



Universiteit
Leiden

The Netherlands

Landscape-aware automated algorithm configuration using multi-output mixed regression and classification

Long, F.X.; Frenzel, M.; Krause, P.; Gitterle, M.; Bäck, T.H.W.; Stein, N. van; ... ; Machado, P.

Citation

Long, F. X., Frenzel, M., Krause, P., Gitterle, M., Bäck, T. H. W., & Stein, N. van. (2024). Landscape-aware automated algorithm configuration using multi-output mixed regression and classification. *Lecture Notes In Computer Science*, 87-104.
doi:10.1007/978-3-031-70068-2_6

Version: Publisher's Version

License: [Creative Commons CC BY 4.0 license](https://creativecommons.org/licenses/by/4.0/)

Downloaded from: <https://hdl.handle.net/1887/4254684>

Note: To cite this publication please use the final published version (if applicable).



Landscape-Aware Automated Algorithm Configuration Using Multi-output Mixed Regression and Classification

Fu Xing Long^{1,2(✉)} , Moritz Frenzel³ , Peter Krause⁴ , Markus Gitterle⁵ ,
Thomas Bäck¹ , and Niki van Stein¹

¹ LIACS, Leiden University, Niels Bohrweg 1, 2333 Leiden, Netherlands
{f.x.long,t.h.w.baeck,n.van.stein}@liacs.leidenuniv.nl

² BMW Group, Knorrstraße 147, 80788 Munich, Germany
fu-xing.long@bmw.de

³ Altair Engineering GmbH, Calwer Straße 7, 71034 Böblingen, Germany
mfrenzel@altair.com

⁴ Divis Intelligent Solutions GmbH, Joseph-von-Fraunhofer-Straße 20, 44227
Dortmund, Germany
krause@divis-gmbh.de

⁵ Munich University of Applied Sciences, Dachauer Straße 98b, 80335 Munich,
Germany
markus.gitterle@hm.edu

Abstract. In landscape-aware algorithm selection problem, the effectiveness of feature-based predictive models strongly depends on the representativeness of training data for practical applications. In this work, we investigate the potential of randomly generated functions (RGF) for the model training, which cover a much more diverse set of optimization problem classes compared to the widely-used black-box optimization benchmarking (BBOB) suite. Correspondingly, we focus on automated algorithm configuration (AAC), that is, selecting the best suited algorithm and fine-tuning its hyperparameters based on the landscape features of problem instances. Precisely, we analyze the performance of dense neural network (NN) models in handling the multi-output mixed regression and classification tasks using different training data sets, such as RGF and many-affine BBOB (MA-BBOB) functions. Based on our results on the BBOB functions in $5d$ and $20d$, near optimal configurations can be identified using the proposed approach, which can most of the time outperform the off-the-shelf default configuration considered by practitioners with limited knowledge about AAC. Furthermore, the predicted configurations are competitive against the single best solver in many cases. Overall, configurations with better performance can be best identified by using NN models trained on a combination of RGF and MA-BBOB functions.

Keywords: Black-box optimization · Exploratory landscape analysis · Multi-output mixed regression and classification · Dense neural network · Randomly generated functions

1 Introduction

In landscape-aware algorithm selection problem (ASP) [25,33], the performance of optimization algorithms has been linked to the landscape characteristics of black-box optimization (BBO) problems that are quantified using fitness landscape analysis [20]. By constructing machine learning (ML) models, for instance, the performance of optimization algorithms can be estimated based on the landscape characteristics of problem instances [11]. In other words, the problem landscape characteristics can be exploited to select well-performing optimization algorithms from an algorithm portfolio prior to the actual optimization runs. Using a large set of problem instances, and preferably from diverse optimization problem classes, the corresponding landscape characteristics and algorithm performances are utilized for the training of ML models. Following this, the effectiveness of predictive models is heavily dependent on the representativeness of training data for unseen BBO problems.

Although landscape features are informative in explaining algorithm performances [31], landscape-aware ASP was mainly investigated on benchmarking problems in previous work, such as the widely-used black-box optimization benchmarking (BBOB) suite [7]. The fact that the BBOB suite is not representative enough for engineering applications, such as crashworthiness optimization [14,15] and control system calibration [34] in the automotive industry, raises concerns that predictive models trained using only the BBOB suite might generalize poorly to unseen problem classes that are not being covered. Moreover, for real-world BBO problems that require expensive function evaluations, e.g., time-consuming and/or costly simulation runs, the function evaluation budget can be particularly limited, hindering the generation of a large data set for the model training. To fill in the gap, we explore an alternative in building pre-trained general purpose models that can generalize well for different applications, including expensive BBO problems, while maintaining an affordable computational effort. Our ultimate vision is to assist practitioners with little domain knowledge about ASP, e.g., from engineering fields, to automatically identify the best suited algorithm configuration for their applications.

Our Contribution: In this work, we investigate the potential of tree-based randomly generated functions (RGF) for the training of predictive models, which are much more diverse than the BBOB suite in terms of optimization landscape characteristics. In this context, we implement a selection process to identify RGF that are appropriate as training data. Furthermore, we extend our investigations towards landscape-aware automated algorithm configuration (AAC) by combining both algorithm selection and hyperparameter optimization (HPO), that is, finding the best suited algorithm and fine-tuning its hyperparameters. For the prediction of optimal configurations, we consider dense neural network (NN) models, which can easily handle multi-output mixed regression and classification tasks. Based on our empirical results, near optimal configurations can be identified using the proposed approach, which can outperform the off-the-shelf default configuration and compete against the single best solver (SBS) for many

BBOB functions. In some cases, NN models can perform better than random forest (RF) models, which are typically considered for landscape-aware ASP.

This paper has the following structure: Related works are introduced in Sect. 2, followed by the explanations of our methodology in Sect. 3 and experimental setup in Sect. 4. Next, results are analyzed and discussed in Sect. 5. Lastly, conclusions and future works are presented in Sect. 6.

2 Related Work

The idea of using RGF for the training of feature-based predictive models has been previously investigated, such as in [42]. In summary, it was reported that RGF were ineffective for the training of high-quality models in terms of prediction accuracy. Independently of the previous work, our work differs mainly in the following extensions.

1. Instead of simply using any RGF, we implement an intermediate step to select RGF that are appropriate for the model training purposes. We argue that this step is crucial to improve model accuracy, as discussed in Sect. 3.1. In fact, we suspect that this might partly explain the low model accuracy in [42].
2. We propose to consider NN-based predictive models to handle the multi-output mixed regression and classifications tasks in AAC, which can sometimes perform slightly better than RF models, refer to Sect. 3.2.
3. Rather than just selecting the best algorithm from a portfolio of limited algorithms, as typically done in ASP, we extend our investigations towards combined algorithm selection and hyperparameter optimization (CASH) [35], or we call AAC [33] in this work.

2.1 Automated Algorithm Configuration

To tackle AAC problems, where the search space can be a mix of continuous, integer, categorical, and conditional variables, various optimization algorithms have been implemented, such as tree-structured Parzen estimator (TPE) [1] and sequential model-based algorithm configuration (SMAC) [13]. As a variant of Bayesian optimization [23], TPE utilizes Parzen estimators as surrogate models, which can handle mixed-integer search space and scale well to high dimensionality. For example, TPE has been previously applied to fine-tune the learning rates of covariance matrix adaptation evolutionary strategy (CMA-ES) [41].

In this work, we focus on fine-tuning the configuration of modular CMA-ES [26], developed based on the original CMA-ES algorithm [8, 9]. In short, different variants, such as active learning, mirrored sampling, threshold convergence, and recombination weights, are integrated as modules that can be individually activated or deactivated, allowing a custom instantiation of CMA-ES. Subsequently, modular CMA-ES offers a convenient examination of the interactions between different modules as well as between modules and hyperparameters, e.g., population size and learning rates.

2.2 Black-Box Optimization Benchmarking

In previous work, landscape-aware ASP was commonly investigated based on BBO benchmarking suites, such as the well-known BBOB suite [7] available in the comparing continuous optimizers (COCO) platform [6] and iterative optimization heuristics profiler tool (IOHProfiler) [4]. Altogether, the BBOB suite consists of 24 single-objective, continuous, and noiseless functions of different optimization landscape complexity, which we refer to this suite as *the* BBOB.

Principally, the BBOB functions can be scaled up to arbitrary dimensionality and different problem instances can be created through transformations of the search space and objective values, which is controlled by an internal identifier. Typically, investigations based on the BBOB suite are carried out within the box-constrained search space $[-5, 5]^d$, where the global optimum is located within $[-4, 4]^d$, where d represents the dimensionality. Extensive analysis of the BBOB problem instances is available in [17].

To complement the diversity of the BBOB suite, additional functions can be generated via affine combination of two BBOB functions [3], which is based on an interpolation between two selected BBOB functions and uses a weighting factor to control the shifting between functions. This approach was later generalized to affine combinations of many BBOB functions, also known as many-affine BBOB (MA-BBOB) functions, where the affine combination is no longer limited to only two functions [38, 39].

2.3 Randomly Generated Functions

Apart from the benchmarking suites, a set of functions can be generated using the function generator proposed in [36], covering a diverse set of optimization problem classes, as shown in [40]. By using a set of selection pressures, mathematical operands and operators are randomly selected from a predefined pool to construct tree-structured function expressions, which we call RGF. In fact, RGF with similar landscape characteristics to automotive crashworthiness optimization problems can be created, which is lacking in the BBOB suite [15].

Nonetheless, the properties of RGF are not known a priori, e.g., the global optimum and optimization complexity, as oppose to the well-studied BBOB suite. To tackle this problem, an extension has been attempted on this function generator to guide the function generation towards specific optimization complexity using genetic programming [16], which is beyond the scope of this work.

2.4 Exploratory Landscape Analysis

In landscape-aware ASP, exploratory landscape analysis (ELA) is one of the popular approaches employed to numerically quantify the high-level landscape characteristics of continuous optimization problems, such as multi-modality and global structure. While initially only six fundamental feature classes were proposed in ELA, namely y -distribution, level sets, meta-models, local searches,

curvature, and convexity [21, 22], more feature classes have been progressively proposed to complement them, e.g., dispersion, nearest better clustering (NBC), principal component analysis (PCA), linear models, and information content of fitness sequences (ICoFiS) [10, 12, 18, 24].

In brief, a design of experiments (DoE) is required for the ELA features computation, consisting of some samples $\mathcal{X} = \{x_1, \dots, x_n\}$ and objective values $\mathcal{Y} = \{y_1, \dots, y_n\}$, which are computed using an objective function f , i.e., $f: \mathcal{X} \rightarrow \mathcal{Y}$, where $x_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}$, and n is the sample size. Consequently, the effectiveness of ELA features can be dependent on the DoE sample size, dimensionality, and sampling strategy [29]. To overcome potential bias of the hand-crafted ELA features in capturing specific landscape characteristics, deep NN-based methods have been explored to characterize BBO problems based on latent space features, e.g., DoE2Vec [32], which we leave for future work.

3 Methodology

The workflow of our landscape-aware AAC approach is visualized in Fig. 1. In the first step, the landscape characteristics of RGF are captured using ELA and the corresponding best configurations are identified using HPO during the training phase. Next, the ELA features and configurations are properly pre-processed for the training of NN models. Eventually, optimal configurations for unseen BBO problems can be predicted based on their ELA features using the trained models. Our approach is described in detail in the following.

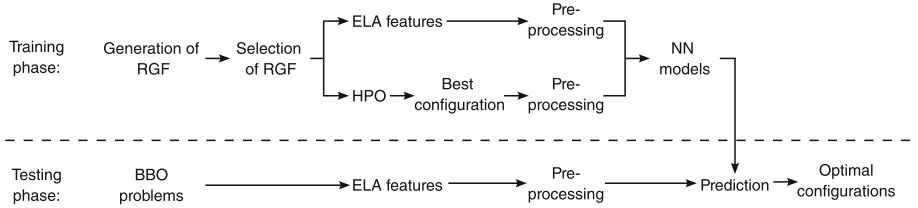


Fig. 1. An overview of our proposed landscape-aware AAC approach that can identify optimal configurations for BBO problems, consisting of a training and testing phase. During the training phase, using a preferably large set of RGF, the respective ELA features and optimal configurations identified through HPO (performed on RGF) are utilized to train NN models. The pre-trained models can then be deployed to predict the best suited configuration for unseen BBO problems based on their ELA features in the testing phase.

Generation and Selection of RGF. Firstly, a large set of RGF is generated for the training of NN models, using the function generator implemented in [15]. Before the model training, a pre-selection step is integrated to identify RGF that are appropriate for AAC purposes, refer to Sect. 3.1 for detailed explanations.

Computation of ELA Features. Secondly, the optimization landscape characteristics of RGF are computed using ELA based on some DoE samples. To combat inherent bias [27], the objective values are normalized using min-max scaling before the ELA features computation. Since many of the ELA features are redundant [30], highly correlated ELA features based on Pearson’s correlation coefficient (>0.95) are discarded, using a similar approach as in [15]. To improve the performance of NN models, we ensure that the remaining ELA features are within a comparable scale range via normalization using min-max scaling.

Identifying the best Configuration using HPO. For each individual RGF, the best performing algorithm configuration found using HPO is considered as the best suited configuration identified for that function. Similar to the ELA features, the configuration data are pre-processed for the model training, where categorical hyperparameters are one-hot encoded, while continuous hyperparameters are linearly re-scaled to the scale range of $[0, 1]$ using Eq. 1.

$$z_{new} = \frac{z_{init} - a_{min}}{a_{max} - a_{min}} \cdot (b_{max} - b_{min}) + b_{min}, \quad (1)$$

where z_{init} and z_{new} are the initial and re-scaled values, a_{max} and a_{min} are the lower and upper bound before re-scaling, and b_{max} and b_{min} are the lower and upper bound after re-scaling. In this work, we focus on finding the best configuration of modular CMA-ES.

Training of NN Models. For the training of NN models, the pre-processed ELA features are employed as input, while the best configurations identified using HPO as output. Detailed descriptions of the NN models are included in Sect. 3.2.

Optimal Configurations for BBO Problems. During the deployment or testing phase, the trained NN models can be used to predict optimal configurations of modular CMA-ES for unseen BBO problems based on their ELA features. Similar to the training phase, the input ELA features of BBO problems are normalized, while the predicted configurations are inversely transformed back to the original configuration search space. To avoid invalid configurations, e.g., negative population size, predicted continuous hyperparameters that fall outside the search space will be set to either the lower or upper boundary.

3.1 Selection of Appropriate RGF

Unlike the well understood BBOB functions, the landscape characteristics and global optimum of RGF are not known a priori. Due to the fact that some RGF are insufficiently discriminative in distinguishing different configurations based on their optimization performances, not all RGF are appropriate for AAC purposes based on our preliminary testing. Using the HPO results on three chosen RGF in Fig. 2 as an example, we consider functions with a similar pattern to

‘RGF1’ as ideal for AAC purposes, where a clear configuration ranking with only a few ties is possible. More importantly, the best configuration can be easily identified. On the other hand, functions similar to ‘RGF2’ are considered as inappropriate for AAC, where many, or in extreme situations, all configurations are equally good, leading to an ambiguous ranking. We suspect that the optimization complexity of such RGF is too low that the choice of configuration does not matter. Surprisingly, two RGF with a small difference in their ELA features can have the opposite patterns, which raises questions for future research. To improve the robustness of trained models, functions similar to ‘RGF3’ are additionally neglected, where the global optimum seems to be an extreme outlier and can be found occasionally by a few configurations.

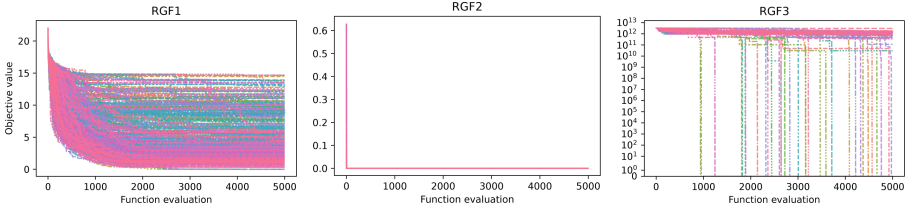


Fig. 2. The optimization convergence of 500 configurations evaluated using HPO on three chosen RGF. The x-axis shows the number of function evaluations, while the y-axis shows the re-scaled objective values, with 0 being the best solution found in all runs. Each curve represents a configuration run using modular CMA-ES (median over 10 repetitions). (*Left*) Ideal for AAC purposes, where a clear ranking of configurations is possible. (*Middle*) Ambiguous ranking of algorithm configurations, where all configurations are equally competitive. (*Right*) The global optimum seems to be an outlier that can only be found by a few configurations.

To overcome these problems, the following measures are implemented to identify RGF that are appropriate for AAC purposes.

1. **Estimation of global optimum:** In a brute-force manner, we perform HPO on each RGF, focusing on finding a better solution, i.e., a smaller objective value, and using a similar setup as described in Sect. 4. Eventually, the global optimum y_{opt} is approximated based on the best solution found in all HPO runs y_{hpo} using Eq. 2.

$$y_{opt} = \begin{cases} \lfloor y_{hpo} \rfloor, & \text{if } 0 \leq |y_{hpo}| < 10 \\ \lfloor y_{hpo}/10 \rfloor \cdot 10, & \text{if } 10 \leq |y_{hpo}| < 100, \\ \lfloor y_{hpo}/10^p \rfloor \cdot 10^p, & \text{otherwise} \end{cases} \quad (2)$$

$$p = \lfloor \log_{10} |y_{hpo}| \rfloor - 1,$$

where y_{hpo} is either rounded to the nearest lower integer for a small $|y_{hpo}|$, or rounded based on the nearest lower power of 10. Having an estimated global

optimum for RGF is essential in our approach to facilitate an evaluation of configuration performance (refer to Sect. 4.1) and a comparison between different functions with varying scale ranges.

2. **RGF appropriate for AAC:** Using the same HPO results from previous step, all configurations evaluated are ranked according to their performances, where ties are assigned with the same rank. The ranking ambiguity is evaluated based on the Kendall rank correlation coefficient between the HPO configuration ranking and a strict ranking (without tie). For a correlation lower than 0.9, e.g., due to too many ties, such ranking is considered as ambiguous. Furthermore, we compute the standard score or z-score of the global optimum found to estimate its deviation from the distribution of other solutions. When the global optimum is 3 standard deviations away from the distribution mean, it is considered as an extreme outlier.
3. **Elimination of RGF:** A RGF is excluded from the training data, if any of the aforementioned conditions is fulfilled.

While additional computational effort is required for the above-mentioned measures in identifying RGF appropriate for AAC purposes, we argue that they are critical in improving the performance of NN models. Moreover, this process needs to be done only once, since the RGF identified can be re-used in the future for the same BBO problem classes.

3.2 Multi-output Mixed Regression and Classification

Dense Neural Network: In this work, we investigate the potential of dense NN models with the following architecture for the multi-output mixed regression and classification tasks in landscape-aware AAC, as visualized in Fig. 3.

- **Input layer:** The size of the input layer is equal to the number of ELA features available in the training data.
- **Hidden layers:** To determine an optimal inner architecture, different combinations of number of hidden layer $\{1, 2, 3\}$, hidden layer sizes $\{16, 32, 64, 128\}$, and epochs $\{100, 150, 200\}$ are evaluated using a grid search approach, 80 : 20 train-test split of the training data, and a repetition of five times. Eventually, the hidden layers are constructed based on the combination with the smallest validation loss and assigned with rectified linear unit (ReLU) as activation function.
- **Output layers:** In short, different layers are assigned for the mixed regression and classification tasks. While a single output layer with linear activation function is dedicated for the multi-output regression task, the multi-output multi-class classification task is split into multiple classifications tasks. Precisely, an output layer with softmax activation function is allocated for each categorical hyperparameter. Consequently, the size of each output layer depends on the number of hyperparameters respectively.

- **Loss functions:** The dense NN models are trained using mean squared error as loss function for regression and categorical cross entropy for classification task.

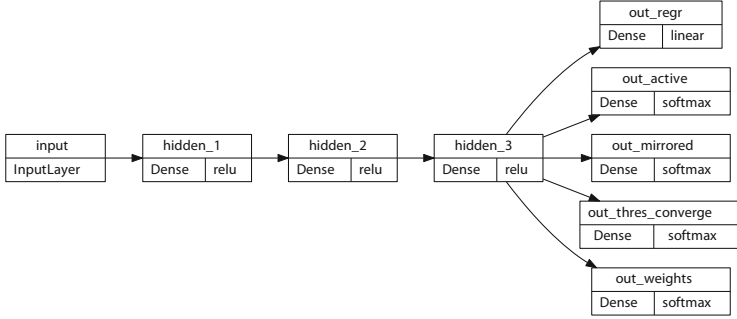


Fig. 3. An example of the architecture of a dense NN model. From left to right, an input layer, three hidden layers, and several output layers, with one output layer for regression and four layers for classification tasks.

Random Forest: For a fair evaluation, the performance of trained NN models is compared against RF models, which are popular in landscape-aware ASP. Precisely, the RF models are optimally constructed with fine-tuned configurations using `auto-sklearn` [5], an automated CASH tool designed for ML, and 80 : 20 train-test split of the training data. Since multi-output multi-class classification is currently limited in `auto-sklearn`, the algorithm configuration problem is defined as a multi-target regression task, where the categorical hyperparameters are encoded as numerical labels.

4 Experimental Setup

In brief, the scope of our investigations can be summarized as follows:

- In $5d$, using a set of 1 000 RGF as training data, while the 24 BBOB functions of the first instance as unseen test problems. For a comprehensive analysis, we also investigate models trained using 1 000 MA-BBOB functions and a combination of both RGF and MA-BBOB functions;
- An optimization landscape is characterized based on a total of 68 ELA features that can be computed without requiring additional function evaluations, using a DoE of $50 \cdot d$ samples, `pflacco` [28], and a similar workflow proposed in [15];
- In this work, we consider fine-tuning the configuration of modular CMA-ES within the configuration search space in Table 1, with all optimization runs are allocated with a budget of $1\,000 \cdot d$ evaluations and 10 repetitions; and

Table 1. An overview of the 11 hyperparameters of modular CMA-ES considered for AAC. The default configuration is highlighted in bold, where the default learning rates are automatically determined based on other hyperparameters. The ‘number of children’ predicted by predictive models is rounded-off to integer. Symbol: $\mathbb{Z}$ for integer, $\mathbb{R}$ for continuous variable, and $\mathbf{C}$ for categorical variable.

Num.	Hyperparameter	Type	Domain
1	Number of children	$\mathbb{Z}$	$\{ 5, \dots, 50 \} (4 + \lfloor (3 \cdot \ln(d)) \rfloor)$
2	Number of parent (as ratio of children)	$\mathbb{R}$	$[0.3, 0.5] (0.5)$
3	Initial standard deviation	$\mathbb{R}$	$[0.1, 0.5] (0.2)$
4	Learning rate step size control	$\mathbb{R}$	$[0.0, 1.0]$
5	Learning rate covariance matrix adaptation	$\mathbb{R}$	$[0.0, 1.0]$
6	Learning rate rank- μ update	$\mathbb{R}$	$[0.0, 0.35]$
7	Learning rate rank-one update	$\mathbb{R}$	$[0.0, 0.35]$
8	Active update	$\mathbf{C}$	$\{ \text{True}, \text{False} \}$
9	Mirrored sampling	$\mathbf{C}$	$\{ \text{none}, \text{‘mirrored’}, \text{‘mirrored pairwise’} \}$
10	Threshold convergence	$\mathbf{C}$	$\{ \text{True}, \text{False} \}$
11	Recombination weights	$\mathbf{C}$	$\{ \text{‘default’}, \text{‘equal’}, \text{‘}1/2^\wedge\text{lambda’} \}$

- The TPE available in **HyperOpt** [2] is employed to identify optimal configurations of modular CMA-ES and assigned with a budget of 500 evaluations.

To analyze the performance of our approach for BBO problems in higher dimensionality, our investigations are extended to $20d$ using a smaller experimental scope to minimize computational effort, namely a DoE of $20 \cdot d$ samples for ELA features computation, $100 \cdot d$ evaluations for optimization runs, 300 evaluations for TPE, and only the seven real-valued hyperparameters of modular CMA-ES are considered.

4.1 Optimization Performance Metric

For real-world applications, (i) it is often practical to find good solutions within a shorter time, rather than finding the global optimum, and (ii) the global optimum is usually not known, making it difficult to use some popular performance metrics, e.g., expected hitting time [37]. Hence, we propose to measure the performance of a configuration based on its area under the curve (AUC) of optimization convergence (Fig. 2). By minimizing the AUC metric, we are essentially searching for configurations that have an optimal trade-off between the solution found and convergence speed. In this work, all AUC during HPO are computed using the min-max normalized objective values based on the global optimum and worst DoE sample.

4.2 Optimization Baseline

Principally, we consider the following three algorithm configurations as comparison reference to evaluate the potential of our approach.

- **Default configuration:** The readily available configuration in its original implementation that is simply utilized by practitioners with limited experience in fine-tuning configurations. Inline with our motivation, our approach is primarily compared against it.
- **SBS:** The configuration that can perform well on average across all 24 BBOB functions and serves as our secondary target to beat in this work. Precisely, it is identified based on the mean performance of configurations evaluated across all BBOB functions.
- **Virtual best solver (VBS):** The best performing configuration for a particular BBOB function, which can be treated as the lower bound.

Unlike typical ASP approaches, where the SBS and VBS are selected from a portfolio of limited algorithms using grid search, evaluating all possible configurations within the large search space in Table 1 is computationally infeasible. Subsequently, we determine both solvers via HPO using TPE within an allocated budget. Due to the stochastic nature of TPE, there might be configurations that can outperform the VBS identified, but are not discovered during HPO.

5 Results

Due to the limited space, experimental results and figures not included in this paper can be found in our repository at <https://doi.org/10.5281/zenodo.10965507>.

5.1 Representativeness of Training Data

Before delving into analyzing the configuration performances, we take a closer look at the representativeness of training data. Naturally, predictive models trained using MA-BBOB functions are expected to perform well, since the problem classes available in the training data should sufficiently cover the BBOB suite. While this can be observed most of the time, it is not always the case, notably for F7 (step ellipsoidal) and F12 (Bent Cigar) in Sect. 5.2. The poor performances could be due to the insufficient coverage of ELA feature space by MA-BBOB functions, as shown in Fig. 4, which might be related to the generation of MA-BBOB functions [38]. In comparison, RGF can better cover the ELA feature space, highlighting the benefits of using RGF as training data. In fact, it seems to be advantageous to combine the large distribution of RGF and the more focused distribution of MA-BBOB on some of the BBOB functions.

5.2 Performance of Predicted Configurations

The optimization performances using different configurations for 24 BBOB functions in 5d are compared in Fig. 5. In general, the optimal configurations identified using predictive models can clearly outperform the default configuration on most BBOB functions. On the other hand, the predicted configurations seem to

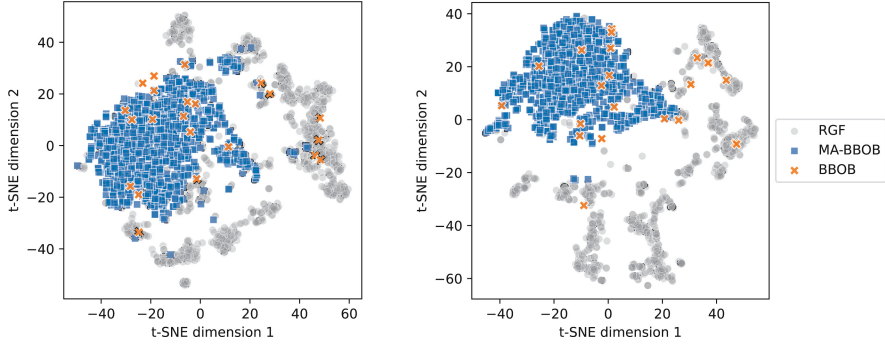


Fig. 4. Projection of the ELA feature space to a $2d$ visualization using t-distributed stochastic neighbor embedding (t-SNE) [19] for 1000 RGF, 1000 MA-BBOB, and 24 BBOB functions in $5d$ (left) and $20d$ (right), using a similar approach as in [15].

be competitive against the SBS, such as for F7 and F17 (Schaffers F7). Not only that, our approach using NN models can perform better than the SBS in some cases, for instance, for F5 (linear slope) and F13 (sharp ridge). Nonetheless, the performance of predicted configurations is lacking for highly multi-modal functions, e.g., F16 (Weierstrass) and F23 (Katsuura), which might be due to the absence of ELA features that can accurately capture the landscape characteristics of such complex functions, revealing the weaknesses in our approach. When compared against the VBS, the predicted configurations sometimes seem to have a comparable performance, e.g., for F21 (Gallagher’s Gaussian 101-me peaks).

Using the Wilcoxon signed-rank test with the hypothesis *optimal configurations identified using our approach can perform better*, we statistically evaluate the performance of different configurations. Precisely, we focus on comparing NN models against the default configuration, SBS, and RF models, using RGF as training data. Inline with our previous observations, optimal configurations predicted using our approach can indeed beat the default configuration for most BBOB functions, while outperforming the SBS on many BBOB functions, as depicted in Fig. 6. It is worth reminding that our approach can be competitive against the default configuration and SBS in a few remaining BBOB functions, as previously discussed in Fig. 5. This analysis also indicates that our current approach is more effective on simple functions (first half of the BBOB suite) compared to complex functions (second half), which might be related to the ELA features. Apart from that, the performances of NN models are as good as or even better than RF models for some BBOB functions, particularly in $5d$.

As illustrated in Fig. 6, we can in general have similar observations for the BBOB functions in $20d$. When compared to the default configuration, our approach are more effective for many BBOB functions. Nevertheless, the performance improvements gained using our approach compared to the SBS in $20d$ are less than in $5d$, showing rooms for improvement in high dimensionality.

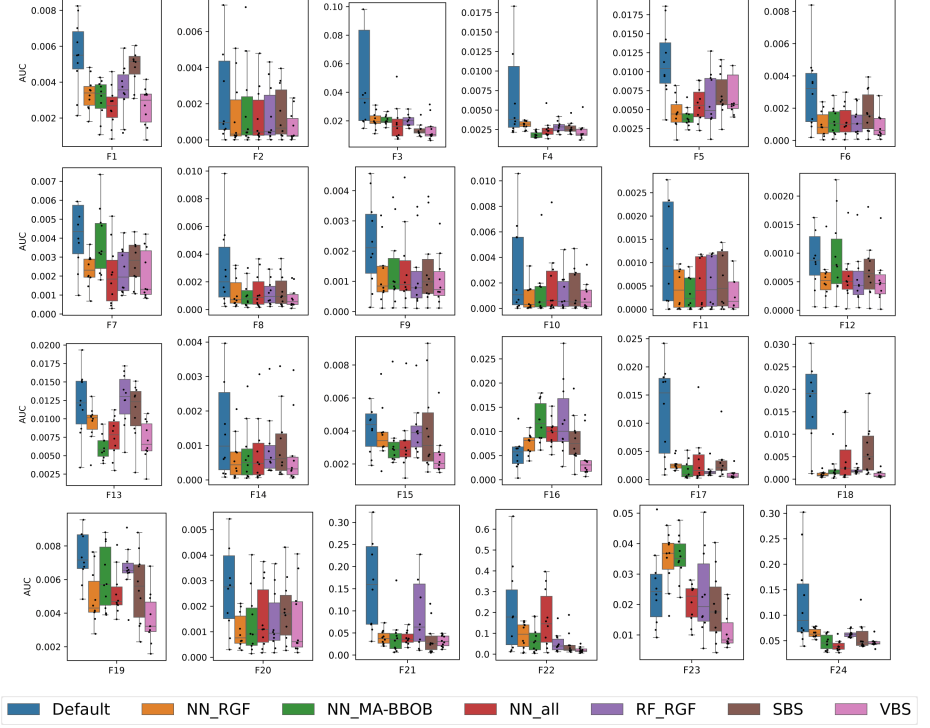


Fig. 5. Performance of modular CMA-ES using different configurations for 24 BBOB functions in $5d$, each repeated for 10 times. The AUC is computed based on objective values min-max normalized using the global optimum and worst solution in all configurations, divided by the evaluation budget. A lower AUC is better.

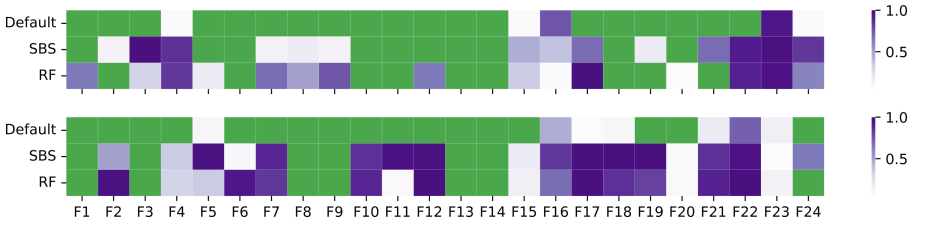


Fig. 6. Pairwise performance comparison between the configuration predicted using NN models (our approach) against the default configuration, SBS, and RF models for 24 BBOB functions in $5d$ (top) and $20d$ (bottom) based on the p-value computed using the Wilcoxon signed-rank test. The green color indicates that there is statistically significant evidence to support the hypothesis *optimal configurations predicted using NN models can perform better*, with a p-value smaller than 0.05. Alternatively, a darker purple color (larger p-value) indicates that the hypothesis is more likely to be rejected, while a lighter purple color (smaller p-value) for a lower chance of rejection.

6 Conclusions and Future Work

Aiming to assist practitioners unfamiliar with fine-tuning of algorithm configurations, we propose to construct general purpose predictive models towards landscape-aware AAC that can identify optimal algorithms as well as hyperparameters for different practical applications. To improve the generalization of our approach, we consider tree-based RGF as training data, which covers a diverse set optimization problem classes. Furthermore, a pre-selection step is implemented to select RGF that are appropriate for AAC purposes, and thus, to improve the prediction accuracy. Moreover, we investigate the potential of dense NN models for the multi-output mixed regression and classification tasks, which can easily handle the mixed-integer search space and large training data sets.

When evaluated on the BBOB suite in $5d$ and $20d$ using modular CMA-ES, our results reveal that we can predict near-optimal configurations that outperform the default configuration and compete against the SBS in most cases. This is particularly encouraging for real-world applications, where such a SBS is usually not available. In fact, properly selected RGF have promising potential as training data for landscape-aware AAC, since they cover a broader spectrum of function complexity compared to BBOB and MA-BBOB functions. Subsequently, we believe that our approach can generalize well beyond the BBOB suite, provided that the unseen problems is well represented by the RGF training set. Overall, configurations with better performance can be best identified using dense NN models trained on a combination of RGF and MA-BBOB functions.

For future work, we plan to improve our investigations as follows:

- The configuration search space can be expanded to include a variety of optimization algorithms and hyperparameters;
- The performance of NN models can be further improved by fine-tuning more hyperparameters using an optimizer, e.g., learning rate and batch size;
- An analysis can be extended to better understand the impact of ELA features pre-processing, e.g., using normalization vs. standardization;
- To further minimize the overall computational costs, alternatives that can efficiently identify RGF appropriate for AAC purposes can be explored;
- Despite the fact that the estimated y_{opt} seems to be robust in our work, i.e., always smaller than all solutions found, further investigations are needed for confirmation and/or improvements; and
- Eventually, we aim to evaluate and quantify the benefits of our approach for real-world expensive BBO problems.

Acknowledgments. The contribution of this paper was written as part of the joint project newAIDE under the consortium leadership of BMW AG with the partners Altair Engineering GmbH, divis intelligent solutions GmbH, MSC Software GmbH, Technical University of Munich, TWT GmbH. The project is supported by the Federal Ministry for Economic Affairs and Climate Action (BMWK) on the basis of a decision by the German Bundestag.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Bergstra, J., Bardenet, R., Bengio, Y., Kégl, B.: Algorithms for hyper-parameter optimization. In: *Advances in Neural Information Processing Systems*, vol. 24 (2011)
2. Bergstra, J., Yamins, D., Cox, D.: Making a science of model search: hyperparameter optimization in hundreds of dimensions for vision architectures. In: *International Conference on Machine Learning*, pp. 115–123. PMLR (2013)
3. Dietrich, K., Mersmann, O.: Increasing the diversity of benchmark function sets through affine recombination. In: *Parallel Problem Solving from Nature–PPSN XVII: 17th International Conference, PPSN 2022, Dortmund, Germany, 10–14 September 2022, Proceedings, Part I*, pp. 590–602. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-14714-2_41
4. Doerr, C., Wang, H., Ye, F., van Rijn, S., Bäck, T.: IOHprofiler: a benchmarking and profiling tool for iterative optimization heuristics. *arXiv e-prints:1810.05281* (2018). <https://arxiv.org/abs/1810.05281>
5. Feurer, M., Eggenberger, K., Falkner, S., Lindauer, M., Hutter, F.: Auto-Sklearn 2.0: Hands-free AutoML via meta-learning. *J. Mach. Learn. Res.* **23**(261), 1–61 (2022)
6. Hansen, N., Auger, A., Ros, R., Mersmann, O., Tušar, T., Brockhoff, D.: COCO: a platform for comparing continuous optimizers in a black-box setting. *Optim. Methods Softw.* **36**(1), 114–144 (2021). <https://doi.org/10.1080/10556788.2020.1808977>
7. Hansen, N., Finck, S., Ros, R., Auger, A.: Real-Parameter black-box optimization benchmarking 2009: noiseless functions definitions. Research Report RR-6829, INRIA (2009). <https://hal.inria.fr/inria-00362633>
8. Hansen, N., Ostermeier, A.: Adapting arbitrary normal mutation distributions in evolution strategies: the covariance matrix adaptation. In: *Proceedings of IEEE International Conference on Evolutionary Computation*, pp. 312–317. IEEE (1996)
9. Hansen, N., Ostermeier, A.: Completely derandomized self-adaptation in evolution strategies. *Evol. Comput.* **9**(2), 159–195 (2001)
10. Kerschke, P., Preuss, M., Wessing, S., Trautmann, H.: Detecting funnel structures by means of exploratory landscape analysis. In: *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation*, pp. 265–272. GECCO 2015, Association for Computing Machinery, New York, NY, USA (2015). <https://doi.org/10.1145/2739480.2754642>
11. Kerschke, P., Trautmann, H.: Automated algorithm selection on continuous black-box problems by combining exploratory landscape analysis and machine learning. *Evol. Comput.* **27**(1), 99–127 (2019). https://doi.org/10.1162/evco_a_00236
12. Kerschke, P., Trautmann, H.: comprehensive feature-based landscape analysis of continuous and constrained optimization problems using the r-package flacco. In: Bauer, N., Ickstadt, K., Lübke, K., Szepannek, G., Trautmann, H., Vichi, M. (eds.) *Applications in Statistical Computing. SCDAKO*, pp. 93–123. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-25147-5_7
13. Lindauer, M., et al.: SMAC3: a versatile bayesian optimization package for hyperparameter optimization. *J. Mach. Learn. Res.* **23**(54), 1–9 (2022). <http://jmlr.org/papers/v23/21-0888.html>

14. Long, F.X., van Stein, B., Frenzel, M., Krause, P., Gitterle, M., Bäck, T.: Learning the characteristics of engineering optimization problems with applications in automotive crash. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 1227–1236. GECCO 2022, Association for Computing Machinery, New York, NY, USA (2022). <https://doi.org/10.1145/3512290.3528712>
15. Long, F.X., van Stein, B., Frenzel, M., Krause, P., Gitterle, M., Bäck, T.: Generating cheap representative functions for expensive automotive crashworthiness optimization. *ACM Trans. Evol. Learn. Optim.* **4**(2) (2024). <https://doi.org/10.1145/3646554>
16. Long, F.X., Vermetten, D., Kononova, A., Kalkreuth, R., Yang, K., Bäck, T., van Stein, N.: Challenges of ELA-guided function evolution using genetic programming. In: Proceedings of the 15th International Joint Conference on Computational Intelligence - Volume 1: ECTA, pp. 119–130. INSTICC, SciTePress (2023). <https://doi.org/10.5220/0012206200003595>
17. Long, F.X., Vermetten, D., van Stein, B., Kononova, A.V.: BBOB Instance Analysis: Landscape Properties and Algorithm Performance Across Problem Instances. In: Correia, J., Smith, S., Qaddoura, R. (eds.) Applications of Evolutionary Computation: 26th European Conference, EvoApplications 2023, Held as Part of EvoStar 2023, Brno, Czech Republic, 12–14 April 2023, Proceedings, pp. 380–395. Springer, Heidelberg (2023). https://doi.org/10.1007/978-3-031-30229-9_25
18. Lunacek, M., Whitley, D.: The dispersion metric and the CMA evolution strategy. In: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation, pp. 477–484. GECCO 2006, Association for Computing Machinery, New York, NY, USA (2006). <https://doi.org/10.1145/1143997.1144085>
19. van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. *J. Mach. Learn. Res.* **9**(86), 2579–2605 (2008). <http://jmlr.org/papers/v9/vandermaaten08a.html>
20. Malan, K.M.: A survey of advances in landscape analysis for optimisation. *Algorithms* **14**(2), 40 (2021). <https://doi.org/10.3390/a14020040>
21. Mersmann, O., Bischl, B., Trautmann, H., Preuss, M., Weihs, C., Rudolph, G.: Exploratory landscape analysis. In: Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation, pp. 829–836. GECCO 2011, Association for Computing Machinery, New York, NY, USA (2011). <https://doi.org/10.1145/2001576.2001690>
22. Mersmann, O., Preuss, M., Trautmann, H.: Benchmarking evolutionary algorithms: towards exploratory landscape analysis. In: Schaefer, R., Cotta, C., Kołodziej, J., Rudolph, G. (eds.) PPSN 2010. LNCS, vol. 6238, pp. 73–82. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15844-5_8
23. Mockus, J.: The Bayesian approach to global optimization. In: Drenick, R.F., Kozin, F. (eds.) System Modeling and Optimization, pp. 473–481. Springer, Heidelberg (1982). <https://doi.org/10.1007/BFb0006170>
24. Muñoz, M.A., Kirley, M., Halgamuge, S.K.: Exploratory landscape analysis of continuous space optimization problems using information content. *IEEE Trans. Evol. Comput.* **19**(1), 74–87 (2015). <https://doi.org/10.1109/TEVC.2014.2302006>
25. Muñoz, M.A., Sun, Y., Kirley, M., Halgamuge, S.K.: Algorithm selection for black-box continuous optimization problems: a survey on methods and challenges. *Inf. Sci.* **317**, 224–245 (2015). <https://doi.org/10.1016/j.ins.2015.05.010>
26. de Nobel, J., Vermetten, D., Wang, H., Doerr, C., Bäck, T.: Tuning as a means of assessing the benefits of new ideas in interplay with existing algorithmic modules. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, pp. 1375–1384. GECCO 2021, Association for Computing Machinery, New York, NY, USA (2021). <https://doi.org/10.1145/3449726.3463167>

27. Prager, R.P., Trautmann, H.: Nullifying the inherent bias of non-invariant exploratory landscape analysis features. In: Correia, J., Smith, S., Qaddoura, R. (eds.) Applications of Evolutionary Computation: 26th European Conference, EvoApplications 2023, Held as Part of EvoStar 2023, Brno, Czech Republic, 12–14 April 2023, Proceedings, vol. 13989, pp. 411–425. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-30229-9_27
28. Prager, R.P., Trautmann, H.: Pflacco: Feature-based landscape analysis of continuous and constrained optimization problems in Python. *Evol. Comput.*, 1–25 (2023). https://doi.org/10.1162/evco_a_00341
29. Renau, Q., Doerr, C., Dreio, J., Doerr, B.: Exploratory landscape analysis is strongly sensitive to the sampling strategy. In: Bäck, T., et al. (eds.) PPSN 2020. LNCS, vol. 12270, pp. 139–153. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58115-2_10
30. Renau, Q., Dreio, J., Doerr, C., Doerr, B.: Expressiveness and robustness of landscape features. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, pp. 2048–2051. GECCO 2019, Association for Computing Machinery, New York, NY, USA (2019). <https://doi.org/10.1145/3319619.3326913>
31. Simoncini, D., Barbe, S., Schiex, T., Verel, S.: Fitness landscape analysis around the optimum in computational protein design. In: Proceedings of the Genetic and Evolutionary Computation Conference, pp. 355–362. GECCO 2018, Association for Computing Machinery, New York, NY, USA (2018). <https://doi.org/10.1145/3205455.3205626>
32. van Stein, B., Long, F.X., Frenzel, M., Krause, P., Gitterle, M., Bäck, T.: DoE2Vec: deep-learning based features for exploratory landscape analysis. In: Proceedings of the Companion Conference on Genetic and Evolutionary Computation, pp. 515–518. GECCO 2023 Companion, Association for Computing Machinery, New York, NY, USA (2023). <https://doi.org/10.1145/3583133.3590609>
33. van Stein, N., Vermetten, D., Kononova, A.V., Bäck, T.: Explainable benchmarking for iterative optimization heuristics. arXiv preprint [arXiv:2401.17842](https://arxiv.org/abs/2401.17842) (2024). <https://arxiv.org/abs/2401.17842>
34. Thomaser, A., Kononova, A.V., Vogt, M.E., Bäck, T.: One-shot optimization for vehicle dynamics control systems: towards benchmarking and exploratory landscape analysis. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, pp. 2036–2045. GECCO 2022, Association for Computing Machinery, New York, NY, USA (2022). <https://doi.org/10.1145/3520304.3533979>
35. Thornton, C., Hutter, F., Hoos, H.H., Leyton-Brown, K.: Auto-WEKA: combined selection and hyperparameter optimization of classification algorithms. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. p. 847–855. KDD 2013, Association for Computing Machinery, New York, NY, USA (2013). <https://doi.org/10.1145/2487575.2487629>
36. Tian, Y., Peng, S., Zhang, X., Rodemann, T., Tan, K.C., Jin, Y.: A recommender system for metaheuristic algorithms for continuous optimization based on deep recurrent neural networks. *IEEE Trans. Artif. Intell.* **1**(1), 5–18 (2020). <https://doi.org/10.1109/TAI.2020.3022339>
37. Vermetten, D., Wang, H., Doerr, C., Bäck, T.: Integrated vs. sequential approaches for selecting and tuning CMA-ES variants. In: Proceedings of the 2020 Genetic and Evolutionary Computation Conference, p. 903–912. GECCO 2020, Association for Computing Machinery, New York, NY, USA (2020). <https://doi.org/10.1145/3377930.3389831>

38. Vermetten, D., Ye, F., Bäck, T., Doerr, C.: MA-BBOB: a problem generator for black-box optimization using affine combinations and shifts (2023). <https://arxiv.org/abs/2312.11083>
39. Vermetten, D., Ye, F., Doerr, C.: Using Affine Combinations of BBOB Problems for Performance Assessment. CoRR **abs/2303.04573** (2023). <https://doi.org/10.48550/arXiv.2303.04573>
40. Škvorc, U., Eftimov, T., Korošec, P.: A complementarity analysis of the COCO benchmark problems and artificially generated problems. In: Proceedings of the Genetic and Evolutionary Computation Conference Companion, p. 215–216. Association for Computing Machinery, New York, NY, USA (2021). <https://doi.org/10.1145/3449726.3459585>
41. Zhao, M., Li, J.: Tuning the hyper-parameters of CMA-ES with tree-structured Parzen estimators. In: 2018 Tenth International Conference on Advanced Computational Intelligence (ICACI), pp. 613–618 (2018). <https://doi.org/10.1109/ICACI.2018.8377530>
42. Škvorc, U., Eftimov, T., Korošec, P.: Transfer learning analysis of multi-class classification for landscape-aware algorithm selection. *Mathematics* **10**(3) (2022). <https://doi.org/10.3390/math10030432>, <https://www.mdpi.com/2227-7390/10/3/432>

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

