Compositional Multi-objective Parameter tuning

This manuscript (permalink) was automatically generated from Valavanca/compositional-system-for-hyperparameter-tuning@3da4232 on March 2, 2020.

Authors

- **Oleksandr Husak**
  XXXX-XXXX-XXXX-XXXX · Valavanca
  Research Student Assistant in TUD

Introduction. Challenges and Problems.

The main goal of this thesis is to investigate portfolio of surrogate models that can be used to improve applicability model-based optimization methods to a verity of problems such as parameter tuning. Surrogate model or models based optimization is a common approach for a deal with expensive black-box function, but as far as the author is aware, there is no published research where the influence of heterogeneous portfolio of surrogate models was studied. The main target problem is an expensive multi-objective problem but the developed approach is also suitable for expensive single-objective optimization. As black-box, we can't say what type of surface does the problem have. That is why it should be customized in the optimization process. The goal is to determine if the variability in extrapolation worth it

Multi-objective optimisation is an established parameter tuning technique. It is especially suited to solve complex, multidisciplinary design problems with an accent on system design.

When we talk about several objectives, the intention is to find good compromises rather than a single solution as in global optimization. Since the solution for multi-objective optimization problems gives the appearance to a set of Pareto-optimal points, evolutionary optimization algorithms are ideal for handling multi-objective optimization problems.

General optimization methods could be classified into derivative and non-derivative methods. In this thesis focuses on non-derivative methods, as they are more suitable for parameter tuning. Therefore, they are also known as black-box methods and do not require any derivatives of the objective function to calculate the optimum. Other benefits of these methods are that they are more likely to find a global optimum.

Motivation

The purpose of this study is to introduce new surrogate-design-criteria for multi-objective hyperparameter optimization software.

Motivation Examples in tuning algorithms:

- Local search: neighbourhoods, perturbations, tabu length, annealing
- Tree search: pre-processing, data structures, branching heuristics, clause learning deletion
- Genetic algorithms: population size, mating scheme, crossover, mutation rate, local improvement stages, hybridizations
- Machine Learning: pre-processing, learning rate schedules

- Deep learning (in addition): layers, dropout constants, pre-training, activations functions, units/layer, weight initialization

Objectives

Black-box multi-objective problems given a finite number of function evaluations

Research Questions

1. RQ(Surrogate portfolio): How a surrogate portfolio influence on the optimization process?

2. RQ(Composition model): How a compositional surrogate model influence on the optimization process?

3. RQ(Sampling plan): Is the relation of the sampling plan with a surrogate validation can reduce samples set size?

In numerous test problems, compositional-surrogate finds comparable solutions to standard MOEA (NSGA-II, MOEAD, MACO, NSPSO) doing considerably fewer evaluations (300 vs 5000). Surrogate-based optimization is recommended when a model is expensive to evaluate.

Foundation

Intro

General background information introduce. What is parameter tuning? Why multi-objective is important? Why we can't use the standard multi-objective approach in real-life problem/parameter tuning task, and why model-based or surrogate optimization is the best solution?

In common old-fashioned software design, engineers carefully convert overall models into domain-specific tools. In this approach, designers codify the current understanding of the problem into the parameters.

Parameter tuning

Given recent advances in computing hardware, software analysts either validate engineer models or find optimal configuration by using parameter tuning tools to explore thousands to millions of inputs for their systems.

In this article assume that parameter tuning is a subset problem of general, global optimizations. It's also mean that we consider some fitness function 'f' that converts the parameter vector to output objectives. Note that the term "real evaluation" or "black-box evaluation" as a synonym for the fitness function 'f'. The goal of parameter tuning as an optimization task lay on fast iterative search with improvements in each objective dimension. The term "fast" means that the convergence to global optimum is achieved with the least real evaluations and shorter time frame.

We consider fitness function 'f' as black-box with parameter and objective space. Parameter space has structure and could consist from continues and categorical dimensions. Sometimes, some combinations of parameter settings are forbidden. Each pony from parameter space lead to some point in objective space. Configurations often yield qualitatively different behavior. Objective space also

could be described as usual objectives as accuracy, runtime, latency, performance, error rate, energy, et.s. On each objective should gain the best possible value and rich system tradeoff.

Optimization technics:

- Grid search vs Random search
- Heuristics and Metaheuristic. (Simulated annealing, Evolutionary algorithm..) These methods aim at generating approximately optimal solutions in a single run. Also could operate with sets of solutions being outcomes of multiple objectives.
- Sequential design (Bayesian optimization, Evolutionary algorithm..) Bayesian methods differ from random or grid search in that they use past evaluation results to extrapolate and choose the next values to evaluate. Limit expensive evaluations of the objective function by choosing the next input values based on those that have done well in the past.

Optimization cost of black-box:

- Evaluation may be very expensive
- Sampling budget is unknown
- Possibly noisy objectives
- Feasibility constraints
- Multi-objectivity

Ideally, we want a method that can explore the search space while also limiting evaluations of hyperparameter choices. The single criterion in parameter tuning may not be sufficient to correctly characterize the behaviour of the configuration space that is why multiple criteria have to be considered. One way to clarify the task of understanding the space of possible solutions is to focus on the non-dominated frontier or Pareto-front, the subset of solutions that are not worse than any other but better on at least one goal. The difficulty here is that even the Pareto frontier can be too large to understand.

Multi-objective optimization

Parameter tuning is present in our daily life and comes in a variety of states. The goal is the rich best possible objective by correctly choosing the system parameters. Common of optimization problems requires the simultaneous optimization of multiple, usually contradictory, objectives. These type of problems are termed as multiobjective optimization problems. The solution to such problems is a family of points, that placing on a Pareto front. Knowledge of the Pareto front allows visualizing appropriate decisions in terms of performance for each objective.

“Multi-objective optimization(MOO) deals with such conflicting objectives. It provides a mathematical framework to arrive at optimal design state which accommodates the various criteria demanded by the application. The process of optimizing systematically and simultaneously a collection of objective functions are called multi-objective optimization (MOO) [1]”.

For a multi-objective problem, we consider “solution” as points from parameter space that lead to non-dominated results in objective space. This set of points approximate real Pareto-front. Improving

“solution” means that sets of points coincide better with real Pareto-front. How to search for an optimal solution to the multi-objective optimization problem?

Scalarizing. Weighted sum methods

Scalarizing approach is built on the traditional techniques to creating an alternative problem with a single, composite objective function. Single objective optimization techniques are then applied to this composite function to obtain a single optimal solution. The weighted-sum methods it’s a well known type of scalarizing technic is applied to simplify a multiobjective problem. Concatenate the objectives into one criterion by using magic weighted sum factors. The merged objective is used to evaluate and define the optimal solution. Weighted sum methods have difficulties in selecting proper weight especially when there is no connected a priori knowledge among objectives. Furthermore, Uniform distribution points in parameters space don’t generate uniform distribution points on objective space. This means that we can’t approximate Pareto-front completely even with multiple optimization rounds. Some scalarizing technics try to improve exploration of parameter space by assigning more “intelligence” aggregation to the objectives. Such solutions may be fragile. They change dramatically if we modify algorithm parameters.

Moreover, the weighting method can not provide a solution among underparts of the Pareto surface due to “duality gap” for not convex cases. Even for convex cases, for example, in linear cases, even if we want to get a point in the middle of a line segment between two points, we hardly get a peak of Pareto surface, as long as the well-known simplex method is used. This implies that depending on the structure of the problem, the linearly weighted sum can not necessarily provide a solution as DM desires. [2]

Multi-Objective Evolutionary Algorithms

Generating the Pareto set can be computationally expensive and is often infeasible because the complexity of the underlying volume limits exact techniques from being applicable. For this reason, a number of stochastic search strategies such as evolutionary algorithms, tabu search, simulated annealing, and ant colony optimization have been developed: they usually do not guarantee to identify optimal trade-offs but try to find a good approximation, i.e., a set of solutions whose objective vectors are (hopefully) not too far away from the optimal objective vectors [3].

The evolutionary algorithm (EA) form a class of heuristic search methods that simulate the process of natural evolution. Using simplifications, this EA is subsequently determined by the two basic principles: selection and variation. While selection imitates the competition for reproduction and resources among living beings, the other principle, variation, imitates the natural ability to create ”new” living beings through recombination and mutation. Evolutionary algorithm possesses several characteristics that are desirable for problems including multiple conflicting objectives, and large and complicated search spaces. However, EA still need many evaluations of the “black box” system to solve a common multi-objective problem. This is further complicated by the fact that many such problems are very expensive. Consolidated, this makes EAs unfeasible for costly and Multy-objective problem. A good solution is the integration of the surrogate model which extrapolate and approximate the fitness landscape from samples. Multi-objective Evolutionary Algorithms (MOEAs) use this surrogate model as a target for optimization. Assumed that solution from surrogate nearby to a global optimum. The goal of this thesis is to understand if the performance of MOEAs approach can be improved by using compositional surrogates. The key idea of compositional surrogates is the splitting objective space to multiple surrogates that extrapolate it independently.Combination of multiple hypotheses should give them the potential to approximate more complicated problems. This approach avoids the idea of a single surrogate model, preferring instead to use the composition hypothesis to split out the terrain of objective space.

The multiple surrogates are analysed on objectives with various complexity, beside the simple and complicated unimodal structure. Generating a cloud of candidates is computationally expensive.

Evolutionary optimizers explore populations of candidate solutions in each generation, some mutator can make changes to the current population. A select operator then picks the best mutants which are then combined in some way to become generation i+1. This century, there has been much new work on multi-objective evolutionary algorithms with two or three objectives (as well as many-objective optimization, with many more objectives). Multi-objective Evolutionary Algorithms (MOEAs) are popular tools to solve optimization problems, because of their applicability to complex fitness landscapes and solid performance on problems with large design spaces. While other methods also exist, in this thesis we will focus on improving approaches with Evolutionary Algorithms for the Multy-objective optimizations. This search-based software engineering is a rapidly expanding area of research and a full survey of that work is beyond the scope of this thesis.

Metrics for multi-objective solution

In single-objective minimization, the quality of a given solution is trivial to quantify: the smaller the objective function value, the better. However, evaluating the quality of an approximation of a Pareto set is non trivial. The question is important for the comparison of algorithms or prediction next configuration.

According to [4], a Pareto front approximation should satisfy the following:

- The distance between the Pareto front and its approximation should be minimized.
- A heigh distribution of the non-dominated points is desirable.
- The range of the approximated front should be maximized, i.e., for each objective, a wide range of values should be covered by the non-dominated points.

Metrics for performance indicators partitioned into four groups according to their properties [5]:

- cardinality
- convergence
- distribution
- spread

Base on the right metrics general multi-objective algorithm keep making progress toward the Pareto front in the objective function space. The goal of optimizing a multi-objective problem is to obtain an approximation solution set to the reference Pareto front, including the following subgoals:

- All solution set are as close as possible to the Pareto front
- All solution set are as diverse as possible in the objective space
- Evaluate as few solution as possible

Straightforward applying of the simple coefficient of determination (R2) is the wrong indicator of success. Evaluations of different sets of Pareto optimal points is multi-objective task. The necessary objectives follow for improving solutions:

- Keep hypervolume low. Reference point is 0 for all objectives.
- Maximize sparsity of points. Average distance. Crowding Distance. Spacing metrics.
- Maximize non-dominant decisions in the total population

Also distribution and spread indicators is consider in this work. According to [6], “the spread metrics try to measure the extents of the spread achieved in a computed Pareto front approximation”. They are not useful to evaluate the convergence of an algorithm, or at comparing algorithms. They only make sense when the Pareto set is composed of several solutions.

For multi-objective optimization (MOO), an algorithm should provide a set of solutions that realize the optimal trade-offs between the considered optimization objectives, i.e., Pareto set. Therefore, the performance comparison of MOO algorithms is based on their Pareto sets. In this study, three popular metrics are used to quantify the performance of the algorithms.

- Hypervolume (HV)[7]. This metric represents the volume of the objective space that is covered by the individuals of a non-dominated solutions set (solutions that belong to a Pareto front). The volume is delimited by two points: one point that is called the anti-optimal point (A) that is defined as the worst solution inside the objective space, and a second optimal point (pseudo-optimal) that is calculated by the proposed solution method. Determining the hypervolume indicator is a computationally expensive task. Even in case of a reasonably small dimension and low number of points (e.g. 100 points in 10 dimensions), there are currently no known algorithms that can yield the results fast enough for use in most multiple-objective optimizers
- Non-dominated Ratio (NDR). This metric employs the non-dominated count of a solution set divided by the total size of solution set. Higher values are preferred to lower ones.
- Spacing [8]. Describe the distribution of Pareto points. Fewer space metrics means better coverage of objectives values range.

Conclusion

For optimization expensive black-box:

- Scalable algorithms that convert multi-objective to single objective problem produce solution that not accurate enough(Scalarizing). Also this approach suitable for a limited type of problem. Also< there are a lot important parameters that significant influence on algorithm performance.
- Genetic algorithms. This approach is costly to perform and not appropriate for expensive problems.

Optimization gap in obtaining high quality, multi/single-obj solutions in expensive to evaluate experiments. Experiments as a black box, derivative-free. Reference to surrogate optimization.

Surrogate optimization

[9]

To dealing with expensive optimization problem more quickly, we can use surrogate models in the optimization process to approximate the objective functions of the problem. Approximation of solution is faster than the whole optimization process can be accelerated. Nevertheless, the extra time needed to build and update the surrogate models during the optimization process. In the case of pre-selecting

the promising individuals, the surrogate model is used to find the likely or drop the low-quality individuals even before they are exactly evaluated, thus reducing the number of exact evaluations.

In the literature, the term surrogate or model-based optimization is used where, during the optimization processes, some solutions are not evaluated with the original objective function, but are approximated using a model of this function. Different approximation methods are used to build surrogate models. For single and multiobjective optimization similar methods are used. These techniques typically return only one approximated value, which is why in multiobjective problems several models have to be used, so that each model approximates one objective. Some of the most commonly used methods are the Response Surface Method [10], Radial Basis Function [11], Neural Network, Kriging [12] and Gaussian Process Modeling [13,14].

General classification [15]: Within surrogate-model-based optimization algorithms, a mechanism is needed to find a balance between the exact and approximate evaluations. In evolutionary algorithms, this mechanism is called evolution control [16] and can be either fixed or adaptive. In fixed evolution control the number of exact function evaluations that will be performed during the optimization is known in advance. Fixed evolution control can be further divided into generation-based control, where in some generations all solutions are approximated and in the others, they are exactly evaluated [17], and individual based control, where in every generation some (usually the best) solutions are exactly evaluated and others approximated [18]. In adaptive evolution control, the number of exactly evaluated solutions is not known in advance but depends on the accuracy of the model for the given problem. Adaptive evolution control can be used in one of two ways: as a part of a memetic search or to pre-select the promising individuals which are then exactly evaluated [19].

Surrogate used to expedite search for global optimum. Global accuracy of surrogate not a priority. Surrogate model is cheaper to evaluate than the objective.

Bayesian optimization (BO) methods often rely on the assumption that the objective function is well-behaved, but in practice, the objective functions are seldom well-behaved even if noise-free observations can be collected. In [20] propose robust surrogate models to address the issue by focusing on the well- behaved structure informative for search while ignoring detrimental structure that is challenging to model data efficiently.

Surrogate-model-based MOEA

In [21] proposed approaches that apply kind of surrogate assistant to evaluations and ranging new population. It allows detecting the most informative examples in population and evaluates them. Identifies and evaluates just those most informative examples at the end done fewer evaluations of the real system. Another way to explore solutions is to apply some heuristic to decompose the total space into many smaller problems, and then use a simpler optimizer for each region.

Surrogates are also used to rank and filter out offspring according to Pareto-related indicators like the hypervolume [22], or a weighted sum of the objectives [23]. The problem with the methods that use hypervolume as a way of finding promising solutions is the calculation time needed to calculate the hypervolume, especially on many objectives. Another possibility is described in [24], where the authors present an algorithm that calculates only non-dominated solutions or solutions that can, because of variance, become non-dominated.

GP-DEMO [15] The algorithm is based on the newly defined relations for comparing solutions under uncertainty. These relations minimize the possibility of wrongly performed comparisons of solutions due to inaccurate surrogate model approximations. Using this confidence interval, we define new dominance relations that take into account this uncertainty and propose a new concept for comparing solutions under uncertainty that requires exact evaluations only in cases where more certainty is needed.

Surrogate with MOEA

Kind of extending the search stage of MOEA with surrogate to simulate evaluation of population. It transform the problem of searching a new better population to improving general hypothesis of how and where Pareto set presented.

In surrogate-model-based multiobjective optimization, approximated values are often mistakenly used in the solution comparison. As a consequence, exactly evaluated good solutions can be discarded from the population because they appear to be dominated by the inaccurate and over-optimistic approximations. This can slow the optimization process or even prevent the algorithm from finding the best solutions [15].

Discussion

Example of each type of optimization. Justification solution. Conclusion: Design gap in optimization/parameter tuning. Need to indicate optimization workflow for expensive process/experiments. The argument(s) why we need a new architecture. Reference to composition architecture.

Surrogate based optimization has proven effective in many aspects of engineering and in applications where data is “expensive”, or difficult, to evaluate.

Compositional architecture

We could describe compositional-based surrogate optimization as compound grey-box system whit a lot of open research areas where surrogate should improve, managing portfolio, compare of predictions Pareto fronts. As a developer, you can be focused on a specific problem and don’t know how to implement other components. This is one of the main advantages of the described approach.

Compositional surrogates

Can the same single-objective models be equally applied to various types of problems in multi-/single-objective optimization? When there is no correlation between the objectives, a very simple way to solve this kind of problem is to build independent models, i.e. one for each objective, and then to use those models to simultaneously extrapolate possible solutions with MOEA. Nevertheless, the output values correlated, but an often naive way to build multiple models that able to extrapolate complex objective space is often given good results.

Later research generalized this approach. MOEA/D (multiobjective evolutionary algorithm based on decomposition [25]) is a generic framework that decomposes a multi-objective optimization problem into many smaller single problems, then applies a second optimizer to each smaller subproblem, simultaneously.

With multiple models, their flaws can combine, as well as the time required to build the models. In memetic algorithms, especially if the surrogate model is not very accurate, a local optimum can be found instead of the global optimum. But in terms of parameter tuning, this point should be better than a predefined sampling plan. Evaluation of this prediction improve surrogate model quality in the near-optimal area and improve prediction in the next round. For example, OEGADO [26] creates a surrogate model for each of the objectives. The best solutions in every objective get also approximated on other objectives, which helps with finding trade-off individuals. The best individuals are then exactly evaluated and used to update the models.

Scope of work

Describe and implement workflow for multi-objective parameter tuning of the derivative-free, black-box system. Parameter estimation is costly. The proposed solutions are also suitable for single-criteria optimization. Problem Setting.

Goal:

1. Globally optimize an objective function(s) that is expensive to evaluate. Single/Multi-objective parameter tuning
2. Simultaneously optimization scalable objectives
3. Components reuse. Extensibility with other frameworks

Problem:

1. A large number of the target black-box evaluations
2. Interfaces not unify
3. Code duplication

Solution:

1. Component-Based Architecture
2. Compositional-based surrogate optimization with MOEA

Concept

Into

1. Classification approaches with surrogate model
2. Define problems that related with surrogates model
3. Ideas on how to solve it
4. Discussion

In this section general applicability of surrogate model presented. With different applicability use cases, there are some obstacles occurred that have a place to be in all circumstances.

General problem trade-off is producing the best possible multi-objective solution with less effort. Because we consider expensive to the evaluation system, an effort first of all means really evaluated examples. Each of this evaluation can require a lot of time, energy or other resources. That is why the main comparison criteria for approach in solving the multi-objective problem is reducing the count of experiments. Moving further is a question: What does it mean, solving a multi-obj problem?

- Single-point, that is Pareto optimal solution
- Multi-point solutions that all is Pareto optimal

In this work under solving a multi-objective problem, we intend to find a set of none-dominated points that cover a wide range of objectives values and close to true Pareto front as possible. For reach, this goal multiple algorithms could be applied but MOEA is favoured choice. The advantage of evolutionary algorithms is that it could be easily modified. It operate on a set of solution candidates, that are well-fitted to approximate Pareto-front. Finally, it has been demonstrated in various usecases that evolutionary algorithms can estimate highly complex problems. With constrains as expensive evaluations, the real count of experiments could be reduced throw applying a multi-objective algorithm on a surrogate model. This technique is the preferred choice for functional optimization when the evaluation cost is large.

As shown before for a black-box expensive problem suited surrogate or model-based optimization. But this approach also has open questions and limitations:

- Multi-objectivity. They are not too many surrogate models that can handle multi-dimensional objective space. Also scaling existing multi-output surrogate model is an open research question

- Surrogate is domain-specific. For improving and reach the best of the best prediction we should know in cross-grain basic objective surface to apply certain surrogate. Universal surrogates can gain optimal results but not the best possible

- Quality of prediction depends on how much samples do we have for a specific type of surrogate. There is additionally trade-off between reducing samples size and maximize the quality of prediction.

- Categorical features. A lot of real-world problems depends on the categorical feature. Parameter tuning with that type of space is not trivial

- Often Optimization algorithm and surrogate model are very tightly coupled. Highly united implementation can reduce the search possibility of algorithm and suitable for a specific type of problem. Reimplementing these algorithms for each usage scenario becomes timeconsuming and error-prone.

Mainly two groups are affected by this problem:

- Application engineers who need to choose, implement, and apply state-of-the-art algorithms without in-depth programming knowledge and expertise in the optimization domain

- Developers of optimization methods who want to evaluate algorithms on different test problems and compare a variety of competing methods

For slow computational problems, it would be useful to modulate a problem using a quite small number of most informative examples. This general topic introduces compositional surrogate, as a proxy model that approximate objectives surfaces and support MOEA to evaluates near a multi-objective solution and predict better multi-objective samples on each iteration.

There is a clear need for a method to provide and distribute ready-to-use implementations of optimization methods and ready-to-use benchmark and real-world problems. These modules should be freely combinable. Since the above- mentioned issues are not constrained to evolutionary optimization a candidate surrogates solution should be applicable to a broader range of search algorithms.

The main objective of this part is to provide a thorough treatment of multi-objective parameter tuning with evolutionary algorithm(s)

Key description how to improve solutions for problems in research questions.

Multi-objective optimizations are frequently encountered in engineering practices. The solution techniques and parametric selections however are usually problem-specific. [27]

Compositional Surrogate Model

The potential for applying surrogate is laid in the fast evaluation of the surrogate model. This advantage should outperform disadvantage in time required to build this surrogate model. In classical model-based optimization is used single surrogate-model that provide a hypothesis on the relation between parameter and objective space. There is a lot type of models that can do it but out and away fewer models that can manage multidimensionality objective space. The perspective way to create multi-objective surrogate is stacking multiple simple models into one that describes complex objective space. Notwithstanding that those models could be completely different and build in parallel, they still related because fitted on intersection features. Splitting optimization problem to multiple stages improves the reusability of code and makes approach scalable. Nevertheless, we can switch from single-obj to multi obj and change optimization technic on the fly.

A surrogate model is either selected randomly or due to its popularity in the area with which the problem is associated. However, there are still some open challenges related to the ensemble of meta-models such as what should be the criterion for choosing different metamodels or how different metamodels can be used simultaneously? In addition, there are no guidelines for using different models for different objective functions [28].

Domain-specific problem

With gain to find the best solution with less effort surrogate models is domain-specific. It's mean that from two surrogate models in two different problems the best surrogate is changing. It could interpreter as Non-free lunch theorem in model-based optimization. If we extend this argument then the same optimization problem in different parameter tuning iteration could be interpreted as another optimization problem. This means that to reduce effort and increase the convergence of an algorithm we should change the surrogate model depend on how much samples do we have. As one would expect, no approximation method is universal. This leads us to use a portfolio with surrogate models. As a negative consequence, the model fitting additional introduces an additional overhead into the optimization.

Surrogate Validation

It is necessary to sacrifice a small portion of the data experiments to check the quality of the surrogate model on it. Based on validation results we can discard inadequate models and in a complex manner evaluated quality of solutions from valid models. If neither model is valid, this means that the best solution right now is a random solution. This may be due to insufficiently count of samples for the model or incorrect surrogate model, which is unable to describe the hypothesis between the configuration and objective space. In the context of parameter tuning a common misconception is that we are interested in the accuracy of the model not in the entire space but in the optimal region. That is why evaluation surrogate validity based only on the coefficient of determination(R2) metrics is incorrect. Global R2 can be anyway used as a threshold in which if the models score becomes smaller of some threshold value it is not valid even without further estimations.

Real parameters for optimization

Parameter tuning is not a trivial task. Coding features for a surrogate model can transform es in mining full form, understandable for a surrogate model. Encoding just represents a feature in another form that help model instanciation relation and a correlation between features. Based on this inner interpretation model can predict the values of labels based on a parameter vector. The problem occurs what we use the model as black-box to find a minimum of prediction and make reverse interpretation of this optimal predictions in the context of parameter space. Offen this prediction is not a feasible point from parameters space. As a possible solution, it partly transforms problem in multiple classification tasks and then considers there relation as a continuous problem for optimization [TPE]. For general surrogate model with mixed parameters, we can solve problem orthogonally with multiple optimizers. All categorical feature are encoded and with other numerical parameters were fitted to the surrogate model. Then we use this model as a source that describes the function of the mapping parameter to objective space. We incrementally solve this optimization problem on a subset of dimension iteratively. For example, we find optimal parameters in categorical dimensions and then fix these values and optimize surrogate on the left sub set of numerical dimensions where categories are fixed.

Discussion

Infill criteria In the case of MOEA, solution of algorithm present as non-dominated final population. Based on unbiased, multi-objective criteria, they all uniformly could be presented as a prediction to the next evaluation. They represents current solution based on the surrogate model. Nevertheless, there is prior knowledge available in samples which can be taken into account. To reduce the number of candidates in the population, it is possible to deny those in which the distance to the nearest available sample is less than their average distance. So there are two strategies for predicting from a population:

- Prior and posterior knowledge. Based on changing metrics in available and proposed solutions
- Posterior knowledge. Proposed solutions are all equal

Also, to the best of our knowledge, has not been previously or stingy reported in the efficient multi-objective optimization. Contribution:

- Surrogate combination/composition. Current approaches use the same model for each dimensionality
- Surrogate portfolio. Search a better hypothesis for a specific problem at a particular stage of parameter tuning
- Metric combination for evaluation Pareto optimal points
- Samples size depends on model(s) validity
- Combination of different(orthogonal) solvers

Some of the major findings were [28]:

1. Kriging and neural networks were the most commonly used surrogate models
2. Most of the algorithms were based on dominance-based evolutionary algorithms
3. Most of the algorithms solved the problems with no more than three objectives
4. The number of decision variables was also limited especially when using Kriging

5. Only few algorithms used an ensemble of metamodels

6. Many algorithms were tested only on benchmark problems which were not at all computationally expensive

Implementation. Development

Without automated tools, it can take days for experts to review just a few dozen examples. In that same time, an automatic tool can explore thousands to millions to billions more solutions. People find it an overwhelming task just to certify the correctness of conclusions generated from so many results.

Separation of concerns

Managing complex execution Strategies

Variants in the evaluation of sets of solutions for each hypothesis. Each hypothesis has quality metrics. Solution(s) from each hypothesis have also own metrics.

There are main approaches how produce single solution:

- Solution from best hypothesis. Sorting
- Bagging solution
- Voting solution

Designing a Sampling Plan

- The most straightforward way of sampling a design space in a uniform fashion is by [9] means of a rectangular grid of points. This is the full factorial sampling technique referred - Latin Squares

Random sampling has the downside that for small sample sizes, there is often signficant clustering of samples, which is not ideal for interpolation since clustered samples can be wasteful. Instead, often a better option is to use a Latin hypercube, which enforces a condition that sample bins may not share the same coordinates for any coordinate axis

Dependencies

Adapted to provide base implementation for stages in parameter tuning with multi-objective

Pagmo2

A Python platform [29] to perform parallel computations of optimisation tasks (global and local) via the asynchronous generalized island model. All test suites and basic multi-objective solvers:

Realization of main MOEA:

- NSGA2. Non-dominated Sorting Genetic Algorithm
- MOEA/D. Multi Objective Evolutionary Algorithms by Decomposition (the DE variant)
- MACO. Multi-objective Ant Colony Optimizer.

- NSPSO.

Tests suits:

- ZDT [4] is 6 different two-objective scalable problems all beginning from a combination of functions allowing, to measure the distance of any point to the Pareto front while creating problems.

- WFG [30] was conceived to exceed the functionalities of previously implemented test suites. In particular, non-separable problems, deceptive problems, truly degenerative problems and mixed shape Pareto front problems are thorougly covered, as well as scalable problems in both the number of objectives and variables. Also, problems with dependencies between position and distance related parameters are covered. In their paper the authors identify the need for nonseparable multimodal problems to test multi-objective optimization algorithms. Given this, they propose a set of 9 different scalable multi-objective unconstrained problems.

- DTLZ [31]. All problems in this test suite are box-constrained continuous n-dimensional multi-objective problems, scalable in fitness dimension.

Reusability in parameter tuning

Parameter tuning can be splitted down into steps that are common for the many/single-objective optimizations. Each step in optimization workflow has variability via implemented interfaces. Single-objective hypotheses can be combined for multi-objective optimization with compositional design.

API of metric-learn is compatible with scikit-learn, the leading library for machine learning in Python. This allows to use all the scikit-learn routines (for pipelining, model selection, etc) with metric learning algorithms through a unified interface.

Surrogate hypothesis portfolio

A Surrogate(s) is a simplified hypothesis of the relation between parameters and objectives space build on examples. The simplifications are mean to discard the superfluous details that are unlikely to generalize to new instances. However, to decide what data to discard and what data to keep, you must make a hypothesis. For example, a linear model makes the hypothesis that the data is fundamentally linear and that the distance between the instances and the straight line is just noise, which can safely be ignored.

If there is no hypothesis about the data, then there is no reason to prefer one surrogate over any other. For some datasets, the best model is a linear model, while for other datasets it is a neural network. No model is a priori guaranteed to work better, this is consequences from the No Free Lunch (NFL) theorem. The only way to know for sure which model is best is to evaluate them all. Since this is not possible, in practice you make some reasonable assumptions about the data and you evaluate only a few reasonable models. For example, for simple tasks, you may evaluate linear models with various levels of regularization, and for a complex problem, you may evaluate various neural networks.

“No Free Lunch” (NFL) theorems demonstrate that if an algorithm performs well on a certain class of problems then it necessarily pays for that with degraded performance on the set of all remaining problems Additionally, the name emphasizes the parallel with similar results in supervised learning.

1. You have to try multiple types of surrogate(models) to find the best one for your data.

2. A number of NFL theorems were derived that demonstrate the danger of comparing algorithms by their performance on a small sample of problems.

As metamodel-based algorithms are generally developed for black box problems, where characteristics of the problems to be solved are not known a priori, one can measure the efficiency of an algorithm by its ability to provide meaning- ful solutions in a least number of function evaluations [28].

A set of models is defined that can form a partial or complete hypothesis to describe the problem. Also during the increase of the experiments may change the model that best describes the existing problem As a result, there is variability for each problem and configuration step at the same time. A set of hypotheses can solve this problem but it takes longer time for cross validation.

Inner interfaces

Supervised learning consists in learning the link between two datasets: the observed data X and an external variable y that we are trying to predict, usually called target or labels. Most often, y is a 1D array of length $n_samples$. All supervised estimators in scikit-learn implement a fit(X, y) method to fit the model and a predict(X) method that, given unlabeled observations X, returns the predicted labels y.

Using arbitrary regression models from scikit-learn as surrogates

Problem that each optimization framework/library use inner interfaces. It is necessary to define a standard that implements best practices for extension libraries [32]. We introduce new Model-based line for parameter tuning.

Validate hypothesis

The main task of learning algorithms is to be able to generalize to unseen data. Surrogate model as learning model should generalize examples to valid hypothesis. Since we cannot immediately check the surrogate performance on new, incoming data, it is necessary to sacrifice a small portion of the examples to check the quality of the model on it. n case if surrogate model have enoughs score (pass metrics threshold) we consider it valid and could be processed as subject for inference(prediction).

Sampling strategy

Oversampling and undersampling in data analysis. Alleviate imbalance in the dataset. Imbalance in dataset is not always a problem, more so for optimization tasks.

The main gain for models not to provide best accuracy on all search space but provide possible optimum regions. Accuracy in prediction optimal regions or points from there will direct the search in the right direction.

Predictor variables can legitimately over- or under-sample. In this case, provided a carefully check that the model assumptions seem valid.

for other set of parameters, and make a choice from more diverse pool of models.

Results

[ref: Multi-Objective Parameter Configuration of Machine Learning Algorithms using Model-Based Optimization] The approach is linked to the field of surrogate assisted optimizations. In many practical settings only a restricted budget is spendable. For example, the arise of Big Data confronts many

machine learning techniques with new expensive parameter configuration problems. A single training of a Support Vector Machine (SVM) on a data-set containing less than a million observations can take several hours.

Evaluation. Experimental Results

This chapter presents the evaluation of the proposed method on test problems with diverse objective landscape and with a various number of search variables.

MOEA is called globally convergent if the produced, non-dominated population converges to the true Pareto front while the number of generations goes to infinity.

Questions to find out:

- Advantages and disadvantages of the model-based optimization over the classic MOEA
- Model-based optimization (MBO): compositional vs single surrogate
- MBO: several identical surrogate vs surrogate portfolio
- Infill criteria: Selection a point from Pareto-front approximated population. Prior vs Posterior
- Efficacy in handling problems having more than two objectives

[33]

Benchmark problems

For comparison was selected several widespread synthetic benchmark suites. All of them are scalable in parameters space and some in objective space also. They simulate real life problem and have main related chalanges such as multi-modality, different surface type,not uniform search space and etsetra.

ZDT

This widespread test suite[4] was conceived for two-objective problems and takes its name from its authors Zitzler, Deb and Thiele. Each test function involves a particular feature that is known to cause difficulty in the evolutionary optimization process, mainly in converging to the Pareto-optimal front (e.g., multimodality and deception).

- ZDT1: function has a convex Pareto-optimal front
- ZDT2: function has a non-convex Pareto-optimal front
- ZDT3: function adds a discreteness feature to the front. Its Pareto-optimal front consists of several noncontiguous convex parts. The introduction of a sine function in this objective function causes discontinuities in the Pareto-optimal front, but not in the parameter space.
- ZDT4: function has 21 local Pareto-optimal fronts and therefore is highly multi-modal
- ZDT5: integer problem

- ZDT6: function has a non-uniform search space: the Pareto-optimal solutions are non-uniformly distributed along the global Pareto front, and also the density of the solutions is lowest near the Pareto optimal front and highest away from the front

In their paper the authors propose a set of 6 different scalable problems all originating from a well thought combination of functions allowing, by construction, to measure the distance of any point to the Pareto front

DTLZ

This benchmark suite[31] was conceived for multiobjective problems with scalable fitness and objective dimensions and takes its name from its authors Deb, Thiele, Laumanns and Zitzler. All problems in this test suite are box-constrained continuous n-dimensional multi-objective problems, scalable in fitness dimension.

- DTLZ1: The optimal pareto front lies on a linear hyperplane
- DTLZ2: The search space is continous, unimodal and the problem is not deceptive
- DTLZ3: The search space is continous, unimodal and the problem is not deceptive. It is supposed to be harder to converge towards the optimal pareto front than DTLZ2
- DTLZ4: The search space contains a dense area of solutions next to the plane
- DTLZ5: This problem will test an MOEA's ability to converge to a cruve and will also allow an easier way to visually demonstrate (just by plotting f_M with any other objective function) the performance of an MOEA. Since there is a natural bias for solutions close to this Pareto-optimal curve, this problem may be easy for an algorithmn to solve. Because of its simplicity its recommended to use a higher number of objectives
- DTLZ6: A more difficult version of the DTLZ5 problem with the non-linear distance function g makes it harder to convergence against the pareto optimal curve
- DTLZ7: This problem has disconnected Pareto-optimal regions in the search space

WFG

This test suite [30] was conceived to exceed the functionalities of previously implemented test suites. In particular, non-separable problems, deceptive problems, truly degenerative problems and mixed shape Pareto front problems are thoroughly covered, as well as scalable problems in both the number of objectives and variables. Also, problems with dependencies between position and distance related parameters are covered. The WFG test suite was introduced by Simon Huband, Luigi Barone, Lyndon While, and Phil Hingston. All these problems, were developed satisfying the following guidelines:

1. A few unimodal test problems should be present in the test suite. Various Pareto optimal geometries and bias conditions should define these problems, in order to test how the convergence velocity is influenced by these aspects.
2. The following three Pareto optimal geometries should be present in the test suite: degenerate Pareto optimal fronts, disconnected Pareto optimal fronts and disconnected Pareto optimal sets
3. Many problems should be multimodal, and a few deceptive problems should also be covered
4. The majority of test problems should be non-separable

5. Both non-separable and multimodal problems should also be addressed

- WFG1: This problems skews the relative significance of different parameters by employing different weights in the weighted sum reduction. Also, this problem is unimodal and with a convex and mixed Pareto optimal geometry

- WFG2: This problem is non-separable, unimodal and with a convex and disconnected Pareto optimal geometry

- WFG3: This is a non-separable, unimodal problem in all its objective except for the last one, which is multimodal

- WFG4: This is a separable, multimodal problem with a concave Pareto optimal geometry. The multimodality of this problem has larger “hill sizes” than that of WFG9: this makes it thus more difficult.

- WFG5: This is a deceptive, separable problem with a concave Pareto optimal geometry.

- WFG6: This problem is non-separable and unimodal. Its Pareto optimal geometry is concave. The non-separable reduction of this problem makes it more difficult than that of WFG2 and WFG3

- WFG7: This problem is separable, unimodal and with a concave Pareto optimal geometry. This, together with WFG1, is the only problem that is both separable and unimodal.

- WFG8: This is a non-separable, unimodal problem with a concave Pareto optimal geometry

- WFG9: This is a multimodal, deceptive and non-separable problem with a concave Pareto optimal geometry. Similar to WFG6, the non-separable reduction of this problem makes it more difficult than that of WFG2 and WFG3. Also, this problem is only deceptive on its position parameters.

Conclusion

The quality of the results obtained with X was similar to the results obtained with Y, but with significantly fewer exactly evaluated solutions during the optimization process.

Neuroevolution of augmenting topologies

*Training Neural Networks (especially deep ones) is hard and has many issues (non-convex cost functions - local minima, vanishing and exploding gradients etc.).

Training Neural Networks (NNs) with Genetic Algorithms (GAs) is not only feasible, there are some niche areas where the performance is good enough to be used frequently. A good example of this is Neuroevolution of augmenting topologies or NEAT, which a successful approach to generating controllers in simple environments, such as games.

In the more general case though, the approach does not scale well to large, deep networks with many parameters to tune.

Genetic algorithms and other global searches for optimal parameters are robust in ways that gradient-based algorithms are not. For instance, you could train a NN with step function activations, or any other non-differentiable activation functions. They have weaknesses elsewhere. One thing relevant in the case of GAs used for NNs, is that weight parameters are interchangeable in some combinations but heavily co-dependent in other combinations. Merging two equally good neural networks with different

parameters - which you would do in cross-over in a GA - will usually result in a third network with poor performance. NEAT's success is partially in finding a way to address that issue by "growing" the NN's connections and matching them up between similar neural networks.

Gradient-based approaches are much more efficient. In general, and not just in domain of NNs, if you can calculate gradient of a function with respect to parameters, then you can find optimal parameters faster than most other optimising techniques. An accurate gradient guarantees at least a small improvement from a single evaluation, and most other optimisers fall into a generate-and-retry paradigm which cannot make that kind of guarantee. The weakness of tending to find local optima has turned out not be a major hindrance for the loss functions in NNs, and has been tackled with some degree of success using extensions to basic gradient descent such as momentum, RPROP, Adam etc.

In practice on a large multi-layer network, gradient methods are likely orders of magnitude faster than GA searches such as NEAT for finding network parameters. You won't find any GA-trained CNNs that solve ImageNet, or even MNIST, where the GA has found the network weights unaided. However, GAs, or at least some variants of them, are not 100'%' ruled out. For instance this 2017 blog reviews recent papers including Large-Scale Evolution of Image Classifiers which explores using GAs to discover NN hyperparameters which is an important task in machine learning, and not very tractable using gradient-based methods.

Related work

Many existing approaches can be categorized as multi-objective optimization. That is why introduce comparison criteria for a clear and concise demarcation of the approach presented in this thesis:

Comparison Criteria for Related Work.

- Variability. Exchange surrogate, solver and sampling algorithms as components. Variants on each optimization workflow step.
- Scalability. Extend single-objective problem on the fly to multi-objective.
- Adaptation. Surrogate portfolio.
- From 0 to hero. Sampling plan depends on surrogate validity. The Sobol sequence (and Latin hypercube).

Important Features: Categorical variables, prior knowledge, multi-objective, feasibility constraints.

Dependencies

AUTO-SKLEARN

[34] - CASH (Combined Algorithm Selection and Hyperparameter optimization) problem

TPOT

[35] Already implemented TPOT automodel as hypothesis candidate

jMetalpy

[36] Partially implemented some solvers.

Hyperopt

raw:PlatEMO

[37] raw:PlatEMO: A MATLAB Platform for Evolutionary Multi-Objective Optimization

mlrMBO: A Modular Framework for Model-Based Optimization of Expensive Black-Box Functions

ParEGO

ParEGO was enhanced to a multi-point proposal by increasing the number of weight vectors randomly drawn in each iteration. If N points are desired, cN (c > 1) weight vectors are selected.

Conclusion and Future Work

General Conclusion

BRISE

Modelbase line in parameter tuning for software Product Line for Parameter Tuning

Future Work

Prior knowledge. Transfer learning

What is already implemented and how it could be improved.

- Model portfolio selection and combination.
- Prior distribution of parameters. Bayesian kernels.
- Human in the loop. Reducing the search space.

References

1. **Review of multi-criteria optimization methods – theory and applications**
   Godwin Odu
   IOSR Journal of Engineering (2013-10)
   DOI: 10.9790/3021-031020114

2. **Multi-objective optimization and its engineering applications**
   Hirotaka Nakayama
   Practical approaches to multi-objective optimization, 7.-12. november 2004 (2005)
   http://drops.dagstuhl.de/opus/volltexte/2005/234

3. **A tutorial on multiobjective optimization: Fundamentals and evolutionary methods**
   Michael T. M. Emmerich, André H. Deutz
   Natural Computing (2018) https://doi.org/10.1007/s11047-018-9685-y
   DOI: 10.1007/s11047-018-9685-y

4. **Comparison of multiobjective evolutionary algorithms: Empirical results**
   Eckart Zitzler, Kalyanmoy Deb, Lothar Thiele
   Evolutionary Computation (2000) https://doi.org/10.1162/106365600568202
   DOI: 10.1162/106365600568202

5. **Performance indicators in multiobjective optimization**
   Charles Audet, Sébastien Le Digabel, Dominique Cartier, Jean Bigeon, Ludovic Salomon
   (2018)

6. **Direct multisearch for multiobjective optimization**
   A. L. Custódio, J. F. Aguilar Madeira, A. Ismael F. Vaz, Luís Nunes Vicente
   SIAM Journal on Optimization (2011) https://doi.org/10.1137/10079731X
   DOI: 10.1137/10079731x

7. **Comparison of multiobjective evolutionary algorithms: Empirical results**
   Eckart Zitzler, Kalyanmoy Deb, Lothar Thiele
   Evolutionary Computation (2000)

8. **Fault tolerant design using single and multicriteria genetic algorithm optimization.**
   Jason R. Schott
   (1995)

9. **Engineering design via surrogate modelling - a practical guide.**
   Alexander I. J. Forrester, Andras Sobester, Andy J. Keane
   Wiley (2008)
   ISBN: 978-0-470-06068-1

10. **Response surface methodology: Process and product in optimization using designed experiments**
    Raymond H. Myers, Douglas C. Montgomery
    John Wiley; Sons, Inc. (1995)
    ISBN: 0471581003

11. **Gaussian processes in machine learning**
    Carl Edward Rasmussen
    Advanced lectures on machine learning: ML summer schools 2003, canberra, australia, february 2 -

14, 2003, tübingen, germany, august 4 - 16, 2003, revised lectures (2004)
https://doi.org/10.1007/978-3-540-28650-9_4
DOI: 10.1007/978-3-540-28650-9_4 · ISBN: 978-3-540-28650-9

12. **Interpolation of spatial data: Some theory for kriging**
Roger Woodard
Technometrics (2000) https://doi.org/10.1080/00401706.2000.10485731
DOI: 10.1080/00401706.2000.10485731

13. **Gaussian processes for machine learning (GPML) toolbox**
Carl Edward Rasmussen, Hannes Nickisch
J. Mach. Learn. Res. (2010) http://portal.acm.org/citation.cfm?id=1953029

14. **Gaussian processes for machine learning**
Carl Edward Rasmussen, Christopher K. I. Williams
MIT Press (2006) http://www.worldcat.org/oclc/61285753
ISBN: 026218253X

15. **GP-DEMO: differential evolution for multiobjective optimization based on gaussian process models**
Miha Mlakar, Dejan Petelin, Tea Tusar, Bogdan Filipic
European Journal of Operational Research (2015) https://doi.org/10.1016/j.ejor.2014.04.011
DOI: 10.1016/j.ejor.2014.04.011

16. **A comprehensive survey of fitness approximation in evolutionary computation**
Yaochu Jin
Soft Comput. (2005) https://doi.org/10.1007/s00500-003-0328-5
DOI: 10.1007/s00500-003-0328-5

17. **An evolutionary multi-objective adaptive meta-modeling procedure using artificial neural networks**
Kalyanmoy Deb, Pawan K. S. Nain
Evolutionary computation in dynamic and uncertain environments (2007)
https://doi.org/10.1007/978-3-540-49774-5_13
DOI: 10.1007/978-3-540-49774-5_13

18. **Optimal sizing, geometrical and topological design using a genetic algorithm**
D. E. Grierson, W. H. Pak
Structural optimization (1993-09-01) https://doi.org/10.1007/BF01743506
DOI: 10.1007/bf01743506

19. **An evolutionary strategy for surrogate-based multiobjective optimization**
Martin Pilát, Roman Neruda
Proceedings of the IEEE congress on evolutionary computation, CEC 2012, brisbane, australia, june 10-15, 2012 (2012) https://doi.org/10.1109/CEC.2012.6256450
DOI: 10.1109/cec.2012.6256450

20. **Modulating surrogates for bayesian optimization**
Erik Bodin, Markus Kaiser, Ieva Kazlauskaite, Zhenwen Dai, Neill D. F. Campbell, Carl Henrik Ek (2019)

21. **GALE: geometric active learning for search-based software engineering**
Joseph Krall, Tim Menzies, Misty Davies

IEEE Trans. Software Eng. (2015) https://doi.org/10.1109/TSE.2015.2432024
DOI: 10.1109/tse.2015.2432024

22. **Single- and multiobjective evolutionary optimization assisted by gaussian random field metamodels**
Michael T. M. Emmerich, Kyriakos C. Giannakoglou, Boris Naujoks
IEEE Trans. Evolutionary Computation (2006) https://doi.org/10.1109/TEVC.2005.859463
DOI: 10.1109/tevc.2005.859463

23. **Practical solutions for multi-objective optimization: An application to system reliability design problems**
Heidi A. Taboada, Fatema Baheranwala, David W. Coit, Naruemon Wattanapongsakorn
Rel. Eng. and Sys. Safety (2007) https://doi.org/10.1016/j.ress.2006.04.014
DOI: 10.1016/j.ress.2006.04.014

24. **Improving multi-objective genetic algorithms with adaptive design of experiments and online metamodeling**
G. Li, M. Li, S. Azarm, S. Al Hashimi, T. Al Ameri, N. Al Qasas
Structural and Multidisciplinary Optimization (2009-02-01) https://doi.org/10.1007/s00158-008-0251-6
DOI: 10.1007/s00158-008-0251-6

25. **MOEA/D: A multiobjective evolutionary algorithm based on decomposition**
Qingfu Zhang, Hui Li
IEEE Trans. Evolutionary Computation (2007) https://doi.org/10.1109/TEVC.2007.892759
DOI: 10.1109/tevc.2007.892759

26. **Multiobjective GA optimization using reduced models**
Deepti Chafekar, Liang Shi, Khaled Rasheed, Jiang Xuan
IEEE Trans. Systems, Man, and Cybernetics, Part C (2005)
https://doi.org/10.1109/TSMCC.2004.841905
DOI: 10.1109/tsmcc.2004.841905

27. **A reinforcement learning hyper-heuristic in multi-objective single point search with application to structural fault identification**
Jiong Tang Pei Cao
CoRR (2018) http://arxiv.org/abs/1812.07958

28. **A survey on handling computationally expensive multiobjective optimization problems with evolutionary algorithms**
Tinkle Chugh, Karthik Sindhya, Jussi Hakanen, Kaisa Miettinen
Soft Computing (2017-12)
DOI: 10.1007/s00500-017-2965-0

29. **Esa/pagmo2: Pagmo 2.12.0**
Francesco Biscani, Dario Izzo, Wenzel Jakob, GiacomoAcciarini, Marcus Märtens, Micky C, Alessio Mereta, Cord Kaldemeyer, Sergey Lyskov, Giacomo Acciarini, … Andrea Mambrini
Zenodo (2019-12) https://doi.org/10.5281/zenodo.3582877
DOI: 10.5281/zenodo.3582877

30. **A review of multiobjective test problems and a scalable test problem toolkit**
S. Huband, P. Hingston, L. Barone, L. While
IEEE Transactions on Evolutionary Computation (2006-10)
DOI: 10.1109/tevc.2005.861417

31. **Scalable test problems for evolutionary multiobjective optimization**
Kalyanmoy Deb, Lothar Thiele, Marco Laumanns, Eckart Zitzler
Evolutionary multiobjective optimization (2005) https://doi.org/10.1007/1-84628-137-7_6
DOI: 10.1007/1-84628-137-7_6

32. **API design for machine learning software: Experiences from the scikit-learn project**
Lars Buitinck, Gilles Louppe, Mathieu Blondel, Fabian Pedregosa, Andreas Mueller, Olivier Grisel, Vlad Niculae, Peter Prettenhofer, Alexandre Gramfort, Jaques Grobler, … Gaël Varoquaux
(2013)

33. **Benchmarking crimes: An emerging threat in systems security**
Erik van der Kouwe, Dennis Andriesse, Herbert Bos, Cristiano Giuffrida, Gernot Heiser
(2018)

34. **Efficient and robust automated machine learning**
Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, Frank Hutter
Advances in neural information processing systems (2015)

35. **Evaluation of a tree-based pipeline optimization tool for automating data science**
Randal S. Olson, Nathan Bartley, Ryan J. Urbanowicz, Jason H. Moore
Proceedings of the genetic and evolutionary computation conference 2016 (2016)
http://doi.acm.org/10.1145/2908812.2908918
DOI: 10.1145/2908812.2908918 · ISBN: 978-1-4503-4206-3

36. **JMetalPy: A python framework for multi-objective optimization with metaheuristics**
Antonio Benitez-Hidalgo, Antonio J. Nebro, Jose Garcia-Nieto, Izaskun Oregi, Javier Del Ser
(2019)

37. **raw:PlatEMO: A MATLAB platform for evolutionary multi-objective optimization**
Ye Tian, Ran Cheng, Xingyi Zhang, Yaochu Jin
IEEE Computational Intelligence Magazine (2017)