Initial Results in Predicting High-Level Features of Constrained Multi-Objective Optimization Problems

Andrejaana Andova
Aljoša Vodopija
Jožef Stefan Institute and
Jožef Stefan International
Postgraduate School
Jamova cesta 39
Ljubljana, Slovenia
andrejaana.andova@ijs.si
aljosa.vodopija@ijs.si

Pavel Krömer
Vojtěch Uher
Department of Computer Science
VSB - Technical University of
Ostrava
17. listopadu 2172/15
Ostrava-Poruba, Czech Republic
pavel.kromer@vsb.cz
vojtech.uher@vsb.cz

Tea Tušar
Bogdan Filipič
Jožef Stefan Institute and
Jožef Stefan International
Postgraduate School
Jamova cesta 39
Ljubljana, Slovenia
tea.tusar@ijs.si
bogdan.filipic@ijs.si

ABSTRACT

Trying numerous algorithms on an optimization problem that we encounter for the first time in order to find the best-performing algorithm is time-consuming and impractical. To narrow down the number of algorithm choices, high-level features describing important problem characteristics can be related with algorithm performance. However, characterizing optimization problems for this purpose is challenging, especially when they include multiple objectives and constraints. In this work, we use machine learning (ML) to automatically predict high-level features of constrained multi-objective optimization problems (CMOPs) from low-level, exploratory landscape analysis features. The results obtained on the MW benchmark show a significant difference in classification accuracy depending on the applied evaluation approach. The poor performance of the leave-one-problem-out strategy indicates the need for further investigation of the relevance of low-level features in CMOP characterization.

KEYWORDS

constrained multi-objective optimization, exploratory landscape analysis, sampling methods, problem characterization, machine learning

1 INTRODUCTION

Predicting high-level features of constrained multi-objective optimization problems (CMOPs) is important as it can help decide which algorithm to use when faced with a new (real-world) CMOP. The structure of the objective and constraint functions are usually unknown for such problems. Moreover, the evaluation of problem solutions might be very time-consuming. In such cases, it is beneficial to know certain high-level features of the CMOP, which eases the selection of an appropriate multi-objective optimization algorithm or constraint handling technique to solve the problem efficiently.

Two frequently considered high-level features of CMOPs are the problem type and connectivity of the feasible region. The problem type characterizes whether and how the constraints change the Pareto front of the problem. As pointed out by Tanabe et al. [8], this feature is useful as it indicates whether the problem needs to be treated as constrained or unconstrained. Moreover, Ma et al. [5] showed which constraint handling techniques are more successful in solving CMOPs, depending on the problem type. Similarly, the connectivity of the feasible region (or problem connectivity for short) defines the multimodality of the problem violation landscape and, therefore, crucially affects the choice of algorithms that can solve the problem efficiently [5].

High-level features of a new problem can be predicted using automatically calculated low-level problem features. The most widely known low-level features in evolutionary optimization are the exploratory landscape analysis (ELA) features. They were initially introduced to characterize single-objective optimization problems and implemented in the flacco package [2]. More recently, Liefooghe et al. [4] proposed a set of ELA features for multi-objective optimization problems, and Vodopija et al. [10] introduced additional ELA features for CMOPs.

In this work, we use the ELA features from [4] and some from [10] to investigate whether they are useful for predicting problem type and connectivity. To the best of our knowledge, this is the first attempt to predict the high-level features of CMOPs. A similar study was performed by Renau et al. [7] on single-objective optimization problems. They used ELA features to classify the optimization problem. When splitting the data into training and test sets, instances from the same problem were used for both training and testing. The first of our three experiments follows this setup. However, because this evaluation methodology is not useful in practice (the class of a new real-world problem is unknown), a second experiment is performed using the leave-one-problem-out methodology. Finally, the third experiment varies the number of target problem instances used for training to gain further insight in the difficult task of predicting high-level features from low-level ones.

The paper is further organized as follows. In Section 2, we introduce the theoretical background of constrained multi-objective optimization. In Section 3, we explain the features used in this study. In Section 4, we present the considered test problems, and in Section 5 the experimental setup. In Section 6, we report on the obtained results. Finally, in Section 7, we provide a conclusion and present the ideas for future work.

2 THEORETICAL BACKGROUND

A CMOP can be formulated as:

$$\begin{aligned} &\text{minimize} \quad f_m(x), \quad m = 1, \ldots, M \\ &\text{subject to} \quad g_k(x) \leq 0, \quad k = 1, \ldots, K, \end{aligned} \tag{1}$$

where $x = (x_1, \ldots, x_D)$ is a *search vector* of dimension D, $f_m : S \to \mathbb{R}$ are *objective functions*, $g_k : S \to \mathbb{R}$ *constraint functions*,

Information Society 2022, 10–14 October 2022, Ljubljana, Slovenia

$S \subseteq \mathbb{R}^D$ is the *search space*, and M and K are the numbers of objectives and constraints, respectively.

A solution x is *feasible*, if it satisfies all constraints $g_k(x) \leq 0$ for $k = 1, \ldots, K$. For each constraint g_k we can define the *constraint violation* as $v_k(x) = \max(0, g_k(x))$. The *overall constraint violation* is defined as

$$v(x) = \sum_{i}^{K} v_k(x). \tag{2}$$

A solution x is feasible iff $v(x) = 0$.

A feasible solution $x \in S$ is said to *dominate* another feasible solution $y \in S$ if $f_m(x) \leq f_m(y)$ for all $1 \leq m \leq M$, and $f_m(x) < f_m(y)$ for at least one $1 \leq m \leq M$. A feasible solution $x^* \in S$ is a *Pareto-optimal solution* if there exists no feasible solution $x \in S$ that dominates x^*. All feasible solutions constitute the *feasible region*, $F = \{x \in S \mid v(x) = 0\}$, and all nondominated feasible solutions form the *Pareto set*, S_o. The image of the Pareto set in the objective space is the *Pareto front*, $P_o = \{f(x) \mid x \in S_o\}$.

3 EXPLORATORY LANDSCAPE ANALYSIS

ELA is a selection of techniques able to analyze the search and objective space of a problem, their correlation and their characteristics with the goal of identifying the features important for the performance of optimization algorithms. To extract the ELA features, one needs to first generate a sample of solutions. The ELA features use statistical methods to characterize the problem landscape. Thus, one can use an arbitrary sample size. However, the ELA features are generally more accurate for large sample sizes. The ELA features proposed by Liefooghe et al. [4] and used also in this work can be divided into four categories: global, multimodality, evolvability, and ruggedness features.

The global features capture certain global problem properties, for example, the correlation between the objective values, average and maximum distance between solutions in the search space and the objective space, the proportion of non-dominated solutions, the average and maximum rank of solutions, etc.

The multimodality features assess the number of local optima in the objective space. They are computed for the bi-objective space and also for each objective separately, in both cases by analyzing the neighbourhood of each solution. If a solution dominates its neighbors (or has a better objective value than its neighbors), it is defined as a local optimum. The multimodality features comprise the proportion of solutions that are locally optimal, the average and maximum distances between local optima, etc.

The evolvability features describe how fast a local optimizer would converge towards an optimum. They are calculated by analyzing how many neighboring solutions are dominated by, dominating, or incomparable with a given solution.

The ruggedness features measure the correlation between the information and quality from neighboring solutions – larger correlation means a smoother landscape. The features are calculated by using Spearman's correlation coefficient on the evolvability features between each pair of neighboring solutions.

In addition, we include four ELA features from [10] that describe the violation landscape and its relation with the objective space. The first feature is the feasibility ratio. It is expressed as the proportion of feasible solutions in the sample and is one of the most frequently used features in categorizing violation landscapes. The second feature is the maximum value of overall constraint violation values in the sample. The last two features measure the relationship between the objectives and constraints. They are the minimum and maximum correlations between the objectives and the overall constraint violation.

Table 1: High-level features of the MW test problems.

Problem	Type	Connectivity
MW1	II	Disconnected
MW2	I	Disconnected
MW3	III	Connected
MW4	I	Connected
MW5	II	Connected
MW6	II	Disconnected
MW7	III	Connected
MW8	II	Disconnected
MW9	IV	Connected
MW10	III	Disconnected
MW11	IV	Disconnected
MW12	IV	Disconnected
MW13	III	Disconnected
MW14	I	Connected

4 TEST PROBLEMS

We base this study on 14 CMOPs proposed by Ma et al. [5] and called MW1–14. In addition to proposing the problems, the authors also describe them with high-level features, such as the problem type and connectivity of the feasible region. The values of these two high-level features for each MW problem are listed in Table 1.

Many of the ELA features proposed by Liefooghe et al. [4] can only be calculated for bi-objective optimization problems. Therefore, we investigate only the bi-objective versions of the MW problems although three of them are scalable in the number of objectives. All MW problems are also scalable in the number of variables. We use 5-dimensional problems to match the experimental setup from [7].

5 EXPERIMENTAL SETUP

In preliminary experiments, we used six sampling methods from the ghalton [1] and scipy [9] Python libraries: gHalton, Halton, Sobol, Latin hypercube sampling, optimized Latin hypercube sampling, and uniform sampling [3]. The results have shown that similar prediction accuracies are obtained when using data provided by any of these sampling methods. For this reason, we only present the results obtained using the Sobol sampling method in the rest of the paper.

The Sobol sampling method generates a sample set by partitioning the search space and filling each partition with a sample solution. We generate additional Sobol sample sets using the Cranley-Patterson rotation [3]. The solutions from the original sample set are rotated using a random shift of each dimension, thus creating new sample sets that preserve the properties of the Sobol sampling. The modulo operation keeps the shifted values within the unitary interval. This approach was also used by Renau et al. [7].

Following this approach, we generate 100 sets of samples, each with 512 solutions, which we then evaluate on all 14 MW benchmark problems. For each problem and sample set pair, we compute 46 ELA features, which represent a single instance in the data. As a result, by evaluating the 100 sample sets on each of the 14 test problems, we get 1400 data instances. We then use

these data instances and the corresponding high-level problem features (problem type and connectivity) to train a classifier for predicting the high-level problem features.

We use two widely used machine learning (ML) methods for classification: the Random Forest (RF) classifier and the k-Nearest Neighbors (KNN) classifier. The reason for choosing these classifiers instead of some others is that, usually, RF performs favorably compared to other ML classifiers. KNN, on the other hand, uses the distance between solutions as a performance metric, which is useful when analyzing the obtained classification results visually. For both RF and KNN, we apply the implementation from the scikit-learn library [6]. For KNN, we keep the default settings, while for RF we train 100 trees.

We perform three experiments that differ in the classifier evaluation methodology. In the first experiment, we base the evaluation methodology on the work by Renau et al. [7], where the data is split by using instances from the same problem for both training and testing. There, 50% of all instances are used for training, and the remaining 50% for testing. Furthermore, we take care of dividing the instances into training and test sets so that the proportion of instances from each problem is equal in both sets.

However, this methodology does not correspond to the real-world scenario where we want to learn the high-level features of a problem encountered for the first time. Therefore, we use the leave-one-problem-out evaluation methodology in the second experiment. Here, the instances from a single problem are used for testing, and the instances from all other problems for training. The procedure is repeated for all problems and the classification accuracy is calculated as the average over all train-test splits.

Finally, the third experiment is performed to see how adding target problem data to the training set influences the resulting classification accuracy. In this experiment, we vary the percentage of target problem data that is used for training between 0% and 99% with the step of 1%. When it equals 0%, no target problem data is used for training, which corresponds to the leave-one-problem-out methodology of the second experiment. Note that this setup never equals the one from the first experiment because here the data of all other (non-target) problems is always used for training. Again, this procedure is repeated for all problems and we report the average classification accuracy.

To better understand the task we are trying to solve, we visualize the classes by first reducing the dimensionality of the feature space from 46-D to 2-D using Pairwise Controlled Manifold Approximation Projection (PaCMAP) [11]. We use the Python package pacmap with default parameter values.

6 RESULTS

The results of the first experiment, where 50% of all data is used for training and 50% for testing, show that both RF and KNN achieve a classification accuracy above 98% (see Table 2). An explanation for such good results can be derived from the two leftmost plots in Figure 1. Here, we can see that PacMAP finds many clusters in the data. However, the clusters are highly correlated to the problems themselves. Thus, leaving some instances from the target problem in the training set results in a high classification accuracy because the classification task is now transformed into identifying to which cluster the new sample belongs, which is a much easier task to perform.

The more realistic scenario of having to predict the high-level feature of a yet unseen problem is tested in the second experiment. Here, the classification accuracy drops to only 7–19% for the problem type prediction, and to 41–57% for the problem connectivity prediction (see the leftmost points corresponding to 0% on the plots in Figure 2). This is comparable to the classification accuracy of the stratified classifier, which achieves 19% for the problem type prediction and 45% for the problem connectivity prediction. We can look at the results of the third experiment to help us understand this decline in classification accuracy. As seen from Figure 2, adding just a few instances of the target problem to the training set drastically increases the classification accuracy.

Table 2: Classification accuracy when 50% of all data is used for training and 50% for testing (first experiment).

Learning method	Problem type	Problem connectivity
RF	98%	99%
KNN	100%	100%

When the training data contains no instances from the target problem, the classifier is forced to find information about the high-level feature from other problems. However, this is a much harder task given that similar problems often have different high-level features (see the middle and right plots in Figure 1).

In the visualizations in Figure 1 the points indicating the correctly classified instances have black edges. As we can see, for many problems, RF has a 0% classification accuracy (top middle and top right plot). There are, however, some problems for which RF finds the correct class for a number of instances. Nonetheless, from these 2-D plots it is hard to understand why certain instances are misclassified by RF. This is because RF detects details in the data that the dimensionality reduction visualization method is unable to capture.

Similar behavior can be observed for KNN. Given that KNN classifies an instance depending on the classes of its most similar instances, the visualization from Figure 1 can help interpret its poor results on the leave-one-problem-out methodology. We can see that the clusters created by PacMAP are not well-aligned with the high-level features of problem type and connectivity. This makes predicting them a hard task for KNN. The clustering by PacMAP suggests that the applied ELA features are not descriptive enough for predicting problem type and connectivity.

7 CONCLUSION AND FUTURE WORK

In this work, we tried to predict high-level features of CMOPs. More specifically, using low-level ELA features, we constructed the classifiers to predict the problem type and connectivity. Two ML classifiers were utilized, RF and KNN.

We employed three evaluation methodologies. The first one follows the related work and splits the data into two halves, one serving as the training set and the other as the test set (instances from the same problem are used in both sets). The second evaluation methodology uses all instances from the target problem for testing, and none for training. The third method gradually adds the target problem data to the training set. We achieved excellent classification accuracy with the first evaluation methodology, but very poor ones with the second one. The drop in classification accuracy was checked by the third methodology, which has shown that already a small number of instances of the same problem increases the classification accuracy.

Visualizations of the data in the form of 2-D plots show that CMOP instances form clusters that are highly correlated to the problem instances, but not to the high-level problem features. For

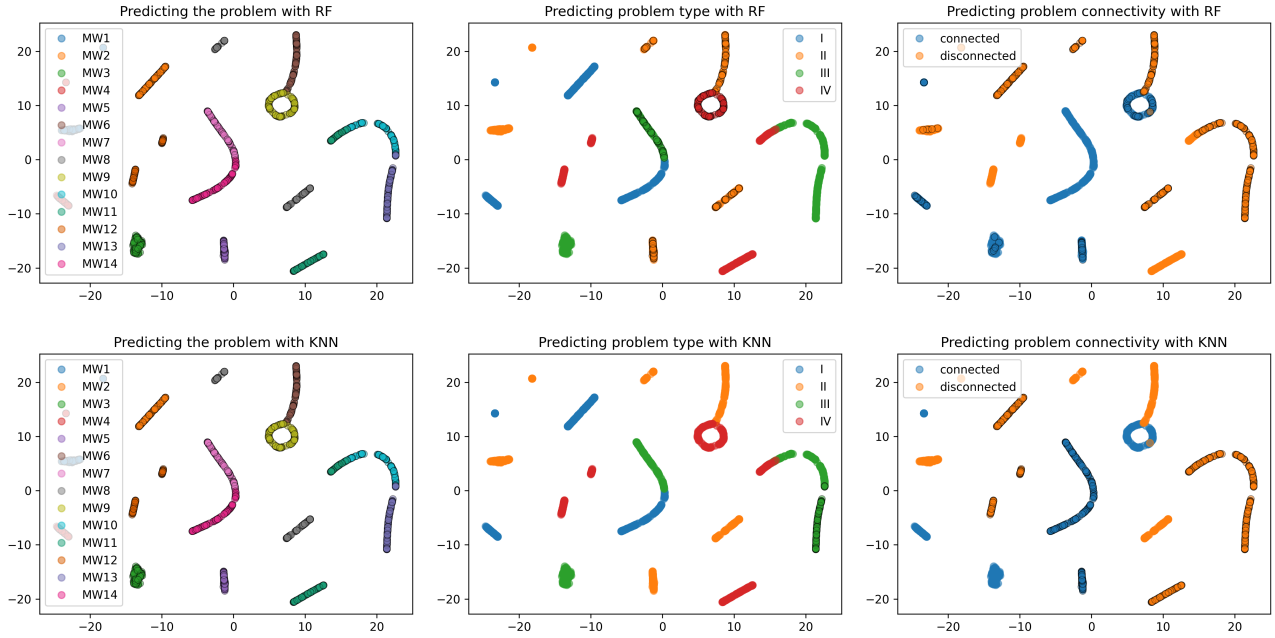

Figure 1: Dimensionality reduction of the ELA feature space using the PacMAP method. Points are colored based on their true values with correct classifications denoted by a black point edge. The top and bottom rows show the results for Random Forest and KNN, respectively, while the different classification targets are arranged in columns: the left column displays the results for the problem, the middle for problem type and the right for problem connectivity.

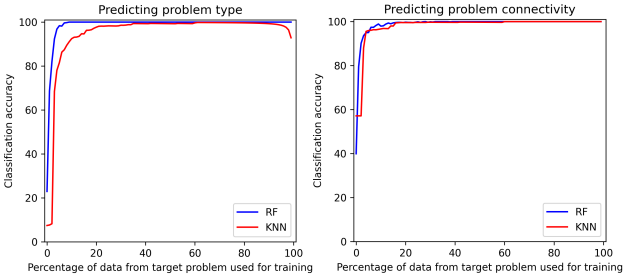

Figure 2: Classification accuracy for different proportions of data from the target problem used for training.

this reason, by including some instances from the target problem in the training set, the classification task becomes an easier task of recognizing to which cluster an instance belongs. Unfortunately, this is not a realistic scenario, since in the real world we have no information on the characteristics of the newly encountered problem. We therefore recommend to use the second evaluation methodology when addressing this task.

However, the initial results obtained using the second evaluation methodology are not so promising. A possible improvement could be considering more ELA features in the learning procedure, either additional ones from [10] or newly created ones. Moreover, using a more representative set of test problems from various benchmark suites may also improve classifier performance.

ACKNOWLEDGMENTS

The authors acknowledge the project "Constrained multi-objective Optimization Based on Problem Landscape Analysis" was financially supported by the Slovenian Research Agency (project no. N2-0254) and the Czech Science Foundation (grant no. GF22-34873K). The Slovenian authors acknowledge additional financial support from the Slovenian Research Agency (young researcher program and research core funding no. P2-0209).

REFERENCES

[1] François-Michel De Rainville, Christian Gagné, Olivier Teytaud, and Denis Laurendeau. 2012. Evolutionary optimization of low-discrepancy sequences. *ACM Transactions on Modeling and Computer Simulation*, 22, 2, 1–25.

[2] Christian Hanster and Pascal Kerschke. 2017. flaccogui: Exploratory landscape analysis for everyone. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO) Companion*. ACM, 1215–1222.

[3] Christiane Lemieux. 2009. *Monte Carlo and Quasi-Monte Carlo Sampling. Springer Series in Statistics*. Springer New York, NY.

[4] Arnaud Liefooghe, Sébastien Verel, Benjamin Lacroix, Alexandru-Ciprian Zăvoianu, and John McCall. 2021. Landscape features and automated algorithm selection for multi-objective interpolated continuous optimisation problems. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*. ACM, 421–429.

[5] Zhongwei Ma and Yong Wang. 2019. Evolutionary constrained multiobjective optimization: Test suite construction and performance comparisons. *IEEE Transactions on Evolutionary Computation*, 23, 6, 972–986.

[6] F. Pedregosa et al. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.

[7] Quentin Renau, Carola Doerr, Johann Dreo, and Benjamin Doerr. 2020. Exploratory landscape analysis is strongly sensitive to the sampling strategy. In *International Conference on Parallel Problem Solving from Nature*. Springer, 139–153.

[8] Ryoji Tanabe and Akira Oyama. 2017. A note on constrained multi-objective optimization benchmark problems. In *IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 1127–1134.

[9] Pauli Virtanen et al. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17, 261–272.

[10] Aljoša Vodopija, Tea Tušar, and Bogdan Filipič. 2022. Characterization of constrained continuous multiobjective optimization problems: A feature space perspective. *Information Sciences*, 607, 244–262.

[11] Yingfan Wang, Haiyang Huang, Cynthia Rudin, and Yaron Shaposhnik. 2021. Understanding how dimension reduction tools work: An empirical approach to deciphering t-sne, umap, trimap, and pacmap for data visualization. *Journal of Machine Learning Research*, 22, 201, 1–73.