Efficient tuning of automated machine learning pipelines

Nguyen, D.A.

CHAPTER 9

Conclusions

Automated Machine Learning (AutoML) has emerged as an effective approach to streamline the machine learning development process for real-world applications. This thesis provides a comprehensive exploration of AutoML and its various important aspects. The research conducted in this thesis encompasses several significant contributions to the field.

To conclude the thesis, in this chapter, we provide a summary of the content of this thesis. We begin by highlighting the main contributions made in this research and addressing the research questions posed. This is presented in Section 9.1, where we summarize the key findings and achievements of this thesis.

Section 9.2 is dedicated to discussing potential avenues for future research. We outline these potential directions and offer insights into how they can contribute to advancing the state-of-the-art in AutoML.

9.1 Summary

Chapter 1 begins with a brief introduction and motivation to Automated Machine Learning (AutoML). It is attempted to give fundamental formulations of optimization approaches to address the AutoML optimization problem (i.e., *HPO-based* and *ML pipeline optimization* approach). Furthermore, the chapter outlines the significant contributions of the thesis within the AutoML domain, highlights the specific research questions and provides an overview of each chapter's main contributions and methodologies. Lastly, this chapter also provides a roadmap for the reader and establishes the overall organization and structure of the thesis.

Chapter 2 discusses the life cycle of machine learning (ML) application development and provides a comprehensive literature review on key technical aspects of ML. It explores the different stages involved in the ML application development process, from data preparation, data preprocessing, ML pipeline

optimization, model evaluation and application development. The chapter also discusses the challenges and considerations specific to each stage, highlighting the importance of an efficient and automated approach to streamline the ML development process. Building upon the ML life cycle, the discussion in this chapter expands to encompass various functions and techniques offered by existing AutoML products. The chapter also examines the underlying techniques and methodologies employed by the existing AutoML products, shedding light on the potential implications for ML application development.

Chapter 3 focuses on providing a thorough literature review of AutoML optimization approaches, with a specific emphasis on adapting hyperparameter optimization (HPO) approaches to address the AutoML optimization problem. The chapter begins by presenting a comprehensive review of common black-box optimization approaches employed in AutoML, including Grid search, Random search, and Bayesian optimization. These approaches serve as the foundation for understanding the evolution and advancements in AutoML optimization. Follow up with a literature review of optimization algorithms of the two well-known multi-fidelity approaches – racing procedure and bandit learning.

Throughout the literature review, the chapter provides insights into the theoretical foundations, algorithmic frameworks, and practical implementations of these optimization approaches in the context of AutoML. This chapter aims to establish the theoretical background and set the stage for the original contributions and research conducted in the subsequent chapters of the thesis.

Chapter 4 provides two benchmark experiments repeatedly used in this thesis. Detailed information about the two experiments is introduced in this chapter, where each includes a search space, examined datasets and a detailed experiment procedure. The first benchmark experiment is designed for class-imbalanced problems, where a set of 44 binary class-imbalanced benchmark datasets and a suitable search space are provided. The datasets are taken from the Keel collection [186]. The search space includes 21 resampling techniques and five classification algorithms, where algorithms and their hyperparameters are selected based on related work recommendations. Moreover, this experimental design is successfully used in Chapters [5-8] that demonstrate its usefulness for imbalanced classification problems. This study also answers **RQ1** on how to handle class imbalance problems.

The second set of benchmark experiments is identical to [22], which includes 73 classification benchmark datasets from OpenML [189], and the search space

is extracted from Auto-sklearn [45]. This set of experiments is later used in the works of chapter 7 and chapter 8.

Chapter 5 investigates the effectiveness of the commonly used HPO optimization approaches – Random search and Bayesian optimization (BO), to solve the CASH problem for the binary class imbalance classification problems. This investigation is to answer **RQ2** in which approach is most effective for optimizing the ML pipeline in addressing class imbalance problems. Besides, we are particularly interested in how CASH improved classification performance compared to using static default hyperparameters (i.e., try all combinations of resampler and classifier without hyperparameter tuning). The key findings from this indicate the following. We observed that CASH optimization significantly improves classification performance compared to using static default hyperparameters. Moreover, the experimental results indicate that BO is always the best method found. This study concludes that BO outperforms other approaches in answering **RQ2**. Additionally, 98% of runs yield the best performance by fine-tuned ML pipelines that contain a resampling and a classification algorithm, demonstrating the experimental design's usefulness as well as supporting our answer to **RQ1**.

Chapter 6 presents our new method to compute performance for classification problems where the distribution between classes is imbalanced and has unequal class importance. In ML, the assessment method is critical in evaluating an ML model's performance and choosing the suitable ML model that works well on the given problem. Many performance metrics, such as F1, geometric mean, recall, and precision, can be used in class imbalanced learning. However, these methods do not consider the unequal class problem.

Built on top of standard performance metrics, we propose a new performance metric incorporating unequal class importance into the standard performance metrics. More precisely, we propose to compute the classification performance based on the new penalized confusion matrix based on the actual confusion matrix and a user-defined penalty matrix. The domain experts define the penalty matrix, which contains the penalty values between actual and predicted classes. The penalized confusion matrix is then produced by multiplying every element of the actual confusion matrix with the corresponding element in the penalty matrix. The final classification performance is later calculated via the new penalized confusion matrix instead of the actual one, as usual. Our approach solves **RQ3** in handling the dual problem of class imbalanced and unequal importance between classes. Moreover, we also investigate the correlation of assessment values between the

new method and the standard one. Our finding indicates that the performance computed by the new approach and the standard metrics are strongly correlated. That is to say, our approach incorporates unequal class importance into the standard performance metrics and does not change their purposes (e.g., metrics for imbalanced problems).

Chapter 7 researches improving BO performance for AutoML optimization problems via maximizing coverage of search space already during the initial sampling of BO to characterize the response surface more accurately. The initial sampling step is the first step of BO to utilize the first response surface. It is typically restricted to a small budget since the effectiveness of BO becomes evident mainly in the later stages of optimization when it learns to produce better configuration. Considering how to improve coverage over AutoML search space within a limited budget, we propose the novel *combination-based sampling approach* for the initial sampling stage of BO. Our proposed initial sampling approach is as follows. We first attempt to group algorithms with similar technical behaviours as in the literature, where one can represent the rest of the group. Then, we reallocate the sampling budget to explore the potential of similarities between algorithms within the group: sampling fewer of the same algorithms frees up the budget to be distributed to other (different) algorithms, thus optimizing the coverage of algorithm-hyperparameter search space.

To investigate the potential of our proposed approach in AutoML optimization scenarios, we compare the performance of BO with and without using it on the two AutoML benchmark experiments (see Chapter 4) over 117 classification problems. The key findings from this indicate the following. In the first set of benchmark experiments, we evaluated them on two scenarios of initial sample sizes – 20 and 50 iterations. With our improvement, the performance of BO significantly improved in several cases and did not significantly worst in any tested cases. In the second experiment, we also compared the two experimented BO approaches against the other six well-known AutoML products (i.e., Auto-sklearn (BO and Random search), H2O, TPOT, ATM and Hyperopt-sklearn). The experimental results indicate that the BO using our initial sampling approach produces the best results and significantly outperforms others in more cases than all compared approaches. In contrast, the experimented BO without our improvement does not significantly win in any tested cases.

In conclusion, the experimental results answer **RQ4**: Our approach, which maximizes the coverage of algorithm-hyperparameter search space during the initial

sampling stage of BO, clearly improved BO performance in solving the AutoML optimization problems.

Chapter 8 introduces a novel contesting procedure algorithm, **D**ivide **A**nd **C**onquer **Opt**imization (DACOpt), to efficiently solve AutoML optimization. Motivated by the fact that BO performs better for low-dimensional problems [31], while AutoML is typically high-dimensional mixed variables. This causes BO to be less robust for solving AutoML. To limit this issue, we first partition the search space into a reasonable number of sub-spaces based on algorithm and budget constraints. Then, multiple BO performs on every sub-space independently (i.e., sub-process) to optimize BO performance. Due to their independence, this also allows them to be explored in parallel. Additionally, we adopt the ideas of the two well-known multi-fidelity approaches (i.e., bandit learning and racing procedure) into our procedure to eliminate ineffective sub-processes to free up the budget to be distributed to the better one, thus optimizing the budget usage.

Generally speaking, the contesting procedure is complementary to the existing BO approaches to handle their limit when optimizing the AutoML problems. The proposed approach is constructed of three main components:

- The *splitter function* to partition the search space into multiple sub-spaces. Then, an existing BO approach is employed to optimize those sub-spaces independently, leading to a corresponding number of optimization sub-processes.

- The *elimination function* decides to stop poorly performing sub-spaces (i.e., terminate the corresponding sub-processes). This function also addresses **RQ5**, which concerns when we should stop tuning in a particular area (sub-space) of the search space. To summarize, we can stop tuning in an area when it demonstrates significantly worse results compared to the most promising area.

- The *controller function* allocates budgets adaptively for tuning each sub-space based on the performance of the corresponding optimization processes. This function provides an answer to **RQ6** on how to allocate computational resources over the search space. In simple terms, we conduct multiple competitions over the available resources, which are calculated based on the input computation resources and the number of areas. After each race, several areas are eliminated, and the remaining areas share the saved resources of the race. This ensures that the most effective area stays longer and has the most tuning resources.

Lastly, we compare the performance of BO with and without our contesting procedures on the two AutoML benchmark experiments Chapter 7. We use Hyperopt [153] and BO4ML [15] as the underlying BO. Besides, the proposed procedure has variants of the elimination function (i.e., it adopts bandit learning and racing procedure). Thus, we have four variants in total. The key findings from this indicate the following.

- First, in both experimental scenarios, both BOs with our contesting procedure significantly perform better than one without using it. Overall, the contesting procedure that used BO4ML achieved more best results than any other competitors. This finding again demonstrated the effectiveness of BO4ML (i.e., also providing an answer to **RQ4**) for solving AutoML problems.
- Comparing the two adopted elimination functions, the one that adopted bandit learning performs better than the competitor. Thus, we recommend that researchers use the bandit learning variant for AutoML problems.
- Lastly, we also compared the proposed approaches against the six state-of-the-art AutoML products (i.e., Auto-sklearn (BO and Random search), H2O, TPOT, ATM and Hyperopt-sklearn), in the second scenario. The proposed contesting procedure produces the best results, performs well (i.e., either achieved the best performance or not significantly worse than the best-found method) and significantly outperforms others in more cases than all compared approaches (i.e., our best contest procedure produces 28 highest, 53 well-performing and 26 significantly outperforms values over 73 tested cases).

In conclusion, the experimental results demonstrated the effectiveness of our new contesting procedures in solving AutoML optimization problems. They notably enhanced BO's performance, and the AutoML, using our proposed contesting procedure as an optimizer, won over the current state-of-the-art AutoML tools, such as H2O, Auto-sklearn, ATM, Hyperopt-sklearn, and TPOT, in a wide range of benchmark tests.

9.2 Future work

This thesis has focused on conducting research in the field of Automated Machine Learning (AutoML) and has presented significant achievements and insights. However, numerous future works within AutoML still require further investigation

and development. In this section, we outline some potential directions for future research and provide insights into how they can contribute to advancing the state-of-the-art in AutoML. In the following discussion, we will explore potential future research directions to extend the work presented in this thesis.

9.2.1 Combination-based sampling

The combination-based sampling approach, introduced in Chapter 7, aims to maximize the coverage of algorithm-hyperparameter samples in the search space. This approach improves the accuracy of characterizing the response surface during the initial sampling stage of the historical-based optimization approach. Here are several potential future research directions for extending this work:

- The combination-based sampling is initially incorporated for Tree Parzen Estimators [153]). It would be interesting to exploit the potential of applying the proposed sampling approach to other BO variants (e.g., SMAC [25], MIPEGO [38]) and historical-based approaches, such as evolutionary strategies [108] and genetic algorithms [103]. By extending the evaluation of our sampling approach to different optimization techniques, we can assess its generalizability and potential for improving the performance of various optimization algorithms. This exploration may provide a comprehensive understanding of how the initial sampling effect to the later stage of historical-based optimization approaches.

- It would be interesting to explore how the combination-based sampling approach, introduced in our work, performs during the sequential sampling stage of BO. By maximizing the coverage of algorithm-hyperparameter samples in the search space, the candidate configurations proposed at each iteration may exhibit better exploration and exploitation properties. This could potentially lead to more efficient and effective sampling. This research direction can potentially enhance the optimization process, improve the quality of candidate configurations, and provide valuable guidelines for selecting suitable sampling strategies for historical-based optimization approaches to address AutoML problems.

9.2.2 Contesting procedures

The contesting procedures algorithm introduced in Chapter 8, serves as an efficient approach for addressing AutoML optimization problems. This algorithm offers

promising results, but several potential avenues for future research can further enhance and expand on this work. The following are some of the possible research directions discussed:

- As part of the proposed contest procedure, the algorithmic hierarchy attribute allows organizing *the choice of algorithms* in the search space in a hierarchical manner, with one algorithm in a branch potentially representing multiple other algorithms. Due to the large set of possible combinations of algorithms over operators (i.e., functional algorithms in ML pipeline structures), it is not possible to try every combination in practice. Hence, the algorithmic hierarchy is a realistic way to identify ineffective combinations. However, we acknowledge that the algorithmic hierarchy is currently constructed based on the experiences of practitioners in the field. As such, the resulting structure of the search space may not be optimized. It would be highly advantageous to explore methods to optimize the hierarchical structure using historical data from experiments. Advanced techniques such as clustering methods can be applied to automatically identify patterns and relationships among algorithms, enabling the creation of an optimized algorithmic hierarchy. This data-driven approach would enhance the efficiency and effectiveness of AutoML by incorporating empirical insights in a systematic and automated manner, reducing the reliance on manual construction.

- Moreover, it is worth considering the integration of meta-learning approaches to identify promising search areas at an early stage. By leveraging meta-learning techniques, we can leverage prior knowledge or learned patterns to guide the optimizer. This can help accelerate the optimization process by focusing on areas that have shown promising performance in previous similar tasks. We believe that the incorporation of meta-learning can enhance the efficiency and effectiveness of the optimizer, enabling them to make informed decisions and prioritize exploration in the search areas likely to yield favorable results.

9.2.3 Benchmarking methods and application domains

Indeed, an important concern in evaluating AutoML optimization algorithms is the inconvenience and high cost associated with using a wide range of **real datasets**. This process can be time-consuming, resource-intensive, and financially burdensome. While benchmarking with **synthetic test functions** is a common strategy in

optimization studies due to their closed-form representation and efficient evaluation, the existing synthetic test functions are not suitable for AutoML benchmarking [22]. This is primarily because they do not simulate categorical hyperparameters, pure categorical hyperparameters, or algorithm choice hyperparameters, nor do they account for structured search spaces. An interesting direction for future research is the development of *synthetic test functions specifically designed for AutoML benchmarking*. These test functions should accurately represent the complexities and characteristics of real-world AutoML problems, including the incorporation of categorical hyperparameters and structured search spaces. By defining such synthetic test functions, researchers and practitioners can evaluate and compare AutoML optimization algorithms in a more controlled, cost-effective, and efficient manner. This would enable the systematic analysis of algorithm performance and facilitate advancements in the field of AutoML.

Empirical Performance Models (EPMs), as an alternative to synthetic test functions, introduced by [249]–[252], provide a surrogate representation of the response surface of a specific performance metric. These models aim to capture the empirical performance characteristics of a real dataset, offering a means to theoretically evaluate algorithms in AutoML scenarios. It is important to note that existing empirical performance models (EPMs) are not tailored for AutoML scenarios, as mentioned in previous studies [22], [253]. This raises the need for further investigation and assessment of its suitability within the AutoML context. Exploring the applicability of EPMs in AutoML can lead to valuable advancements in the evaluation and benchmarking of AutoML algorithms.

These future works to AutoML benchmarking approaches would drive progress and advancements in the field, promoting the development of robust and efficient AutoML solutions for a wide range of practical applications.

Lastly, in this thesis, we have conducted comprehensive investigations into AutoML, focusing primarily on supervised machine learning problems. However, it would be beneficial to expand the discussion to include other domains of machine learning as well. Exploring studies and research in **unsupervised machine learning**, **reinforcement learning**, and **deep learning** can provide valuable insights and a broader understanding of the topic. This would allow for a more holistic analysis of AutoML's applications and effectiveness across various ML domains.