Comparison of Multi-objective Evolutionary Algorithms for Prototype Selection in Nearest Neighbor Classification

Giovanni Acampora
Department of Physics “Ettore Pancini”
University of Naples Federico II
80126 Naples, Italy
Email: giovanni.acampora@unina.it

Genoveffa Tortora and Autilia Vitiello
Department of Computer Science
University of Salerno
84084 Fisciano, Italy
Email: {tortora, avitiello}@unisa.it

***Abstract*—The nearest neighbor classifiers are popular supervised classifiers due to their ease of use and good performance. However, in spite of their success, they suffer from some defects such as high storage requirements, high computational complexity, and low noise tolerance. In order to address these drawbacks, prototype selection has been studied as a technique to reduce the size of training datasets without deprecating the classification accuracy. Due to the need of achieving a trade-off between accuracy and reduction, Multi-Objective Evolutionary Algorithms (MOEAs) are emerging as methods efficient in solving the prototype selection problem. The goal of this paper is to perform a systematic comparison among well-known MOEAs in order to study their effects in solving this problem. The comparison involves the study of MOEAs' performance in terms of the well-known measures such as hypervolume, Δ index and coverage of two sets. The empirical analysis of the experimental results is validated through a statistical multiple comparison procedure.**

I. Introduction

The nearest neighbor classifiers are a class of lazy learners since they perform classification by using the closest training examples in the feature space. As a consequence, they are very simple because of the lack of a training phase before classifying new incoming data. The simplest version of the nearest neighbor classifiers is 1-NN that gets one only nearest neighbor in the training set and assigns its class label to the new sample to be classified. Although 1-NN is an easy technique, it has established itself as one of the most important and effective methods in Data Mining (DM) and pattern recognition [1].

However, in spite of the success of the nearest neighbor classifiers, they are characterized by some drawbacks [2], out of which three negatively impact on their application [3]: (1) the need for high storage demand in order to contain the training set necessary to execute the decision rule; (2) the need of multiple computations of distances between the new sample and training samples during the computation of the decision rule that leads to the low efficiency; (3) low tolerance to noise caused by considering all data as significant, even when the training set may contain wrong data. In order to address these drawbacks, a technique named *Prototype Selection* (PS) has been suggested and studied. This technique aims to reduce the original training set to a prototype representative set with a lower dimension that allows to achieve a similar or even higher classification accuracy. Thanks to this capability, the Prototype Selection Methods (PSMs) can speed up the nearest neighbor classification runtime with no loss of accuracy. Hence, our interest in PS in this work.

Among the literature methods for PS, approaches based on evolutionary optimization have shown the best performance in last years [4]. In particular, the objectives of a PSM are: (1) maximize the number of the instances correctly classified by a nearest neighbor classifier when using the prototypes and (2) maximize the amount of reduction achieved with respect to the original training set. PSMs that provide the best trade-off between both objectives are the best performers. Multi-Objective Evolutionary Algorithms (MOEAs), therefore, are emerging as an innovative and efficient methodology to face the PS problem [5].

However, no existing works perform a formal performance evaluation to establish the efficiency of different MOEAs in handling this problem. Consequently, in this paper, a systematic comparison among well-known MOEAs used to optimize simultaneously accuracy and reduction is carried out in terms of well-known performance indices such as hypervolume, Δ index and coverage of two sets. The compared MOEAs are an improved version of the Non-dominated Sorting Genetic Algorithm [6] (NSGA-II), an improved version of the Strength Pareto Evolutionary Algorithm [7] (SPEA2), an improved version of the Pareto Envelope based Selection Algorithm [8] (PESA-II), the Fast Pareto Genetic Algorithm [9] (FastPGA), the Pareto Archived Evolution Strategy [10] (PAES) and the Duplicate Elimination Non-dominated Sorting Evolutionary Algorithm [11] (DENSEA). The experiments involve well-known datasets taken from the UCI Machine Learning Database Repository [15] characterized by different features and belonging to several application domains. The comparison is performed through a statistical multiple comparison procedure involving the Friedman's test [12] followed by the Finner's method [13].

II. Prototype selection in Neighbour Nearest Classification

Prototype Selection (PS) is an instance reduction technique which expects to produce training sets having the best classification accuracy by means of a nearest neighbor classifier [3]. In detail, its aim is to detect the smallest set of training samples which predicts the class label of a new sample with the same accuracy provided by the original training set [5]. This is a useful technique since it allows to reduce the computational cost of the classification, improving generalization capability thanks to the elimination of noise.

Formally, let TR be the original training set composed of n instances. Each instance I_i is a pair (x_i, y_i) with $i = 1, \ldots, n$, where x_i refers to an input vector of attributes and y_i refers to the related class label. Each input vector contains m input attributes that are quantitative or qualitative information that describe the corresponding instance. Any Prototype Selection Method (PSM) produces a set of prototypes $S \subseteq TR$. This set S is used to classify new data through the nearest neighbor classifier.

The PS problem can be formulated as an multi-objective optimization problem. In general, a multi-objective optimization problem is defined as a vector function f that maps a set of m decision variables to a set of n objectives. Formally (by considering a maximization problem):

$$\max \mathbf{y} = f(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_n(\mathbf{x}))$$

$$\text{subject to } \mathbf{x} = (x_1, x_2, \ldots, x_m) \in X$$

$$\mathbf{y} = (y_1, y_2, \ldots, y_n) \in Y$$

where $\mathbf{x}$ is called the *decision vector*, X is the *parameter space*, $\mathbf{y}$ is the *objective vector* and Y is the *objective space*. The set of all decision vectors for which the related objective vectors cannot be improved in any dimension without degrading in another one is denoted as *Pareto-optimal front*. Mathematically, the definition of Pareto optimality can be given as follows. Let us consider a problem to be maximized and two decision vectors $\mathbf{c}, \mathbf{d} \in X$. Then, $\mathbf{c}$ is said to dominate $\mathbf{d}$ (also written as $\mathbf{c} \prec \mathbf{d}$) iff

$$\forall i \in \{1, 2, \ldots, n\} : f_i(\mathbf{c}) \geq f_i(\mathbf{d}) \wedge$$

$$\exists j \in \{1, 2, \ldots, n\} : f_j(\mathbf{c}) > f_j(\mathbf{d})$$

Additionally, $\mathbf{c}$ is said weakly dominate $\mathbf{d}$ (also written as $\mathbf{c} \preceq \mathbf{d}$) iff $\mathbf{c} \prec \mathbf{d}$ or $f(\mathbf{c}) = f(\mathbf{d})$. All decision vectors which are not dominated by any other decision vector of a given set are known as *non-dominated* with respect to this set.

By specializing the aforementioned multi-objective problem principles, it is possible to formulate the PS as a multi-objective optimization problem as in Definition 1.

Definition 1 (PS problem): Let us consider TR be the original training set and A_{TR} be the set of all possible subsets S of TR, the PS problem can be formulated as below:

$$\max \mathbf{y} = F(S) = [f_1(S), f_2(S)] \text{ with } S \in A_{TR} \tag{1}$$

where $f_1 : A_{TR} \rightarrow \mathbb{R}$ is the first objective defined as follows:

$$f_1(S) = acc(TR, S) \text{ with } S \in A_{TR} \tag{2}$$

where acc is a function that computes the classification accuracy in percentage obtained by 1-NN classifier by considering TR as testing set and S as the training set and $f_2 : A_{TR} \rightarrow \mathbb{R}$ is the second objective defined as follows:

$$f_2(S) = red(TR, S) = \frac{|TR| - |S|}{|TR|} \cdot 100 \text{ with } S \in A_{TR} \tag{3}$$

where red is a function that computes the reduction rate of the set S with respect to the original set TR.

Once the PS problem has been defined as multi-objective optimization problem, in the next section, the discussion about how multi-objective optimization algorithms can solve it is given.

III. Multi-objective evolutionary algorithms for prototype selection problem

Over years, a great number of Multi-Objective Evolutionary Algorithms (MOEAs) has been introduced. The primary reason for this development is their capability to produce multiple Pareto-optimal solutions in a single run. In fact, typically, a problem has a multi-objective formulation because it is not possible to have a single solution which simultaneously optimizes all objectives. Hence, an algorithm that produces a large number of alternative solutions lying on or near the Pareto-optimal front is of great practical value [14]. In order to apply MOEAs to address the prototype selection problem, it is required to answer the following designer questions:

- which is the solution encoding for the prototype selection problem?
- what and how much objectives are considered to evaluate candidate solutions, i.e., a reduced training set?

Hereafter, before discussing some MOEAs existing in literature, these issues will be addressed as an indispensable step for implementing MOEAs for prototype selection problem.

A. The solution encoding

As described in Section II, the solution to the prototype selection problem consists into identifying the smallest set of training instances which predicts the class label of a new instance with the same accuracy provided by the original training set TR. Hence, a solution of the prototype selection problem should represent a subset of TR. In this work, this is achieved by means of a binary encoding. Precisely, a solution is represented as a vector of n variables (one for each sample in the training set TR) where each variable can be set to two possible states: 0 and 1. In detail, if the variable is set to 1, the corresponding training instance is included in the subset of TR, otherwise, it is not included.

B. Objectives

In this work, the quality of a solution is computed by considering two objectives involving the set S of training instances obtained starting from the solution at issue:

- the number of the instances correctly classified using the 1-NN classifier trained with the set S (see Eq. 2);
- the reduction rate of the set S with respect to the original set TR (see Eq. 3).

Since natively MOEAs deal with minimization problem whereas the aforementioned objectives are to be maximized, the two objectives taken into account are $\frac{1}{f1}$ and $\frac{1}{f2}$where $f1$ and $f2$ are the functions defined respectively in Eqs. 2 and 3.

C. Six existing multi-objective evolutionary algorithms

This section presents the compared MOEAs in our study. We consider six state-of-the-art approaches:

1) NSGA-II: it was developed to enhance NSGA proposed by Deb et al. [15]. The main characteristics of this algorithm are (1) the exploitation of elitism and (2) the crowded tournament selection. In detail, crowded tournament selection is a mechanism based on tournament selection, therefore, it consists in a tournament where a group of individuals takes part and whose the winner depends on the fitness levels that each individual brings to the tournament [16]. In NSGA-II, the fittest individuals are identified by a ranking mechanism composed of two phases. The first phase pulls out tiers of non-dominated fronts, and sets solutions in earlier fronts as better. The second phase calculates the so-called *crowding distance* to establish how much the solutions of the nearest neighbors are close (larger distances are better). At each iteration, the best solutions with respect to these two measures are retained as the next population, and genetic operators are executed to create a new child population.

2) SPEA2: it is an improved version of that originally proposed in [17]. It returns a set of non-dominated solutions stored externally in a second, continuously updated, population named *archive*. At each generation, SPEA2 computes the fittest individuals within the union of archive and child populations according to fitness values computed through a mechanism composed of two steps: (1) computation of a raw fitness value based on how many solutions the target solution dominates; (2) a density estimation based on target solution proximity to other solutions in the objective space [16]. The computed fittest solutions are saved as the next population, and genetic operators are executed to create a new child population. Precisely, as for the selection operator, SPEA2 uses the binary tournament like NSGA-II.

3) PESA-II: it uses an archive as in SPEA2 but archive size is not fixed. Besides, the archive only allows non-dominated solutions to be members. If the archive size is superior to the number of solutions in a population, a *squeeze factor* is calculated for all members of the archive and used to reduce the size of the archive (the solutions with the highest squeeze factor are removed) [16]. PESA-II applies genetic operators only to archive members to form a new child population.

4) FastPGA: it is a population-based evolutionary algorithm that includes a new fitness assignment and ranking strategy. The novel ranking strategy is based on separating the candidate solutions into two different categories according to solution dominance. The non-dominated solutions belong to the first category. The fitness of these solutions is computed by means of the crowding distance approach suggested by Deb et al. [6]. Instead, the dominated solutions which belong to the second category are compared to all other solutions to compute their fitness values. In detail, the fitness value is depending on the number of solutions it dominates. The fast propagation of the Pareto optimal solution set is ensured through an elitism operator. A population regulation operator is used to dynamically adjust the population size until achieving a user-specified maximum population size (representing the size of the set of non-dominated solutions).

5) PAES: it is identified as being a $(1 + 1)$ evolution strategy. In detail, it includes three components: the candidate solution generator, the candidate solution acceptance function, and the Non Dominated-Solutions (NDS) archive. The candidate solution generator is analogous to a random mutation hill climbing. In detail, it retains a single current solution, and at each iteration, it gives in output a new candidate one by means of a random mutation. As the objective of multi-objective search is to produce a spread of non-dominated solutions, an NDS-list is used to explicitly maintain a limited number of these as and when they are found by the hill climber. The adaptive archiving algorithm presented in [18] is used in this paper. The design of the acceptance function is obvious in the case of the mutant dominating the current solution or vice versa. In the non-dominated case, a comparison set is used to help decide between the mutant and the current solution as described in [19].

6) DENSEA: it is based on the non-domination sorting criterion selection and it offers population diversity maintenance based on deletion and replacement of duplicate solution. In detail, the implemented deletion operator for duplicate solutions works as follows: the algorithm deletes the accumulated duplicate solutions due to the reduced non-dominated solutions quantity and replaces each deleted solution by inserting the individual that has the same ordering in the second half of the population until the completion of 50% of the population size [16]. Hence, the inclusion of diverse solutions replacing duplicates is fostered, helping to maintain the population diversity.

IV. Experiments and results

In this section, we study the effects of the six considered multi-objective evolutionary methods (NSGA-II, SPEA2, PESA-II, FastPGA, PAES and DENSEA) on solving the prototype selection problem. In the performed experiments, each compared algorithm ends after 5000 evaluations of fitness and runs by using the following parameters: population size equals to 50 and crossover and mutation probabilities equal to, respectively, 0.9 an 0.025. As for the genetic operators, all MOEAs execute the Single Point Crossover and the Bit Flip

Mutation. Regarding the selection mechanism, each MOEA uses its own selection operator (see section III-C).

Hereafter, more details about the performed comparison are given after a description of the exploited datasets and performance metrics.

A. Datasets

In all our experiments, we have exploited ten well-known datasets belonging to the UCI Machine Learning Database Repository [20]. The properties including the number of instances, the number of the features and the number of classes of the exploited datasets are reported in Table I. The selected datasets have different characteristics in order to execute a comparison independent from the datasets features and application domains.

TABLE I
DATASET DESCRIPTIONS

Name	*N. instances*	*N. features*	*N. classes*
appendicitis	106	7	2
automobile	205	25	6
balance	625	4	3
bupa	345	6	2
glass	214	9	7
hepatitis	155	19	2
led7Digit	500	7	10
mammographic	961	5	2
monk-2	432	6	2
pima	768	8	2

B. Performance metrics

According to [21], the quality of a non-dominated front obtained by a MOEA can be assessed by taking into account the following three aspects: (1) to minimize the distance between the produced non-dominated front and the Pareto-optimal front; (2) to obtain a good distribution of the solutions; (3) to maximize the spread of the produced non-dominated front. Since these three features cannot be assessed opportunely by using an only measure, over years, many metrics have been developed for evaluating the quality of the solution sets produced by MOEAs. In particular, Okabe et al. present in [22] a survey of all existing Performance Indices (PIs) categorized in three groups: cardinality-based PIs, accuracy PIs and distribution and spread PIs. Starting from this survey, we evaluate the performance of the compared MOEAs on the prototype selection problem by using the most used metrics for each one of the defined groups [16]. A description of these performance metrics is reported below:

- Hypervolume [17] (accuracy PI): this metric takes into consideration the dimension of the volume dominated in the objective space. In the two-dimensional (2-D) case, this metric is mathematically described as follows:

$$H = \{\sum_i S_i | x_i \in P\}$$

where P is the non-dominated front under evaluation and S_i is the area dominated by the solution x_i. The areas S_i are computed with regard to a reference point which typically is assumed to be composed of the maximum value for each objective.

- Δ index [6] (distribution and spread PI): this metric is based on distance and includes information about both spread and distribution. The Δ index is computed by the following formula:

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{|P|-1} |d_i - \overline{d}|}{d_f + d_l + (|P|-1) \cdot \overline{d}}$$

where P is the front to be evaluated, d_f and d_l are the Euclidean distances between the extreme solutions and the boundary solutions of P, $\overline{d}$ is the average of all distances d_i, $i \in [1, |P|-1]$, representing the Euclidean distance between consecutive solutions. It is worth noting that for the most widely and uniformly spreadout set of nondominated solutions, the numerator would be zero, making the metric to take a value zero [6].

- Coverage of two sets [17] (binary cardinality-based PI): this metric is a binary one because it is computed by considering two fronts to be compared one against the other. It is also referred as C metric and it is obtained by the following formula:

$$C(D,E) = \frac{|\{y \in E : \exists x \in D \text{ t.c. } x \preceq y\}|}{|E|}$$

where D and E are the two fronts to be compared and $\preceq$ represents the weak dominance relation. The function $C(\cdot,\cdot)$ associates the ordered pair (D,E) to the [0,1] interval. In detail, the value $C(D,E)$ equal to 1 indicates that all the solutions in E are dominated by the front D. As opposite, $C(D,E)$ equal to 0 indicates that none of the solutions in E is dominated by the front D. It is worth mentioning that $C(D,E)$ does not have to be equal to $1 - C(D,E)$. Thus, both $C(D,E)$ and $C(D,E)$ are computed for evaluation. In particular, we consider that D outperforms E if $C(D,E) > C(D,E)$.

C. Comparison among the considered MOEAs

This section is devoted to compare the considered MOEAs to identify the best performer for the prototype selection problem. The experiments consist of running each algorithm 20 times on each one of the considered datasets and storing the resulting non-dominated set as the outcome of each run. The comparison is carried out by using the aforementioned metrics of performance: hypervolume, Δ index and coverage of two sets (C).

Figs. 1, 2, 3 show, respectively, the hypervolume, the Δ index and C values for each algorithm and for each dataset by using the *box plot diagrams*. In detail, the upper and lower boundaries of the box are the upper and lower quartiles, while a thick line within the box represents the median. Dashed appendages summarize the spread and shape of the distribution whereas crosses are outside values.

By analysing Figs. 1, 2, 3, it possible to carry out the following discussion. As for the hypervolume metric, PAES,

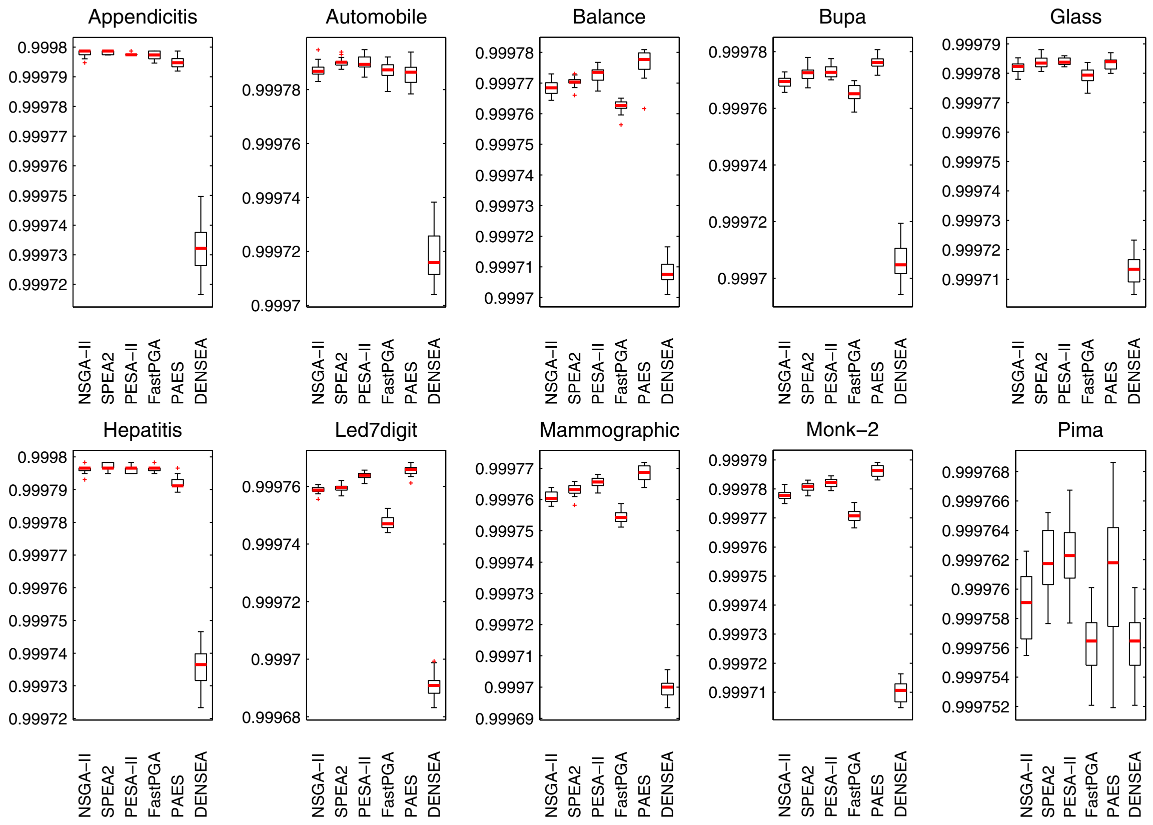

Fig. 1. Box plots for the hypervolume metric. Each rectangle shows six box plots reporting the distribution of the hypervolume values for each algorithm for a specific dataset.

SPEA2 and PESA-II seem to provide the best performance among all compared multi-objective algorithms. Indeed, PAES obtains a value for the median of the hypervolume greater than the other five EAs on six out of ten datasets (Balance, Bupa, Glass, Led7Digit, Mammographic, Monk-2). SPEA2 obtains a value for the median of the hypervolume greater than the other five EAs on three out of ten datasets (Appendicitis, Automobile, Hepatitis). Only for he dataset Pima, the best hypervolume value is obtained by PESA-II, but this algorithm results the second best performer on seven out of remaining nine datasets (Automobile, Balance, Bupa, Glass, Led7digit, Mammographic, Monk-2). As for the Δ index values, SPEA2 seems be the best performer. Indeed, it outperforms the other EAs in terms of the median of the Δ index values on eight out of ten datasets (Appendicitis, Automobile, Bupa, Hepatitis, Led7Digit, Mammographic, Monk-2, Pima). As for the dataset Balance, the best Δ index value is obtained by NSGA-II, whereas FastPGA achieves the best Δ index value on the dataset Glass. Finally, as for the C metric, PAES seems to be the best performer followed by the PESA-II. Indeed, by considering the median value, PAES covers 100% of the fronts computed by DENSEA in eight out of ten datasets (all datasets except for Appendicitis and Hepatitis), and, on the same eight datasets, it covers more than 75% of the fronts generated by NSGAII and FastPGA, more than 70% of the fronts generated by SPEA2 and more than 60% of the fronts generated by PESA-II. In turn, PESA-II covers 100% of the fronts generated by DENSEA on all datasets, and, on the eight out of ten datasets (all datasets except for Appendicitis and Hepatitis), it covers more than 85% of the fronts produced by NSGAII and FastPGA, more than 75% of the fronts produced by SPEA2 but only little more than 8% of the fronts produced by PAES. The worst performer is DENSEA that covers 0% of the outcomes of the other compared algorithms in all datasets.

In order to examine the existence of a statistical significance in the performance of the considered algorithms, we apply a statistical multiple comparison procedure for each metric. In general, a multiple statistical comparison procedure is composed of two steps [23]: in the first one, a statistical test is performed to establish if the results provided by the compared algorithms present a difference; in the second one, a post-hoc method is performed in order to establish which algorithm better outperforms. In particular, we use Friedman's test in the first step and Finner's method as post-hoc procedure since they are among the most used statistical procedures [24].

Shortly, Friedman's test is a non-parametric statistical procedure which ranks the algorithms under comparison for each data set by giving rank 1 to the best performing algorithm, rank 2 to the second one and so on. Similar to other multiple comparison procedure, Friedman's test states that all

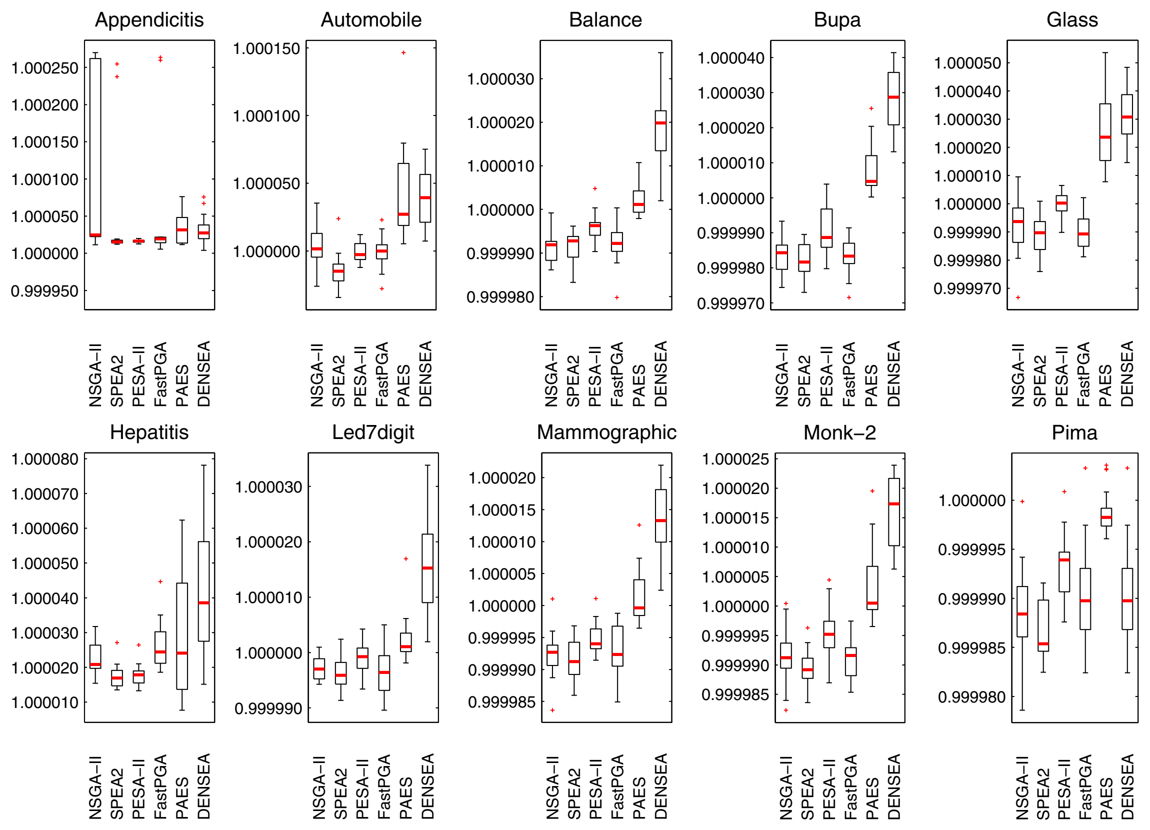

Fig. 2. Box plots for the Δ index metric. Each rectangle shows six box plots reporting the distribution of the Δ index values for each algorithm for a specific dataset.

algorithms are equivalent (null-hypothesis), hence, rejecting this hypothesis leads to the existence of differences among at least two of the compared algorithms [23]. In order to reject the null hypothesis, the Friedman's test statistic χ^2_α obtained by considering the aforementioned ranks has to be equal to or greater than the critical chi-square value at the specified level of significance [25]. As for Finner's test, it is a multiple comparison procedure that sets a control algorithm and compares this with the remaining ones. Normally, the control algorithm is the method which has the lowest value of ranking in the Friedman's test. Precisely, Finner's test considers a family of hypotheses. Each one of them is related to a comparison between the control method and one of the remaining algorithms. In detail, Finner's method orders the null hypotheses by means of the p-values and, then, it checks them sequentially. If the p_i-value is below the corresponding $1 - (1 - \alpha)^{(i/(k-1))}$, the null hypothesis is rejected and the next p_i-value is compared. When one achieves a null hypothesis that cannot be rejected, all the resting hypotheses are considered accepted [24].

The samples used to perform statistical tests are composed of the medians of the values for each dataset in the case of the hypervolume and Δ index metrics. Instead, as for the C metric, the sample of each algorithm is composed of a set of integer values, each one of them represents the number of the outperformed algorithms for a dataset. Table II shows

TABLE II
FRIEDMAN'S TEST RANKING FOR ALL CONSIDERED METRICS.

algorithm	hypervolume	Δ index	C
NSGA-II	3.60	2.80	4.00
SPEA2	2.40	**1.30**	2.90
PESA-II	**2.20**	3.50	1.80
FastPGA	4.55	2.65	4.90
PAES	2.30	5.10	**1.40**
DENSEA	5.95	5.65	6.00

the ranking computed for all compared MOEAs during the Friedman's tests performed for hypervolume, Δ index and C metrics. The computed Friedman's statistics are, respectively, 32.73, 37.81 and 46.06. Since they are greater than the critical value for five degrees of freedom $\chi^2_{0,1} = 9.2364$ (to be considered being six the number of compared algorithms), the null hypothesis is rejected for each metric and it is possible to state that there is a significant difference between at least two of the compared algorithms as for all considered metrics.

According to this result, a post-hoc test is performed to detect concrete differences among compared algorithms. In our experimentation, as shown in Table II, PESA-II is characterized by the lowest value of ranking for hypervolume metric, SPEA2 for Δ index metric and PAES for C metric, hence, they represent the control algorithms for the Finner's test for

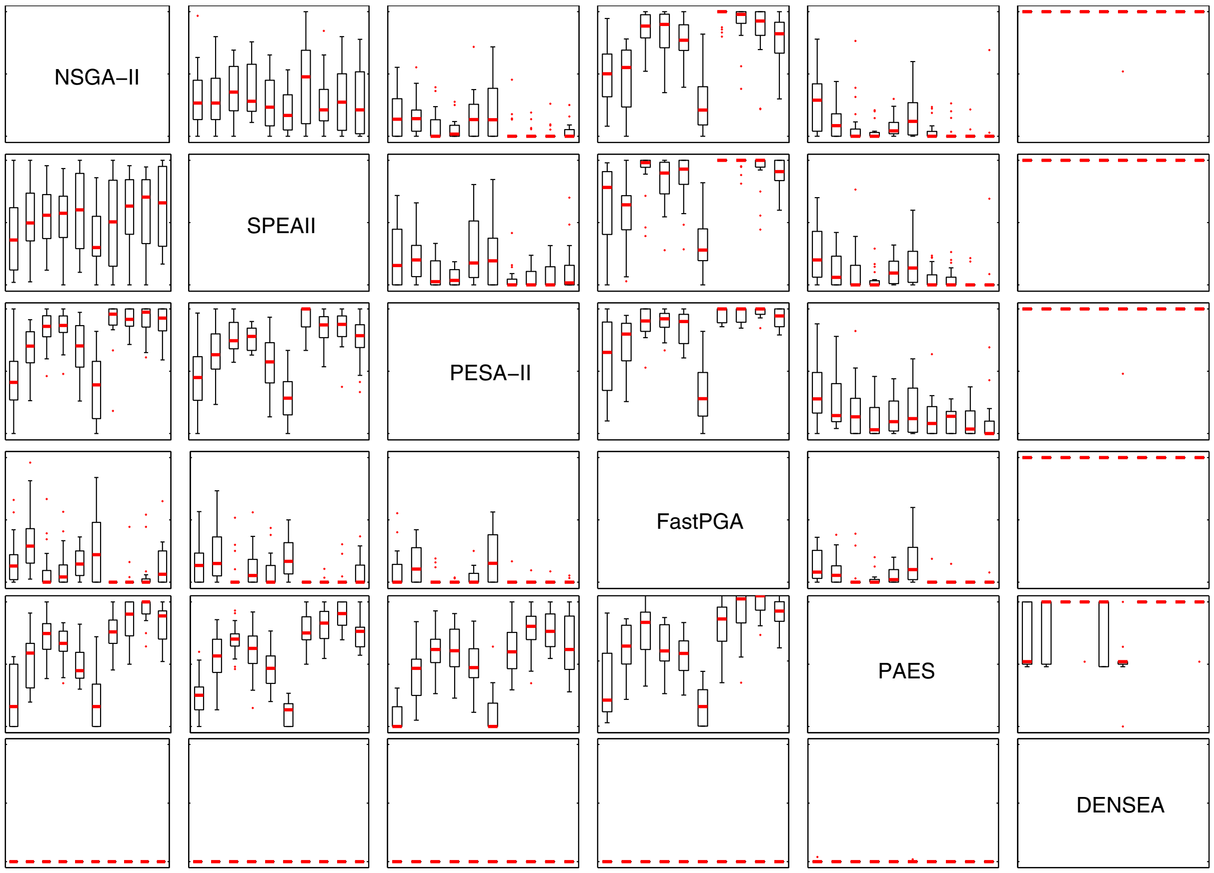

Fig. 3. Box plots based on the C metric. Each rectangle shows ten box plots reporting the distribution of the C values for a certain ordered pair of algorithms; the leftmost box plot is related to the dataset *appendicitis* and the next ones are in order as presented in Table I. The scale is 0 at the bottom and 1 at the top per rectangle. Moreover, each rectangle refers to algorithm D associated with the related row and algorithm E associated with the related column and gives the fraction of E covered by D $C(D, E)$.

the corresponding metrics. All data generated by the Finner's procedure for each metric are displayed, respectively, in Tables III, IV and V. By analyzing Table III, Finner's procedure rejects until the hypothesis 2, hence, it states that PESA-II statistically outperforms DENSEA and FastPGA, but not NSGA-II, SPEA2 and PAES at 90% confidence level (α is set to 0.1). By analyzing Table IV, Finner's procedure rejects until the hypothesis 4, hence, it states that SPEA2 statistically outperforms DENSEA, PAES, PESA-II and NSGA-II, but not FastPGA at 90% confidence level (α is set to 0.1). Finally, by analyzing Table V, Finner's procedure rejects until the hypothesis 4, hence, it states that PAES statistically outperforms DENSEA, FastPGA, NSGA-II and SPEA2 but not PESA-II at 90% confidence level (α is set to 0.1).

V. Conclusions

This paper is aimed at studying the performance of different MOEAs in solving the prototype selection problem. After performing a statistical multiple comparison procedure composed of Friedman's test and Finner's method for three well-known metrics (hypervolume, Δ index and coverage of two sets C), it is possible to conclude that there is not an algorithm that is the best performer in all metrics. The best performers for hypervolume metric are PESA-II, SPEA2, PAES and NSGA-II; the best ones for the Δ index metric are SPEA2 and FastPGA; the best ones for the C metric are PAES and PESA-

TABLE III
Finner's test for hypervolume metric. The control algorithm is PESA-II

i	algorithm	z value	unadjusted p-value	$1-(1-\alpha)^{\frac{i}{k-1}}$
5	PAES	0.1195	0.9049	0.1000
4	SPEA2	0.2390	0.8111	0.0808
3	NSGA-II	1.6733	0.0943	0.0613
2	FastPGA	2.8088	$4.9729 \cdot 10^{-3}$	0.0413
1	DENSEA	4.4821	$7.3910 \cdot 10^{-6}$	0.0209

TABLE IV
Finner's test for Δ index metric. The control algorithm is SPEA2.

i	algorithm	z value	unadjusted p-value	$1-(1-\alpha)^{\frac{i}{k-1}}$
5	FastPGA	1.6136	0.1066	0.1000
4	NSGA-II	1.7928	0.0730	0.0808
3	PESA-II	2.6295	0.0086	0.0613
2	PAES	4.5419	$5.5758 \cdot 10^{-6}$	0.0413
1	DENSEA	5.1993	$2.0010 \cdot 10^{-7}$	0.0209

TABLE V
FINNER'S TEST FOR C METRIC. THE CONTROL ALGORITHM IS PAES.

i	algorithm	z value	unadjusted p-value	$1-(1-\alpha)^{\frac{i}{k-1}}$
5	PESA-II	0.4781	0.6326	0.1000
4	SPEA2	1.7928	0.0730	0.0808
3	NSGA-II	3.1076	0.0019	0.0613
2	FastPGA	4.1833	$2.8731 \cdot 10^{-5}$	0.0413
1	DENSEA	5.4981	$3.8401 \cdot 10^{-8}$	0.0209

II. By analyzing these results, it is possible to state that PAES, PESA-II and SPEA2 (the best performers in two out of three metrics) can be good MOEAs to be applied to address the prototype selection problem.

REFERENCES

[1] G. Shakhnarovich, P. Indyk, and T. Darrell, *Nearest-neighbor methods in learning and vision: theory and practice*, 2006.

[2] I. Kononenko and M. Kukar, *Machine learning and data mining: introduction to principles and algorithms*. Horwood Publishing, 2007.

[3] I. Triguero, J. Derrac, S. Garcia, and F. Herrera, "A taxonomy and experimental study on prototype generation for nearest neighbor classification," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 42, no. 1, pp. 86–100, 2012.

[4] S. Garcia, J. Derrac, J. R. Cano, and F. Herrera, "Prototype selection for nearest neighbor classification: Taxonomy and empirical study," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 3, pp. 417–435, 2012.

[5] G. Acampora, G. Tortora, and A. Vitiello, "Applying spea2 to prototype selection for nearest neighbor classification," in *IEEE International Conference on Systems, Man, and Cybernetics*. IEEE, 2016.

[6] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[7] E. Zitzler, M. Laumanns, L. Thiele, E. Zitzler, E. Zitzler, L. Thiele, and L. Thiele, "Spea2: Improving the strength pareto evolutionary algorithm," 2001.

[8] D. W. Corne, N. R. Jerram, J. D. Knowles, M. J. Oates *et al.*, "Pesa-ii: Region-based selection in evolutionary multiobjective optimization," in *Proceedings of the genetic and evolutionary computation conference (GECCO2001*. Citeseer, 2001.

[9] H. Eskandari, C. Geiger, and G. Lamont, "Fastpga: A dynamic population sizing approach for solving expensive multiobjective optimization problems," in *Evolutionary Multi-Criterion Optimization*. Springer, 2007, pp. 141–155.

[10] J. Knowles and D. Corne, "The pareto archived evolution strategy: A new baseline algorithm for pareto multiobjective optimisation," in *Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on*, vol. 1. IEEE, 1999.

[11] D. Greiner, J. Emperador, and G. Winter, "Enhancing the multiobjective optimum design of structural trusses with evolutionary algorithms using densea," in *44th AIAA (American Institute of Aeronautics and Astronautics) Aerospace Sciences Meeting and Exhibit, paper AIAA-2006-1474*, 2006.

[12] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *Journal of the American Statistical Association*, vol. 32, no. 200, pp. 675–701, 1937.

[13] H. Finner, "On a monotonicity problem in step-down multiple test procedures," *Journal of the American Statistical Association*, vol. 88, no. 423, pp. 920–923, 1993.

[14] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii," in *International Conference on Parallel Problem Solving From Nature*. Springer, 2000, pp. 849–858.

[15] N. Srinvas and K. Deb, "Multi-objective function optimization using non-dominated sorting genetic algorithms," *Evolutionary Computation*, vol. 2, no. 3, pp. 221–248, 1994.

[16] G. Acampora, H. Ishibuchi, and A. Vitiello, "A comparison of multi-objective evolutionary algorithms for the ontology meta-matching problem," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*. IEEE, 2014, pp. 413–420.

[17] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach," *IEEE transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.

[18] J. Knowles and D. Corne, "Properties of an adaptive archiving algorithm for storing nondominated vectors," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 100–116, 2003.

[19] J. Horn, N. Nafpliotis, and D. E. Goldberg, "A niched pareto genetic algorithm for multiobjective optimization," in *Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on*. Ieee, 1994, pp. 82–87.

[20] A. Asuncion and D. Newman, "Uci machine learning repository," 2007.

[21] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evolutionary computation*, vol. 8, no. 2, pp. 173–195, 2000.

[22] T. Okabe, Y. Jin, and B. Sendhoff, "A critical survey of performance indices for multi-objective optimisation," in *Evolutionary Computation, 2003. CEC'03. The 2003 Congress on*, vol. 2. IEEE, 2003, pp. 878–885.

[23] S. García, D. Molina, M. Lozano, and F. Herrera, "A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the cec'2005 special session on real parameter optimization," *Journal of Heuristics*, vol. 15, no. 6, pp. 617–644, 2009.

[24] S. García, A. Fernández, J. Luengo, and F. Herrera, "Advanced non-parametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power," *Information Sciences*, vol. 180, no. 10, pp. 2044–2064, 2010.

[25] D. J. Sheskin, *Handbook of parametric and nonparametric statistical procedures*. crc Press, 2003.