm.

In both of the aforementioned approaches the quality of each solution, i.e., the population member, is determined based on the evaluation of the obtained binary vector, using a fitness function that is defined in the binary space.

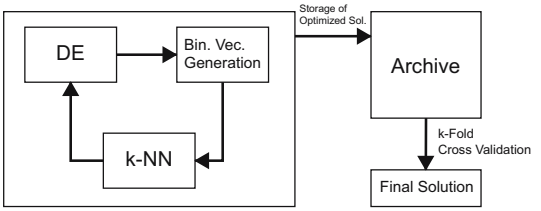


Fig. 1. Mode of operation of DE-kNN.

4. Proposed approach

As mentioned earlier, in this paper an approach is proposed based on FSS using a wrapper method. Accordingly, the proposed approach, hereinafter referred to as DE-kNN, combines the DE and k-NN algorithms. The former is used as a wrapper in the search for a subset of the given feature set of good quality, and the latter is used for the needs of evaluating found solutions. The employed DE for FSS, DE-kNN, is based on DE/rand/1/bin as previously described in Section 3. In DE-kNN the scale factor F is varied randomly as proposed by Das *et al.* (2005). More precisely, the scale factor used during mutation of a particular vector has a uniform random value in the range $[0.5, 1]$ – $F = 0.5\cdot(1+\mathcal{U}[0,1))$. According to Das *et al.* (2005), the randomly varied scale factor should help maintain the population diverse throughout the search. Also, DE-kNN incorporates an archive of fixed size which is used to store good quality solutions found during the optimization process, i.e., search. The mode of operation of DE-kNN is shown in Fig. 1.

The archive size was set to 50 to include a variety of good quality solutions found during the search. The population is initialized randomly in DE-kNN with $\mathbf{v}_i^j = U[0,1)$ for every $i = 1,\ldots,NP$ and $j = 1,\ldots,n$. Also, the first 50 solutions are copied to the archive, unless the population is smaller, in which case all population members are copied to the archive and the rest $(50-NP)$ is randomly generated. The population of trial vectors is created according to the selected strategy, DE/rand/1/bin. During the selection of vectors that will comprise the new population, a new binary vector is formed using the process described in Section 3.1 for each member of the current population and the trial vector population. The size of the binary vector corresponds to the number of features for the given classification problem. The binary vector, $\mathbf{b}_i = (b_i^1,\ldots,b_i^n) \in \{0,1\}^n$, determines which features will be used (1), and which will not (0) during classification. This vector is evaluated with the following negative value fitness function (Yang *et al.*, 2011),

$$g(\mathbf{b}_i) = o(\mathbf{b}_i) - \lambda\cdot p(\mathbf{b}_i), \quad (6)$$

where $o(\mathbf{b}_i)$ is the percentage of the classification accuracy using the k-NN algorithm, $p(\mathbf{b}_i)$ is the feature

subset size, i.e., the number of features used, and λ is a penalty factor which has, in this paper, a value of 0.001. Larger values of $g(\mathbf{b})$ indicate small feature subsets that yield high classification accuracy.

If any of the binary vectors contains only 0s, it is penalized, i.e., its fitness is set to a relatively very high constant value (in our case this value is 1000) to ensure the elimination of such a solution from transferring into the next generation,

Each time a trial vector is generated, it is considered for inclusion in the archive. A trial vector enters the archive only if it is better than the current worst and different from all the vectors present in the archive. The vectors in the archive are differentiated via the Hamming distance, which must be at least 1 (between the considered vector and each archive vector) for a vector to be granted access.

Once the DE execution is finished, a final solution is obtained from the archive. The motivation for using the archive is the fact that the optimization process tends to over-fit to training data resulting in loss of generality. As described, the archive contains distinct solutions of good quality, and the final solution is chosen from among them. The choice is made based on the quality of each solution obtained from a k-fold cross-validation. The k-fold cross-validation is a commonly employed cross-validation method that makes clever use of the available data. Thus, it should provide a more general evaluation of feature subsets compared with a single k-NN run. The archive solution that performed best in the cross-validation is chosen as the final solution. A motivation for this approach stems from the fact that it is possible to achieve similar classification quality with various feature subsets. Adding cross-validation and forcing the archive to hold only distinct feature vectors, but only those of high quality, provides an option to deduce which of them performs best on various test sets generated by the cross-validation. A solution that prevails should prove to be good in general. This way, the knowledge about the feature vectors that perform best is maintained throughout the search, and only a step of post-evaluation is added. It is worth mentioning that using the k-fold cross-validation during the optimization process would be computationally too demanding, but applying it to the archive solutions does not produce a significant computational overhead since the archive is relatively small.

5. Experimental analysis

With the purpose of evaluating the effectiveness of the proposed approach, DE-kNN, an experimental analysis was conducted on several datasets. All datasets were obtained from the UCI machine learning repository (Frank and Asuncion, 2010), except the *Texture* (Alcalá-Fdez *et al.*, 2011) dataset. The analysis is based on the evaluation of classification accuracy applying the k-NN algorithm with the full feature set and using the same measure and classifier with the use of the reduced feature set.

Table 1. Datasets used in the experimental analysis.

Dataset	Data-type	# inst.	# feat.	# class
Breast Can. Wis. (Or.)	Int.	683	9	2
Dermatology	Cat., Int.	358	34	6
Glass id.	Real	214	9	6
Image seg.	Real	2310	19	7
Ionosphere	Int., Real	351	34	2
Musk version 1	Int.	476	166	2
Libras Mov.	Real	360	90	15
Parkinsons	Real	197	22	2
100 plant sp. leaves	Real	1600	64	100
Spambase	Int., Real	4597	57	2
Statlog (Veh. Silh.)	Int.	946	18	4
Texture	Int., Real	5500	40	11

The selected datasets and their characteristics after preprocessing are shown in Table 1. The sets were chosen to cover various parameter values in order to test different input pattern cases, i.e., the influence of the number of instances, feature set size and similar on the feature selection process results. The datasets were preprocessed to remove the features that are known to have no meaning for the classification, such as the instance name, the ordinal number of an instance and similar. Instances with missing features were not considered.

5.1. Genetic algorithm used for comparison. Genetic algorithms (Xinjie and Mitsuo, 2010) are optimization and search methods inspired by genetics and natural selection, originally proposed for binary optimization problems. GAs have been successfully applied to a wide variety of problems (Yan *et al.*, 2013; Martinović and Bajer, 2011). In order to better evaluate the performance of DE-kNN, it was compared to a GA. A generational GA with binary tournament selection, one-point crossover and bit-flip mutation was implemented. Also, elitism was incorporated. A high-level outline of the GA used is given in Algorithm 2.

Binary tournament selection without replacement was chosen because it is easy to implement, has a small time complexity and exhibits a relatively small selection pressure. One-point crossover and bit-flip mutation were chosen since they are employed in the simple GA (Eiben and Smith, 2003; Xinjie and Mitsuo, 2010); also, according to Eiben and Smith (2003), bit-flip mutation is the most common mutation operator for binary encodings. Elitism was incorporated since it can significantly enhance GA performance, and it was implemented as follows. The best individual in the current population replaces the worst in the new (offspring) population if it does not contain an equally good or better individual.

In the GA, the population is initialized randomly, i.e.,

every individual in population is a randomly generated binary vector. The fitness of very individual is calculated as in Eqn. (6).

Algorithm 2. GA: high-level outline.

```
Initialization and parameter setting
while termination condition not met do
    while offspring population not complete do
        select 2 distinct parents form current population
        cross over parents with probability p_c to produce 2 offspring
        mutate offspring with probability p_m
        evaluate offspring
    end while
    replace current population with offspring population
    apply elitism
end while
```

5.2. Experiment set-up. The experimental analysis was carried out on a computer with a dual core processor (Intel E5800 @ 3.20 GHz), 4 GB of RAM and Windows 7 OS. Due to the sequential algorithm implementation only one core was utilized during algorithm execution. The performance of DE-kNN was compared with a number of optimization algorithms which were used as wrappers. More precisely, it was compared with a GA (as described in Section 5.1), DE/rand/1/bin (adapted for the binary space the same way as DE-kNN, and hereinafter referred to as stdDE) and AMDE. The aforementioned algorithms were implemented in the C# programming language.

Before the start of the experiment, all data were normalized as

$$\mathbf{N}_i^j = \begin{cases} 1 & \text{if } \Delta = 0 \\ 1 + \dfrac{9}{\Delta}\left(\mathbf{X}_i^j - \min\limits_{s=1,\ldots,m} \mathbf{X}_s^j\right) & \text{otherwise,} \end{cases} \tag{7}$$

where

$$\Delta = \max_{s=1,\ldots,m} \mathbf{X}_s^j - \min_{s=1,\ldots,m} \mathbf{X}_s^j,$$

and $\mathbf{N}_i^j$ is the normalized value of the feature $\mathbf{X}_i^j$, where $i = 1, \ldots, m$ and $j = 1, \ldots, n$. If the difference between the maximum and minimum values is 0, then it is clear that the value of this feature is equal for all feature vectors in the dataset. The normalized value is then set as 1 (for all feature vectors in the set) and it does not have any influence on the classification. Accordingly, it is expected that such a feature will be eliminated. Through normalization the values of all features were converted to values from the interval $[1, 10]$, which reduced the influence of the difference between different features on the classification results, as described by Raymer *et al.* (2000). In other words, the prevailing influence of one feature is disabled if the only reason for this influence is the range of its values. For instance, the influence of human height would be greater than that of width just because a person is usually taller than wider. In the case of non-numerical data, other distance measures can be employed (Li and Li, 2010), or several binary attributes can be added (1 representing the existence and 0 non-existence of the attribute), enabling the normalization of the feature and creating the same distance among categories. Once normalized, the dataset is divided into three proper subsets. The first subset contains 50% of the total feature vectors and represents the training set. These are the feature vectors with known class labels based on which the classification is performed. The second set contains 25% of the total feature vectors and represents the tuning set. These are the feature vectors used by the algorithms to determine which features are rejected, i.e., this set enables the k-NN algorithm to determine the quality of the solutions found by the wrapper algorithms. The third subset is the test subset which contains 25% of the vectors from the dataset. This subset is used to evaluate the final solution, i.e., the feature subset obtained by the one of the algorithms. Feature vectors selected for any of these sets are chosen from the initial dataset of feature vectors randomly, while maintaining the class distribution of the original set.

The parameter values for all algorithms used in the experimental analysis, obtained through extensive preliminary analysis, are displayed in Table 2. Furthermore, even though the proposed approach was designed with the general case in mind, the value of parameter k for the k-NN algorithm for a given input dataset is 1, thus creating its special case, the nearest neighbour algorithm. This parameter was used to alleviate the influence of parameter determination on the achieved results, since it should be determined through experimentation and the focus of this research was not classifier performance.

The value of k used in the k-fold cross-validation of archive solutions was set based on the size of the dataset used. Accordingly, its value was 3 for datasets containing less than 500 instances, 5 for datasets containing less than 1000 instances, and 10 for datasets containing 1000 or more instances. This way, the folds were of reasonable size. The union of both the training and tuning subsets was used for the creation of the folds in order to provide the largest possible number of data for the cross-validation.

The population size and the maximum iteration number were 50 and 300, respectively, and were the same for all algorithms. Also, the termination condition was the same for all algorithms. More precisely, the algorithm execution is terminated if it reaches the assumed maximum number of iterations or earlier, if in 30 consecutive runs a solution of higher quality was not found. Relatively small values were used to keep the total

number of evaluations on a reasonable level since each evaluation is time consuming. Since all the employed algorithms are stochastic search methods, ten independent runs were carried out for each algorithm and dataset. For each of these solutions (feature subsets), as well as for the full feature set, the NN algorithm was executed only once because it is a deterministic one.

5.3. Results and discussion. The first part of the experimental analysis results is presented in Table 3 and Fig. 2. The results show the datasets used, and display the mean classification accuracy (μ) that each of the employed algorithms achieved. The table also displays the standard deviation (σ), maximum ($\mathbf{bst}$), minimum ($\mathbf{wst}$) and range (calculated as $\mathbf{bst} - \mathbf{wst}$) of the classification accuracy for each algorithm. Also, a statistical analysis of the pairwise comparison of the performance in terms of the resulting classification accuracy of DE-kNN with the other wrappers considered is shown in Table 4. The table includes the sum of ranks (W), the obtained p-value, and the corresponding 95% confidence interval. The statistical analysis was performed using the Mann-Whitney U (Wilcoxon rank-sum test) test—two-sided test, provided by the R software environment for statistical computing (R Core Team, 2013). This test was chosen since, according to Trawiński *et al.* (2012), it is more sensible than the t-test when the number of observations is small (10 in our case).

As can be seen from the table, the proposed approach shows promising results. In most (7 out of 12) test cases it outperforms the other feature selection algorithms (wrappers considered) and yields a higher classification accuracy than the NN algorithm. According to Table 4, the higher performance of DE-kNN compared with the other wrappers is in most cases statistically significant. The performance improvements are most notable on the *Breast Can. Wis. (Or.)*, *Glass identification*, and *Image segmentation* datasets. The improvements are shown to be statistically significant. In several cases (e.g., *Spambase* and *Parkinsons*), considerably higher performance is achieved compared with the other wrappers, and it is shown to be statistically significant compared with two of the three utilized wrapper methods. In several cases the deterministic NN algorithm shows the best performance, but even then the difference is small and DE-kNN outperforms other tested wrappers. The accuracy it achieves is close to that of the NN but is attained with far fewer features. Since classifier development has to be performed only once, the cost should prove its worth in the long run since using fewer features use less time to classify an unknown sample. It should be noted that the costs involved might not only be related to time consumption, so the performance and speed should be weighted on the case-to-case basis. The performance of the wrapper depends on the fitness function guiding the search process, and the over-fitting occurring due to the adaptation of the solutions to the data used in classifier development could lead to under-performance on the independent set of data used for testing. Based on these remarks, it can be concluded that, as discussed in Section 1, each of the datasets used contains some irrelevant or redundant features and/or some that are detrimental to the classification accuracy.

The second part of the results is shown in Table 5 and Fig. 3. They represent feature reduction results and display the mean number of features (μ) that each of

Table 2. Algorithm parameters used.

Algorithm	Parameters
DE-kNN	$CR = 0.95$, $F = 0.5 \cdot (1 + \mathcal{U}[0, 1))$
GA	$p_c = 0.9$, $p_m = 0.03$
stdDE	$CR = 0.95$, $F = 0.3$
AMDE	$CR = 0.95$, $F = 0.25$

Table 3. Classification accuracy.

Dataset		NN	GA	stdDE	AMDE	DE-kNN
B.C.W. Or.	μ	0.9647	0.9176	0.9217	0.9176	0.9518
	bst	0.9647	0.9176	0.9588	0.9176	0.9588
	wst	0.9647	0.9176	0.9176	0.9176	0.9353
	σ	0.0000	0.0000	0.0130	0.0000	0.0082
	range	0.0000	0.0000	0.0412	0.0000	0.0235
Dermatology	μ	0.9659	0.9295	0.9239	0.9488	0.9443
	bst	0.9659	0.9545	0.9659	0.9545	0.9773
	wst	0.9659	0.8864	0.8977	0.9318	0.8977
	σ	0.0000	0.0238	0.0215	0.0080	0.0265
	range	0.0000	0.0681	0.0682	0.0227	0.0796
Glass	μ	0.6731	0.7115	0.7134	0.7115	0.7596
	bst	0.6731	0.7115	0.7308	0.7115	0.8269
	wst	0.6731	0.7115	0.7115	0.7115	0.7115
	σ	0.0000	0.0000	0.0061	0.0000	0.0437
	range	0.0000	0.0000	0.0193	0.0000	0.1154
Image seg.	μ	0.9460	0.9469	0.9464	0.9526	0.9606
	bst	0.9460	0.9469	0.9464	0.9526	0.9606
	wst	0.9460	0.9443	0.9443	0.9460	0.9547
	σ	0.0000	0.0040	0.0019	0.0039	0.0038
	range	0.0000	0.0121	0.0052	0.0104	0.0122
Ionosphere	μ	0.8391	0.8656	0.8587	0.8690	0.8793
	bst	0.8391	0.8966	0.8966	0.9425	0.9080
	wst	0.8391	0.8276	0.8161	0.8161	0.8391
	σ	0.0000	0.0217	0.0282	0.0364	0.0225
	range	0.0000	0.0690	0.0805	0.1264	0.0689
Musk 1	μ	0.8390	0.8339	0.8356	0.8441	0.8585
	bst	0.8390	0.8983	0.8729	0.8898	0.8983
	wst	0.8390	0.7627	0.8051	0.7966	0.8220
	σ	0.0000	0.0400	0.0230	0.0247	0.0256
	range	0.0000	0.1356	0.0678	0.0932	0.0763
Libras Mov.	μ	0.8778	0.8489	0.8600	0.8533	0.8700
	bst	0.8778	0.8889	0.8889	0.9000	0.9000
	wst	0.8778	0.8111	0.8222	0.8111	0.8444
	σ	0.0000	0.0217	0.0175	0.0309	0.0158
	range	0.0000	0.0778	0.0667	0.0889	0.0556
Parkinson	μ	0.9375	0.9292	0.9333	0.9458	0.9500
	bst	0.9375	0.9583	0.9583	0.9583	0.9583
	wst	0.9375	0.9167	0.9167	0.9167	0.9375
	σ	0.0000	0.0145	0.0132	0.0145	0.0107
	range	0.0000	0.0416	0.0416	0.0416	0.0208
100 plants	μ	0.5950	0.5910	0.5905	0.5948	0.5900
	bst	0.5950	0.6050	0.6000	0.6050	0.6050
	wst	0.5950	0.5750	0.5750	0.5750	0.5775
	σ	0.0000	0.0094	0.0067	0.0083	0.0080
	range	0.0000	0.0300	0.0250	0.0300	0.0275
Spambase	μ	0.8895	0.9065	0.9036	0.8844	0.9169
	bst	0.8895	0.9234	0.9112	0.8956	0.9304
	wst	0.8895	0.8825	0.8903	0.8747	0.8999
	σ	0.0000	0.0136	0.0067	0.0074	0.0113
	range	0.0000	0.0409	0.0209	0.0209	0.0305
Statlog	μ	0.6381	0.6871	0.6881	0.6852	0.7081
	bst	0.6381	0.6952	0.7143	0.7286	0.7476
	wst	0.6381	0.6714	0.6619	0.6381	0.6524
	σ	0.0000	0.0093	0.0168	0.0232	0.0313
	range	0.0000	0.0238	0.0524	0.0905	0.0952
Texture	μ	0.9855	0.9820	0.9835	0.9838	0.9839
	bst	0.9855	0.9855	0.9862	0.9862	0.9884
	wst	0.9855	0.9789	0.9775	0.9796	0.9818
	σ	0.0000	0.0018	0.0030	0.0026	0.0020
	range	0.0000	0.0066	0.0087	0.0066	0.0066

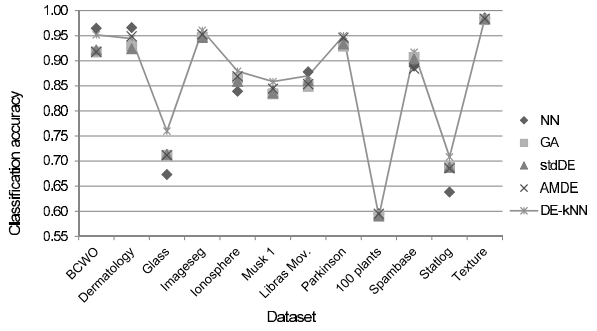


Fig. 2. Comparison of the classification accuracy of the NN algorithm using the full feature set and the average classification accuracy of the NN using the solutions found by wrapper algorithms.

the employed algorithms reduced the full set to. The table also contains the standard deviation (σ), minimum ($\mathbf{bst}$), maximum ($\mathbf{wst}$) and range ($\mathbf{bst} - \mathbf{wst}$) of the number of features for each algorithm. Since the goal of the proposed approach is not to achieve the minimum amount of features, but to generate the best general solution, it is understandable that the proposed approach is not producing feature vectors with minimum features. However, it still gives reasonable reduction and in some cases (e.g., *Statlog Vehicle Silhouettes*) even achieves the best results with the smallest feature subset. When compared with the full feature set, the result is substantial for each of the wrappers considered since the average feature number across all datasets is more than halved. All datasets display a high degree of feature reducibility, and that fact is even more evident on datasets with higher dimensional data (e.g., *Libras Movement*).

It is interesting to observe that the ratio of feature set size and the average size of the feature subset found by the DE-kNN and other algorithms varies for different datasets. If the dataset is quite large, DE-kNN will not necessarily discard a substantial amount of the features and vice versa, which is noticeable from the *Glass identification*, *Parkinsons* and *Libras Movement* datasets. This is understandable since the reduction can be pursued up to a certain level that depends on the data in the dataset and the fitness function that determines the quality of an individual feature subset, which is given in (6) for the proposed approach, and for the other considered wrappers as well. Furthermore, the proposed approach evaluates candidates not based on the reduction, but on their performance on k-fold cross-validation. Although the candidates are all fairly reduced, it is quite possible that although the most reduced feature set performs exquisitely on the tuning set, it is not so good in general. Therefore the additional features included in the solution by DE-kNN in regard to the sets provided by other tested wrappers is justified, since the achieved accuracy is

Table 4. Statistical analysis of the pairwise comparison of DE-kNN with the other wrappers.

Dataset	W	p-Value	95% Confidence interval	Significance
GA–DE-kNN				
B.C.W. Or.	0	<0.0001	-0.0412 to -0.0295	Extremely significant
Dermatology	34	0.2199	-0.0342 to 0.0112	Not significant
Glass	5	0.0002	-0.0962 to -0.0193	Extremely significant
Image seg.	2	0.0002	-0.0174 to -0.0104	Extremely significant
Ionosphere	32.5	0.18	-0.0345 to 0.0115	Not significant
Musk 1	32	0.172	-0.0593 to 0.0085	Not significant
Libras Mov.	20.5	0.0233	-0.0335 to -2.5207e-05	Significant
Parkinson	15	0.0048	-0.0415 to -6.9761e-05	Very significant
100 plants	55.5	0.6757	-0.0076 to 0.01	Not significant
Spambase	27	0.0818	-0.0235 to 0.0009	Weakly significant
Statlog	26	0.0678	-0.0477 to 0.0095	Weakly significant
Texture	22	0.0326	-0.0031 to -0.00002	Significant
stdDE–DE-kNN				
B.C.W. Or.	8	0.0008	-0.0411 to -0.0236	Extremely significant
Dermatology	27.5	0.0862	-0.0454 to 4.7788e-05	Weakly significant
Glass	7.5	0.0005	-0.0962 to -0.0193	Extremely significant
Image seg.	0	0.0001	-0.01745 to -0.012	Extremely significant
Ionosphere	28	0.0883	-0.04601 to 3.6177e-05	Weakly significant
Musk 1	25	0.0577	-0.0508 to 7.3172e-5	Weakly significant
Libras Mov.	33.5	0.1988	-0.0223 to 0.011	Not significant
Parkinson	19	0.0101	-0.0209 to -5.9978e-05	Significant
100 plants	55	0.702	-0.005 to 0.0076	Not significant
Spambase	20	0.0232	-0.0218 to -0.0034	Significant
Statlog	27	0.081	-0.0477 to 0.0094	Weakly significant
Texture	53.5	0.7899	-0.0022 to 0.0022	Not significant
AMDE–DE-kNN				
B.C.W. Or.	0	<0.0001	-0.0412 to -0.0295	Extremely significant
Dermatology	53	0.8138	-0.0227 to 0.0228	Not significant
Glass	5	0.0002	-0.0962 to -0.0193	Extremely significant
Image seg.	6	0.0007	-0.0105 to -0.0052	Extremely significant
Ionosphere	37.5	0.3318	-0.0344 to 0.0115	Not significant
Musk 1	35.5	0.2707	-0.0339 to 0.0085	Not significant
Libras Mov.	30.5	0.1349	-0.0444 to 0.0111	Not significant
Parkinson	43	0.5469	-0.0208 to 0.00006	Not significant
100 plants	70	0.128	-0.0025 to 0.0125	Not significant
Spambase	0	<0.0001	-0.0435 to -0.0226	Extremely significant
Statlog	27.5	0.0879	-0.0524 to 0.0095	Weakly significant
Texture	54.5	0.7311	-0.0022 to 0.0022	Not significant

higher for the independent test set containing previously unknown data.

It is worth noting that it is possible to achieve the same classification accuracy with feature subsets of different sizes, and also different classification accuracy with subsets of the same size. In the latter case, even though these solutions have the same amount of features, they are not necessarily the same features. The aforementioned is not strange since the search is guided by the samples contained in the tuning subset of the dataset which are different than the ones in the independent test subset. That signifies the possibility of different feature subsets achieving the same or very similar classification accuracy on the tuning subset, but a different accuracy on

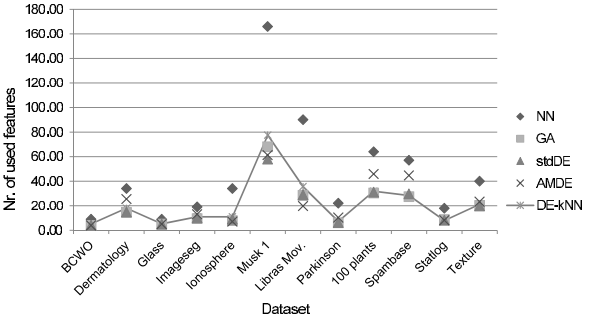


Fig. 3. Number of employed features per algorithm for each of the datasets used.

Table 5. Feature reduction analysis.

Dataset		NN	GA	stdDE	AMDE	DE-kNN
B.C.W. Or.	μ	9.00	4.00	4.10	4.00	5.20
	bst	9.00	4.00	4.00	4.00	4.00
	wst	9.00	4.00	5.00	4.00	7.00
	σ	0.00	0.00	0.32	0.00	0.92
	range	0.00	0.00	-1.00	0.00	-3.00
Dermatology	μ	34.00	14.50	14.80	25.40	18.00
	bst	34.00	10.00	14.00	19.00	16.00
	wst	34.00	17.00	16.00	30.00	19.00
	σ	0.00	1.90	0.92	4.62	1.15
	range	0.00	-7.00	-2.00	-11.00	-3.00
Glass	μ	9.00	5.00	4.90	5.00	5.30
	bst	9.00	5.00	5.00	5.00	7.00
	wst	9.00	5.00	4.00	5.00	4.00
	σ	0.00	0.00	0.32	0.00	1.06
	range	0.00	0.00	1.00	0.00	3.00
Image seg.	μ	19.00	9.50	10.10	13.00	11.00
	bst	19.00	8.00	9.00	11.00	10.00
	wst	19.00	11.00	11.00	15.00	14.00
	σ	0.00	1.08	0.99	1.41	1.41
	range	0.00	-3.00	-2.00	-4.00	-4.00
Ionosphere	μ	34.00	7.90	9.00	7.10	11.00
	bst	34.00	6.00	7.00	3.00	8.00
	wst	34.00	10.00	12.00	10.00	13.00
	σ	0.00	1.20	1.56	2.64	1.41
	range	0.00	-4.00	-5.00	-7.00	-5.00
Musk 1	μ	166.00	68.00	58.00	61.20	77.50
	bst	166.00	54.00	48.00	45.00	71.00
	wst	166.00	81.00	67.00	101.00	85.00
	σ	0.00	8.71	5.87	18.52	4.77
	range	0.00	-27.00	-19.00	-56.00	-14.00
Libras Mov.	μ	90.00	28.40	28.80	19.70	35.60
	bst	90.00	24.00	18.00	9.00	28.00
	wst	90.00	34.00	37.00	35.00	41.00
	σ	0.00	3.69	5.20	8.31	4.12
	range	0.00	-10.00	-19.00	-26.00	-13.00
Parkinson	μ	22.00	6.10	6.20	10.30	7.20
	bst	22.00	8.00	8.00	15.00	8.00
	wst	22.00	5.00	5.00	7.00	6.00
	σ	0.00	1.37	1.03	2.54	0.79
	range	0.00	3.00	3.00	8.00	2.00
100 Plants s. l.	μ	64.00	30.30	31.00	45.80	31.80
	bst	64.00	26.00	24.00	40.00	27.00
	wst	64.00	33.00	37.00	52.00	38.00
	σ	0.00	2.00	3.65	4.29	3.71
	range	0.00	-7.00	-13.00	-12.00	-11.00
Spambase	μ	57.00	27.40	29.60	44.60	28.30
	bst	57.00	23.00	26.00	38.00	23.00
	wst	57.00	32.00	32.00	49.00	35.00
	σ	0.00	2.99	1.65	3.41	3.23
	range	0.00	-9.00	-6.00	-11.00	-12.00
Statlog V.S.	μ	18.00	8.70	8.20	8.10	7.90
	bst	18.00	7.00	6.00	6.00	5.00
	wst	18.00	10.00	10.00	13.00	10.00
	σ	0.00	1.34	1.40	2.18	1.45
	range	0.00	-3.00	-4.00	-7.00	-5.00
Texture	μ	40.00	20.30	19.70	23.30	21.20
	bst	40.00	19.00	17.00	19.00	17.00
	wst	40.00	23.00	23.00	28.00	26.00
	σ	0.00	1.34	2.00	2.67	2.86
	range	0.00	-4.00	-6.00	-9.00	-9.00

Table 6. Execution time analysis.

Dataset		NN	GA	stdDE	AMDE	DE-kNN
B.C.W. Or.	μ	-	2.80	3.00	4.10	3.00
	bst	-	2.00	2.00	3.00	3.00
	wst	-	3.00	4.00	6.00	3.00
	σ	-	0.42	0.47	0.88	0.00
	range	-	-1.00	-2.00	-3.00	0.00
Dermatology	μ	-	7.30	7.50	5.60	8.10
	bst	-	4.00	5.00	3.00	4.00
	wst	-	11.00	10.00	10.00	11.00
	σ	-	2.16	1.58	2.01	2.08
	range	-	-7.00	-5.00	-7.00	-7.00
Glass	μ	-	0.00	0.00	0.00	0.00
	bst	-	0.00	0.00	0.00	0.00
	wst	-	0.00	0.00	0.00	0.00
	σ	-	0.00	0.00	0.00	0.00
	range	-	0.00	0.00	0.00	0.00
Image seg.	μ	-	40.60	39.60	26.40	29.20
	bst	-	12.00	11.00	1.00	1.00
	wst	-	58.00	57.00	53.00	56.00
	σ	-	16.01	12.95	14.05	17.29
	range	-	-46.00	-46.00	-52.00	-55.00
Ionosphere	μ	-	9.30	6.90	5.80	9.00
	bst	-	6.00	5.00	3.00	5.00
	wst	-	13.00	10.00	9.00	14.00
	σ	-	2.16	1.45	2.10	3.27
	range	-	-7.00	-5.00	-6.00	-9.00
Musk 1	μ	-	32.50	23.40	40.00	25.10
	bst	-	17.00	2.00	14.00	5.00
	wst	-	51.00	42.00	59.00	56.00
	σ	-	10.68	14.54	14.97	17.85
	range	-	-34.00	-40.00	-45.00	-51.00
Libras Mov.	μ	-	33.80	38.40	16.70	31.00
	bst	-	21.00	25.00	9.00	1.00
	wst	-	47.00	58.00	29.00	56.00
	σ	-	9.44	11.30	7.39	14.71
	range	-	-26.00	-33.00	-20.00	-55.00
Parkinson	μ	-	0.90	0.70	0.20	1.00
	bst	-	1.00	1.00	1.00	1.00
	wst	-	0.00	0.00	0.00	1.00
	σ	-	0.32	0.48	0.42	0.00
	range	-	1.00	1.00	1.00	0.00
100 Plants s. l.	μ	-	25.20	27.10	38.90	40.90
	bst	-	0.00	5.00	9.00	22.00
	wst	-	45.00	57.00	58.00	60.00
	σ	-	16.03	18.69	17.69	12.05
	range	-	-45.00	-52.00	-49.00	-38.00
Spambase	μ	-	24.30	31.80	31.10	43.00
	bst	-	0.00	2.00	7.00	17.00
	wst	-	54.00	56.00	55.00	65.00
	σ	-	17.99	15.11	16.47	16.94
	range	-	-54.00	-54.00	-48.00	-48.00
Statlog V.S.	μ	-	16.70	14.30	13.50	17.80
	bst	-	12.00	11.00	10.00	10.00
	wst	-	22.00	22.00	26.00	30.00
	σ	-	3.16	3.47	5.25	6.49
	range	-	-10.00	-11.00	-16.00	-20.00
Texture	μ	-	24.40	38.00	27.00	49.00
	bst	-	6.00	11.00	7.00	19.00
	wst	-	52.00	55.00	55.00	67.00
	σ	-	13.73	14.02	16.17	17.01
	range	-	-46.00	-44.00	-48.00	-48.00

the test subset.

The final part of the analysis is given in Table 6 and Fig. 4. The results represent the execution time of each of the algorithms per dataset. Both show the average time while the table simultaneously presents the standard deviation (σ), minimum ($\mathbf{bst}$), maximum ($\mathbf{wst}$) and range ($\mathbf{bst} - \mathbf{wst}$) of the execution time for each algorithm.

As can be deduced from the presented data, the proposed approach has a slightly longer execution time. This is most evident on datasets with most instances. The reason lies in the fact that DE-kNN efficiently explores the search space for candidate solutions and then evaluates each candidate from the archive. This puts a strain on execution time as the number of available data increases (i.e., as the number of instances rises). On several datasets it was still the fastest of the wrappers compared, which indicates that it could be adjusted and improved to reduce the time. It could be possible to reduce the maximum number of wrapper evaluations (since it archives top solutions, it could still find good candidates), reduce the number of instances for post evaluation (use just a part of the development portion of the dataset), or find some other sort of compromise that would not jeopardize the quality of the final solution. AMDE proved to be the fastest, but on average it also yielded the largest feature subsets.

6. Conclusion

In this paper, an approach to dimensionality reduction in pattern classification was proposed. Feature subset selection using a wrapper method was used. A differential evolution algorithm was employed as a wrapper, while the nearest neighbour algorithm was applied as a classifier. The proposed approach, DE-kNN, consists of the classifier, the wrapper, the archive for storing solutions and a voting system. Differential evolution finds feature subsets of high quality from the full feature set which are then evaluated by the k-NN algorithm. A designated number of solutions found throughout DE execution is archived and a method of post-evaluation using k-fold cross-validation is used to generate the final solution.

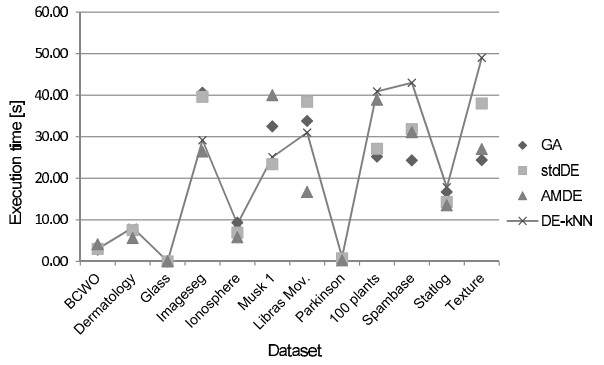


Fig. 4. Execution times per algorithm for each of the datasets used.

Experimental analysis was carried out on several standard datasets of varying sizes and numbers of features. The results of the analysis show the usefulness of this approach because in almost every case the results are better than the ones of the full feature set and achieve higher accuracy than other tested wrapper methods. The promising results achieved by the proposed method were statistically evaluated by a pairwise comparison between it and the other utilized wrappers using the Mann–Whitney U test. It was shown that the differences in performance were in most cases statistically significant. This means that, alongside the problem complexity reduction (through the reduction of features used), classification accuracy is improved.

Future work includes potential improvements of the proposed solution through combining it with some filter method to discard some features up-front and to reduce the execution time or by using a more advanced classifier such as an artificial neural network. It should also focus on adjusting the solution using other methods of post-evaluation, other than k-fold cross-validation and parameter optimization to achieve reduced running times while maintaining the solution quality.

Acknowledgment

This work was supported by the research project grant no. 165-0362980-2002 of the Ministry of Science, Education and Sports of the Republic of Croatia. The authors would like to thank the anonymous reviewers for their useful comments that helped improve this paper.

References

Alcalá-Fdez, J., Fernández, A., Luengo, J., Derrac, J. and García, S. (2011). KEEL data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework, *Multiple-Valued Logic and Soft Computing* **17**(2–3): 255–287.

Balakrishnan, S., Narayanaswamy, R., Savarimuthu, N. and Samikannu, R. (2008). SVM ranking with backward search for feature selection in type II diabetes databases, *Proceedings of the IEEE International Conference On System, Man and Cybernetics, Singapore*, pp. 2628–2633.

Bhatia, N. and Vandana, A. (2010). Survey of nearest neighbor techniques, *International Journal of Computer Science and Information Security* **8**(2): 302–305.

Chuang, L.-Y., Tsai, S.-W. and Yang, C.-H. (2011). Improved binary particle swarm optimization using catfish effect for feature selection, *Expert Systems with Applications* **38**(10): 12699–12707.

Das, S., Konar, A. and Chakraborty, U.K. (2005). Two improved differential evolution schemes for faster global search, *Proceedings of the 2005 Conference on Genetic and Evolutionary Computation, Washington DC, USA*, pp. 991–998.

Das, S. and Suganthan, P.N. (2011). Differential evolution: A survey of the state-of-the-art, *IEEE Transactions on Evolutionary Computation* **15**(1): 4–31.

Dash, M. and Liu, H. (1997). Feature selection for classification, *Intelligent Data Analysis* **1**(1–4): 131–156.

Debska, B. and Guzowska-Swider, B. (2011). Application of artificial neural network in food classification, *Analytica Chimica Acta* **705**(1–2): 283–291.

Duda, R., Hart, P. and Stork, D. (2001). *Pattern Classification, 2nd Edition*, Wiley and Sons Inc., New York, NY.

Eiben, A.E. and Smith, J.E. (2003). *Introduction to Evolutionary Computing*, Springer-Verlag, Berlin/Heidelberg.

Engelbrecht, A.P. and Pampara, G. (2007). Binary differential evolution strategies, *Proceedings of the IEEE Congress on Evolutionary Computation 2007, Singapore*, pp. 1942–1947.

Ferreira, A.J. and Figueiredo, M.A.T. (2012). Efficient feature selection filters for high-dimensional data, *Pattern Recognition Letters* **33**(13): 1794–1804.

Frank, A. and Asuncion, A. (2010). UCI machine learning repository, `http://archive.ics.uci.edu/ml`.

Garcia, E.K., Feldman, S., Gupta, M.R. and Srivastava, S. (2010). Completely lazy learning, *IEEE Transactions on Knowledge and Data Engineering* **22**(9): 1274–1285.

Gocławski, J., Sekulska-Nalewajko, J. and Kuźniak, E. (2012). Neural network segmentation of images from stained cucurbits leaves with colour symptoms of biotic and abiotic stresses, *International Journal of Applied Mathematics and Computer Science* **22**(3): 669–684, DOI: 10.2478/v10006-012-0050-5.

Hsu, C.-W. and Lin, C.-J. (2002). A comparison of methods for multiclass support vector machines, *IEEE Transactions on Neural Networks* **13**(2): 415–425.

Hsu, H.-H., Hsieh, C.-W. and Lu, M.-D. (2011). Hybrid feature selection by combining filters and wrappers, *Expert Systems with Applications* **38**(7): 8144–8150.

Jain, A.K., Duin, R.P.W. and Mao, J. (2000). Statistical pattern recognition: A review, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **22**(1): 4–37.

Javed, K., Babri, H. and Saeed, M. (2012). Feature selection based on class-dependent densities for high-dimensional binary data, *IEEE Transactions on Knowledge and Data Engineering* **24**(3): 465–477.

Jeleń, L., Fevens, T. and Krzyżak, A. (2008). Classification of breast cancer malignancy using cytological images of fine needle aspiration biopsies, *International Journal of Applied Mathematics and Computer Science* **18**(1): 75–83, DOI: 10.2478/v10006-008-0007-x.

Jiang, L., Cai, Z., Wang, D. and Jiang, S. (2007). Survey of improving k-nearest-neighbor for classification, *Proceedings of the 4th International Conference on Fuzzy Systems and Knowledge Discovery, Haikou, Hainan, China*, Vol.1, pp. 679–683.

Khushaba, R.N., Al-Ani, A. and Al-Jumaily, A. (2008). Differential evolution based feature subset selection, *Proceedings of the 19th International Conference on Pattern Recognition, Tampa, FL, USA*, pp. 1–4.

Kubir, M.M., Shahajan, M. and Murase, K. (2011). A new local search based hybrid genetic algorithm for feature selection, *Neurocomputing* **74**(17): 2914–2928.

Kubir, M.M., Shahajan, M. and Murase, K. (2012). A new hybrid ant colony optimization algorithm for feature selection, *Expert Systems with Applications* **39**(3): 3747–3763.

Li, C. and Li, H. (2010). A survey of distance metrics for nominal attributes, *Journal of Software* **5**(11): 1262–1269.

Lichtblau, D. (2012). Differential evolution in discrete optimization, *International Journal of Swarm Intelligence and Evolutionary Computation* **1**(2012): 1–10.

Loughrey, J. and Cunningham, P. (2004). Overfitting in wrapper-based feature subset selection: The harder you try the worse it gets, *in* M. Bramer, F. Coenen and T. Allen (Eds.), *The Twenty-fourth SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence*, Springer, Berlin/Heidelberg, pp. 33–43.

Martinović, G. and Bajer, D. (2011). Impact of double operators on the performance of a genetic algorithm for solving the traveling salesman problem, *in* B.K. Panigrahi, P.N. Suganthan, S. Das and S.C. Satapathy (Eds.), *Proceedings of the Second International Conference on Swarm, Evolutionary, and Memetic Computing Part I*, Springer-Verlag, Berlin/Heidelberg, pp. 290–298.

Michalak, K. and Kwaśnicka H. (2006). Correlation-based feature selection strategy in classification problems, *International Journal of Applied Mathematics and Computer Science* **16**(4): 503–511.

Pampara, G., Engelbrecht, A.P. and Franken, N. (2006). Binary differential evolution, *Proceedings of the IEEE Congress on Evolutionary Computation 2006, Vancouver, BC, Canada*, pp. 1873–1879.

Price, K.V., Storn, R.M. and Lampinen, J.A. (2005). *Differential Evolution. A Practical Approach to Global Optimization*, Springer-Verlag, Berlin/Heidelberg.

R Core Team (2013). *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, `http://www.R-project.org`.

Raymer, M.L., Punch, W.F., Goodman, E.D., Kuhn, L.A. and Jain, A.K. (2000). Dimensionality reduction using genetic algorithms, *IEEE Transactions on Evolutionary Computation* **4**(2): 164–171.

Storn, R. and Price, K. (1997). Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces, *Journal of Global Optimization* **11**(4): 341–359.

Trawiński, B., Smętek, M., Telec, Z. and Lasota, T. (2012). Nonparametric statistical analysis for multiple comparison of machine learning regression algorithms, *International Journal of Applied Mathematics and Computer Science* **22**(4): 867–881, DOI: 10.2478/v10006-012-0064-z.

Vegh, V., Pierens, G.K. and Tieng, Q.M. (2011). A variant of differential evolution for discrete optimization problems requiring mutually distinct parameters, *International Journal of Innovative Computing, Information and Control* **7**(2): 897–914.

Wang, G., Jian, M. and Yang, S. (2011). IGF-bagging: Information gain based feature selection for bagging, *International Journal of Innovative Computing, Information and Control* **7**(11): 6247–6259.

Woźniak, M. and Krawczyk, B. (2012). Combined classifier based on feature space partitioning, *International Journal of Applied Mathematics and Computer Science* **22**(4): 855–866, DOI: 10.2478/v10006-012-0063-0.

Wu, O., Zuo, H., Zhu, M., Hu, W., Gao, J. and Wang, H. (2009). Rank aggregation based text feature selection, *Proceedings of the IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Tech, Milano, Italy*, Vol. 1, pp. 165–172.

Xinjie, Y. and Mitsuo, G. (2010). *Introduction to Evolutionary Algorithms*, Springer-Verlag, London.

Yan, F., Dridi, M. and Moudni, A.E. (2013). An autonomous vehicle sequencing problem at intersections: A genetic algorithm approach, *International Journal of Applied Mathematics and Computer Science* **23**(1): 183–200, DOI: 10.2478/amcs-2013-0015.

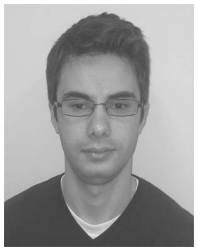
Yang, W., Li, D. and Zhu, L. (2011). An improved genetic algorithm for optimal feature subset selection from multi-character feature set, *Expert Systems with Applications* **38**(3): 2733–2740.

Yusof, R., Khairuddin, U. and Khalid, M. (2012). A new mutation operation for faster convergence in genetic algorithm feature selection, *International Journal of Innovative Computing, Information and Control* **8**(10(B)): 7363–7379.

Zhang, J., Avasarala, V., Sanderson, A.C. and Mullen, T. (2008). Differential evolution for discrete optimization: An experimental study on combinatorial auction problems, *Proceedings of the IEEE Congress on Evolutionary Computation 2008, Hong Kong, China*, pp. 2794–2800.

Zhua, M., Chena, W., Hirdes, J.P. and Stolee, P. (2007). The k-nearest neighbor algorithm predicted rehabilitation potential better than current clinical assessment protocol, *Journal of Clinical Epidemiology* **60**(10): 1015–1021.

Goran Martinović is a full professor of computer science. He obtained his B.Sc.E.E. degree from the Faculty of Electrical Engineering, J.J. Strossmayer University of Osijek, in 1996. In 2000 and 2004, he obtained his M.Sc. and Ph.D. degrees in computer science, both from the Faculty of Electrical Engineering and Computing, University of Zagreb. His research interests include distributed computer systems, fault-tolerant systems, real-time systems, artificial intelligence and medical informatics. He is a member of the IEEE, ACM, IACIS, Cognitive Science Society, KOREMA and IEEE SMC Technical Committee on Distributed Intelligent Systems.



Dražen Bajer received the Bachelor and Master degrees in computer engineering from the Faculty of Electrical Engineering, J.J. Strossmayer University of Osijek in 2008 and 2010, respectively. He is currently pursuing the Ph.D. degree at the Faculty of Electrical Engineering. His research interests are computational intelligence methods and their applications, and unsupervised classification. He is an IEEE graduate student member.

Bruno Zorić received the Bachelor and Master degrees in computer engineering from the Faculty of Electrical Engineering, J.J. Strossmayer University of Osijek in 2008 and 2011, respectively. He is currently pursuing the Ph.D. degree at the Faculty of Electrical Engineering. His research interests are supervised classification and affective computing. He is an IEEE graduate student member.

Received: 6 January 2013
Revised: 20 August 2013
Re-revised: 27 November 2013