A Systematic Literature Review on AutoML

Hanying Zhang

Multi Agent System Course
Two Year Master's Degree of Artificial Intelligence
University of Bologna

July 23, 2022

Abstract

Machine Learning has become very popular these years and has deeply changed our daily lives. However, most of the Machine Leaning tasks need heavy work from human experts to achieve good results. Therefore, an end-to-end Machine Learning technique without the assistance of humans –AutoML– will make it much easier to apply Machien Learning in different areas. In this report, a survey on AutoML will be carried out, focusing on the theoretical background, the main approaches and techniques used in AutoML.

1 Introduction

Automatically working machines and intelligent agents have long been the dream of human beings, from eastern to western worlds, from remote history to modern days. Due to the quick development of Machine Learning in recent years, we have never been so close to that dream. Machine Learning has already deeply changed our daily lives. Deep Learning, among the various techniques of Machine Learning, is the one that attracted most attention and achieved most impressive results.

However, most of the Machine Learning tasks require heavy assistance from human beings, especially experts of specific domains to achieve good results. This heavily obstacles the spreading of machine learning applications. Therefore, an end-to-end machine learning technique without human intervene, i.e., AutoML, will make it much easier to apply Machine Learning in different areas.

Since being first introduced in 1990s, AutoML has experienced great development, especially in the past decade. Many novel techniques have been introduced and challenges like Chalearn[10] have been proposed. Open source community and big tech companies like Google and Amazon have also released their AutoML tools. Thus, I would like to make a survey on updated AutoML, focusing on the background theory, the main approaches and techniques used in AutoML.

How this article is organized is as follows: in section 2 the research objective and questions are proposed to guide the whole process of this SLR. Search Strategy, Study Selection and Quality Assessment Criteria are introduced from section 3 to section 5. In section 6 the whole process of AutoML, i.e., the pipeline, is introduced. The main components in the pipeline are introduced from section 7 to section 9. In section 10, many AutoML frameworks are introduced, including the benchmark and evaluations on these tools. A discussion on Human-Machine Relationship is discussed in Section 11. Section 12 concludes this survey and answers the two research questions. Future research gaps and challenges are also discussed here.

2 Research Objective and Questions

The goal of this Systemacit Literature Review is providing a up-to-date survey of AutoML. There are two related search questions. The first one is how automation is achieved in AutoML. The second one is what's the similarities and differences of approaches and techniques used in AutoML.

3 Search Strategy

3.1 Sources and Keywords

Several search sources are included in this SLR, namely Google Scholar, the university library of Unibo (AlmaStart), arXiv and OAIster. Google Scholar, arXiv and OAIser are among the most famous engines for scholar search. Google Scholoar provides very useful functionality such as sorting results according to citation index and publicatin date. ArXiv is an open-access archive is extrodinaryly popular for Machine Learning researchers. AlmaStart is especially useful for accessing papers which are only available through the university account.

The keywords used during the search process are: AutoML; AutoML + background / approach / technique / framework / tool / comparasion. AutoML is used as the main keyword. It is used to make sure that the most famous papers on AutoML are not missed. All the other keywords are used to search papers focused on specific areas of AutoML.

3.2 Include/Exclude Criteria

The critera for inclusion is as following:

1. the paper has high citations on Google Scholar and OAIster(>=50)
2. the paper is focused on the theoratical background of AutoML
3. the paper is focused on the approaches of AutoML
4. the paper is focused on the tools of AutoML

The criteria for exclusion is as following:

1. the paper is unrelated to AutoML
2. the paper is a duplicate of one of the previous papers
3. the paper's full text is not available
4. the topic of the paper is an application of AutoML on a specific area

4 Study Selection

During the Search process, applying the include/exclude criteria, a total of 44 papers are selected as the primary sources. All these papers are published between 2016 and 2021. Most of the papers are from Europe, China and the US.

5 Quality Assessment Criteria

After reading the full text, all the sources will be split into 3 groups: excluded, good and excellent. A paper will be excluded if most of its content is duplicated or unrelated to theoratical backgound/approaches/techniques of AutoML. Group excellent contains primary sources which provied insightful information. If a source provides some useful information, it will be included in group good.

Four papers are excluded according to this quality assessment criteria. Two papers named Evaluating Generic AutoML Tools for Computational Pathlogy and AM-LFS AutoML for Loss Function Search are excluded because they are applications on a specific area. One paper named AutoML to Date and Beyond is excluded because it is a survey itself. The reason one paper named Low-code AutoML-augmented Data Pipeline: A Review and Experiment is excluded is that it's low quality and unrelated.

Two papers are grouped as excellent whose names are AutoML Zero: Evolving Machine Learning Algorithms from Scratch and Transfer Learning with Neural AutoML specifically. The reason is that both papers provide very insghtful information. All other papers are gouped as of good quality.

6 AutoML Pipeline

AutoML systems consists of many components, each of which takes charge of one specific task, one by one. This is called the AutoML Pipeline.

Briefly speaking, the pipeline consists of the following essential components: Data Preprocessing, Feature Engineering, Model Selection and Optimization, as shown in the following diagram. In this survey, the last two components are sometimes combined together to discuss because there are many papers which take them as a Combinatorial Optimization problem[12][18][21][23].

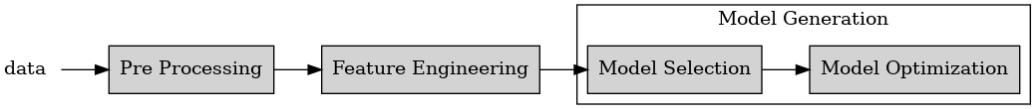

Figure 1. AutoML Pipeline

7 Data Preprocessing

The first step in AutoML pipeline, after receiving the input data, is Data Preprocessing. This step could be done in two main processes, namely Data Cleaning and Data Augmentation.

The purpose of Data Cleaning is to filter noisy data, in order to avoiding the unreliable of outcomes and algorithms, even though they may look correct. Typically, Data cleaning consists of fixing or removing incorrect, corrupted, incorrectly formatted, duplicate, or incomplete data within a dataset. Data Augmentation is essential especially when the dataset contains no enough number of samples. Data Augmentation could also be viewed as a technique to void the problem of Overfitting.

As the first step of the whole AutoML pipeline, the impact of Data Preprocessing is hard to be measured, especially in a quantitive way. The problem would be worse when data transformations are combined into preprocessing pipelines. In paper[11], the impact of transformations and transformation pipelines is studied. The paper shows that the overall impact of optimizing preprocessing is not eligible and it may boost the performance of the overall result. A generic method which allows to find effective pipeline prototypes is also developed. Specifically, the impact of precedence order between pairs of various transformations is examined. As a result, a method is defined that allows to generate effective pre-processing pipelines which consists of 1) compatible pairs of transformations with respect to the framework used, 2) meaningful pairs of transformations in terms of general knowledge (best practice), and 3) promising pairs of transformations that once applied are expected to provide higher overall impact (domain knowledge).

8 Feature Engineering

Feature Engineering is the process of transforming raw data into features that can better represent an underlying problem for computational predictive models, resulting in improved model accuracy.

Traditionally, Feature Engineering includes two subproblems, namely Generating new features form data and Enhancing current features' discriminative ability. The first problem heavily relies on human experts and there's little progress. For the second problem, there are several common methods to deal with it. Specifically, when the feature dimensionality is very high or the features have high redundancy, Dimension Reduction could be applied. Feature Generation method is used to construct new features by exploring interactions among original features.

The requirement of good features becomes less urgent when Deep Neural Networks is included in the search space of pipelines as deep neural networks could learn good feature representations automatically.

Nowadays, Meta Features are widely used to mitigate the problem of heavy computational cost in AutoML[24][31][18][30][23]. However, these features are not features IN the datasets, but features OF the datasets, which is clearly different from the traditional feature engineering. Technique using meta features can dramatically reduce the computational cost and time requirement.

9 Model Generation

Model Generation consists of several consecutive processes. First of all, a Search Space containing both models and hyper-parameters must be established. Model Selection Strategy is used to select models from search space. When the models are chosen, we still need to optimize the related hyper-parameters. When neural networks are included in the model, we still need to optimize the model architecture.

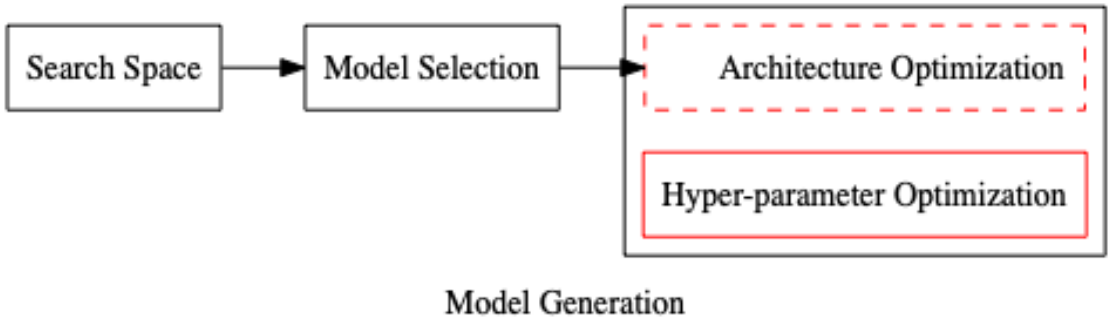

Figure 2. Components in Model Generation

Instead of taking care of the previous modules one by one, like many researches do, there are also several techniques which take them as a Combinatorial Optimization problem[12][18][21][23]. For instance, in research[12], the automation of search space and model selection strategies are formulated as a combinatorial optimization problem. In research[21] and [18], the automation of the pipeline is configured by jointly selecting models and hyper-parameters. This is also known as the famous CASH (Combined Algorithm Selection and Hyper-parameter Selection). However, combinatorial optimization requires more computational resources.

The importance of the modules in model generation is also studied in these researches. Specifically, in research[12], it is shown that identifying the suitable search space is more important than choosing a sophisticated model selection strategy. In search [28], model architecture is proven to be more important than hyper-parameter selection.

9.1 Search Space of Models

Search space includes various models and their hyper-parameters. Different AutoML tools use search spaces including different kinds of base models. One significant difference between search spaces is whether they include neural networks as a base model. If neural network is included, the pipeline has to consider the problem of neural network architecture optimization. Neural Architecture Search is the technique to handle this problem.

In research [19], users can express their preferences over the components in search space to appear in the generated pipeline. This is implemented by letting users to provide a weak pipeline specification, from which the AutoML system use to produce a strong specification, i.e., a full candidate pipeline. For instance, Logistic Regression could be used as a weak specification. The AutoML system could strengthen this specification by adding other models, regulations, and a model selection strategy, to form a final pipeline.

AutoML systems suffer from scalability issues when facing large, high-dimensional search spaces. VOLCANOML[7] is a scalable and extensible framework that facilitates systematic exploration of large AutoML search spaces, by implementing basic building blocks that decompose a large search space into smaller ones. These building blocks are similar to the relational operators in a database system. Users are allowed to design decomposition strategies in a structured fashion similar to relational execution plans.

Almost all AutoML tools construct their search spaces using common base models, such as logistic regression and neural networks, i.e., restrictive search spaces. However, one research[26] shows that AutoML can go further: it's possible to use only basic mathematical operations as building blocks. This research significantly reduces human bias through a generic search space. Evolutionary search is used and proved to be able to find two-layer neural networks, despite the vastness of the search space.

9.2 Model(Algorithm) Selection and Optimization

After fixing the search space, the AutoML tools need to choose specific models which are most suitable for the task from search space, also the hyper-parameters of these models. Generally speaking, there are two type of methods doing this. The first type, takes care of search spaces which include only traditional models, such as KNN, Logistic Regression and SVM. The second type of method, whose search space includes deep neural networks, is called Neural Architecture Search. NAS focused on selecting DNN architectures and hyper-parameters.

9.2.1 Random Search and Grid Search

Theoratically, Grid Search and Random Search could be used to solve the model selection and optimization problem. For example, by randomly searching some models and hyper-parameters from the search space, one final pipeline could be constructed. However, the generated pipeline could probably perform badly. Generally speaking, the whole process is very inefficient. Therefore, we need to find more efficient strategies.

9.2.2 Bayesian Optimization

Bayesian Optimization is a powerful strategy using Bayes Theorem to direct the search in order to find the best pipeline configuration. A Surrogate Function representing the posterior probability is used to estimate the optimal pipeline. An Acquisition Function is used to guide the sampling process. Specifically, We want to use our belief about the optimal pipeline to sample the area of the search space that is most likely to pay off, therefore the acquisition will optimize the conditional probability of locations in the search to generate the next sample. Acquisition function needs to trade-off between exploration and exploitation.

9.2.3 Ensemble

An ensemble of models is a model that utilized predictions from multiple models to produce an even better model. It is widely used in Machine Learning field. Stacking is an ensemble machine learning algorithm that learns how to best combine the models in an ensemble to achieve the best performance. Therefore, besides the base models, a higher level meta model is also included to learn how to combine them.

The training dataset of meta-model consists of predictions of base models on out-of-sample (via k-fold cross-validation, for instance) data and the label of out-sample data. Specifically, data not used to train base models is fed to the base models to generate predictions, these predictions along with labels, provide the input and output pairs of the training dataset.

In addition, stacking the ensembled models into a multi-layer structure is also shown to be much efficient[3]. When combined with fast random search, stacked ensembles could achieve competitive results compared to other frameworks using more complex model tuning techniques[17].

The diversity of current AutoML tools could also be exploited by leveraging the differences in their search spaces and heuristics. Ensemble Squared[27] is the AutoML tool that ensembles the results of SOTA open source AutoML tools. The performance of this framework is competitive to SOTA AutoML tools.

9.2.4 Evolutionary Algorithm

Evolutionary algorithms are a heuristic-based approach to solving problems that cannot be easily solved. An Evolutionary Algorithm uses mechanisms inspired by biological evolution, such as reproduction, mutation, recombination and selection.

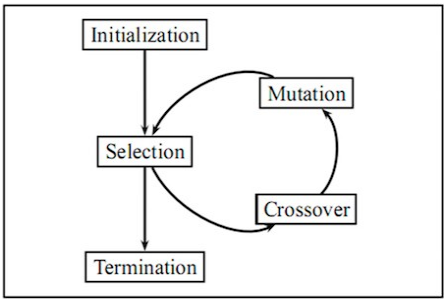

Figure 3. Evolutionary Algorithm

Evolutionary algorithms may perform better than grid search or random search. TPOT is a AutoML tool which uses evolutionary algorithms for pipeline construction. But evolutionary algorithms rely heavily on the initial configurations, meaning that the final output may be unstable and the process may be time consuming. Therefore, evolutionary algorithms is mainly used in Neural Architecture Search, which will be introduced below.

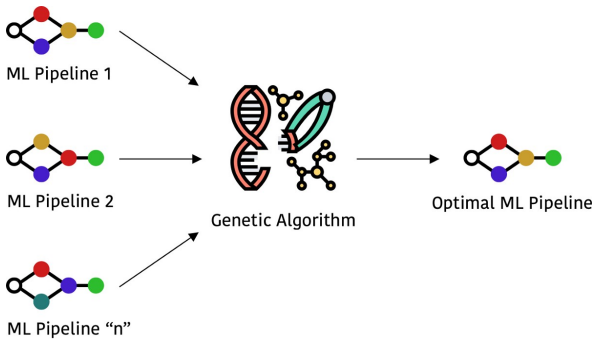

Figure 4. Evolutionary Algorithm in TPOT

9.2.5 Meta Learning

Meta learning is also widely used in AutoML with the purpose to learn more deep features to guide the construction of AutoML pipelines. In other words, meta learning builds pipelines based on the observations of various configurations on previous ML tasks.

First of all, meta features of datasets and algorithms are computed during a 'offline' process. These features can be used to measure the similarity between the new dataset and the existed ones[30]. The size of search space could be reduced by selecting only the models which performed better on the similar datasets.

When dealing with a new machine learning problem, human often first choose to read the description of the dataset and the documentation of related algorithms. The same idea could also be used in AutoML to improve the performance, i.e., using language embeddings from modern NLP to improve AutoML systems by augmenting their recommendations with vector embeddings of datasets and algorithms[24].

9.2.6 NAS

Neural architecture search is the task of automatically finding one or more architectures for a neural network that will yield models with good results for a given dataset. It is the technique used when the search space consists of neural networks.

Search Space of NAS

The search space of NAS evolves according to the development of Deep Learning. For instance, at early stages, there are only Fully Connected layers, and the hyper-parameters such as number of layers and number of activations in one layer. As the development of Deep Learning field, more sophisticate structures emerged, such as CNN, Resnet and Inception, these structures are also included in the search space.

Most of the search space consists of building blocks which are human designed, making it a restrictive search space. However, it is also shown that it is possible to automatically discover complete machine learning algorithms just using basic mathematical operations as building blocks[26]. This approach could significantly reduce human bias on the design of search spaces. Using evolutionary search and back-propagation, two-layer neural networks could still be discovered.

Architecture Optimization

After fixing the search space of NAS, we need to search for the most suitable architecture for the given problem. This problem is defined as Architecture Optimization. In order to cope with the problems of high computational consumption and instability in NAS, two techniques named Reinforcement Learning and Evolutionary Algorithms are introduced[29]. Reinforcement Learning will be introduced below as the theory of evolutionary algorithms is already discussed before.

Reinforcement Learning

Reinforcement Learning is an area of machine learning focusing on how intelligent agents ought to take actions in an environment in order to maximize cumulative reward. The purpose of reinforcement learning is for the agent to learn an optimal policy that maximizes the "reward function" or other user-provided reinforcement signal that accumulates from the immediate rewards.

An overview of an RL-based NAS algorithm is shown in the following figure. The RNN controller executes an action At to generate a new architecture and receives an observation St and reward Rt from the environment. These are used to update the controller's sampling strategy.

Figure 5. Reinforcement Learning in NAS

AutoML with NAS could outperform many SOTA AutoML systems, even some hand designed solutions. The architecture, components and hyper-parameters could all be optimized to fit the requirements of the task, with a superior performance[28].

9.3 Limitation Handling

The AutoML tools must be finished within a specific time duration during which many pipelines must be constructed and evaluated. Also, the computational resources are limited. Therefore, the tools try to use several techniques to avoid the time and computational resource limitation problem.

Typically, in AutoML a search procedure is used to repeatedly generate and validate candidate pipelines, subject to a limited execution time budget. This could be a problem when facing a large dataset. Therefore, one research that uniformly downsample the datasets shows that downsampling can actually improve the performance[9]. Also, downsampling more aggressively allows the search procedure to generate and evaluate more pipelines, those pipelines are also tend to be more complex.

9.3.1 Successive Halving

Successive Halving Algorithm (SHA) is a bandit strategy which could be used to take care of the budget allocation problem. SHA allows to specify a minimum and maximum budget for each configuration. It starts with the minimum budget and iteratively increases the budget for the pipeline that perform best with the current budget. By successively reducing the number of candidates and at the same time increasing the allocated resources (number fo samples) per run, we could quickly finish the selection while receiving a relatively good performance. Research [5][4] both implemented this strategy.

9.3.2 Proxy Model

Proxy models could be used to predict candidate pipeline configuration performance before building the full final model. In such a way the computational cost could be reduced a lot. In research [31], each pipeline stage makes decisions based on meta-learned proxy models. This approach builds and tunes only the best candidate pipeline, achieves better performance at only a fraction of previous methods.

9.3.3 Decomposition

By decomposing one problem into several sub-problems, the computational complexity could be reduced exponentially. This strategy is applied in research[7], which decompose a large search space into smaller ones. Paper[23] uses low rank tensor decomposition as a surrogate model for efficient pipeline search.

9.3.4 Transfer Learning

Transfer Learning is a research problem in machine learning that focus on storing knowledge gained while solving one problem and applying it to a different but related problem. But using the stored knowledge, the new problem could learn much more quickly.

In research[13], the computational cost of Neural AutoML is reduced by applying transfer learning. Specifically, knowledge from prior tasks is used to speed up network design. RL-based architecture search methods are extended to support parallel training on multiple tasks and then transfer the search strategy to new tasks. A so-called Transferable AutoML approach which leverages previously trained models to speed up the search process for new tasks and datasets is also presented in paper[15].

10 AutoML Tools

AutoML has always been one of the hottest field in Machine Learning since its born. Many organizations and companies have developed AutoML frameworks or tools of their own. These tools use different techniques and strategies, forming a prosperous AutoML world. For instance, besides the traditional 'offline' AutoML strategies, one 'online' framework named ChaCha[25] is also provided to cope with the online learning manner. However, this section will focus on the traditional AutoML tools.

10.1 Main Tools

Several reprensentative tools are going to be introduced here. They are selected according to several criteria, including techniques used, reputation in open source community, support of big companies, etc.

10.1.1 H2O

H2O AutoML supports the training of Stacked Ensemble models. In H2O these stacked ensemble models are constructed by a Meta Learner called Super Learner. H2O's core code is written in Java, a Python API is also provided.

10.1.2 TPOT

Tree-based Pipeline Optimization Tool, relies on Genetic Evolutionary Algorithms to construct and optimize pipeline. It could automatically optimize a machine learning pipeline built around Scikit-Learn(uses Scikit-Learn as the ML backend). The model search space includes several models such as Naiva Bayes, Random Forest and Logistic Regression.

10.1.3 Auto-Sklearn

Auto-Sklearn is one of the most famous AutoML frameworks. Auto-Sklearn uses Bayesian optimization with warm start(meta-learning) to find the optimal model pipeline and build an ensemble from the individual model pipelines at the end.

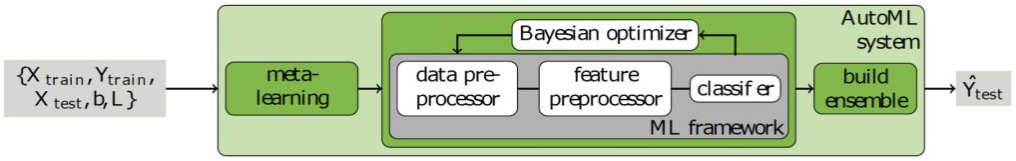

Figure 6. Auto-Sklearn structure

10.1.4 Google AutoML

The main technique used in Google AutoML is the combination of Deep Reinforcement Learning and Neural Architecture Search. It makes no surprise that Google choses the fields which it is really good at.

10.1.5 Amazon SageMaker Autopilot

The name of Amazon's AutoML tool is SageMaker Autopilot[20], which operates only on tabular datasets. It also includes many models in search space to balance between different needs. The main novel point is that it exposes not only the final models but also the pipelines, catering the needs of users with different levels of expertise. Autopilot is built as an AWS managed service.

10.1.6 Oracle AutoML

The main feature of Oracle AutoML is Iteration-Free, which means that it could run in a shorter runtime. This is done by making decisions based on Metalearned Proxy Models at each pipeline stage. These models can predict candidate pipeline configuration performances before building the full final model. Oracle AutoML is also planned to be made could-native, to further take advantage of the Oracle Cloud.

10.1.7 Huawei VEGA

VEGA[16] is similar to Google AutoML as the main technique used is also NAS. The main novel point of VEGA is the design of a novel fine-grained search space and its description language, to support a variety of search algorithms and task. A unified front-end interface is also provided for training frameworks and supports different deep learning frameworks.

10.2 Tool Adoption and Critics

In spite of the presence of many AutoML tools and the various techniques they used, the adoption rate of these tools is still not high[32]. This survey shows that around 20% to 30% of the respondents have not adopted AutoML at all and many more only partially. It also indicates that the adoption may be inhibited by usability issues with AutoML frameworks and the increased computational resources needed for adoption. Similar issues, such as confusing log output, low-support for training time estimates, and a lack of support for visualizing internal state are also reported[36]. Another interview study[8] also studied the benefits and deficiencies of AutoML with participants ranging from novice hobbyists to industry researchers. According to the results, ease of use, effectiveness and efficiency are the highest-rated qualities of AutoML. The deficiencies include: lacks comprehensive end-to-end support, causes system failures due to compute intensive workload, lacks customizability and lacks transparency and interpretability. Currently, most of the AutoML tools focus on supervised learning, and the number of components in pipelines generated by AutoML frameworks is less than 2.5[6].

One critic on current AutoML tools is that most of them focus solely on the predictive performance of models built by these tools[40][34]. Other criteria, such as fairness, transparency and interpretability should also be taken seriously because predictions need to be explained to users or society.

One study[35] investigates the robustness of pipelines generated by AutoML systems, in particular the influence of dirty data on accuracy and how using dirty training data may help create more robust solutions. The results show that when using dirty training data, the accuracy may drop a bit but the robustness increases a lot.

The potential security risks incurred by NAS is studies in one search[33], which shows that compared with the manually designed models, NAS generated ones tend to suffer greater vulnerability to various malicious attacks (adversarial evasion for instance). The possible explanation is that given the prohibitive search space and training cost, NAS methods favor models that converge fast at early training stages, and this preference results in architectural properties associated with attack vulnerability.

10.3 Evaluation and Benchmarks of AutoML Tools

Benchmarks have been established to measure the performance of the AutoML tools. However, comparing different AutoML systems is not easy and done incorrectly. For instance, the selection of datasets is often limited in scope, resulting the possibility of overfitting on specific set of datasets. Authors may even only select datasets on which their tools perform well. Furthermore, insufficient compute resources may also introduce bias.

Thus, a standard benchmark is needed to evaluate different AutoML tools. In research[22], an open source and extensible AutoML benchmark is presented. This benchmark uses fewer but larger datasets which are all not very easy to solve. As of performance metrics, AUROC (area under the receiver operator curve) is used for binary classification problems and log loss is used for multi-class classification problems. The scores of different tasks are normalized such that the constant predictor is 0 and the tuned random forest is 1.

Another research[37] outlined three critical AutoML evaluation best practices, including: 1) the separation of datasets used for system development from those used for evaluation, 2) using the same set of ML building blocks systems use, and 3) standardizing a common ML program description language. A unified, open source machine learning framework upon which AutoML systems can be built is also presented, to enable a thorough evaluations of AutoML systems.

11 Human Role in AutoML

There are two different directions where different tools are forwarding. The first group is still forwarding to the initial goal of AutoML: no human intervene at all. For these tools, human is expected to be totally excluded. The second group of tools, in contrast, realized that total automation is not realistic nowadays[8]. Therefore, they turned to the direction where human and tools interact and cooperate. However, generally speaking, the first group still dominates the research[6].

In research[1], an environment named MILE (Mixed-Initiative machine Learning Environment) where humans and machines together drive the search for desired ML solutions is presented. Three settings representing different levels of automation is also identified: user-driven, cruise-control and autopilot. Intermediate results, additionally, should be made available to practitioners.

Gradual AutoML is the research where data scientists can control certain choices while AutoML explores the rest. In this way, data scientists can apply their intuition. One AutoML library named Lale[38] is introduced to simplify gradual AutoML by introducing the concept of combinators from functional programming. Lale gives users fine-grained control over AutoML without requiring them to be AutoML experts.

12 Conclusion and Future Reseach

In this SLR on AutoML, the techniques and apporaches used in AutoML are studied. Several reprensentative tools from open source community and componies are also introduced, including evalutations and critics on them.

The automation in AutoML is achieved by applying automation on consecutive components in the AutoML pipeline. The data preprocessing could be automated by using diffent pairs of data transformations. Dimension Reduction could be automated applied when the dimension of features are too high and Feature Generation exploring interactions between original features could be used to generate new features. Deep neural networks could mitigate the feature engineering problem as they can learn feature representations automatically. The automation of model generation, in essential, is the optimization of the searching process from search space which includes various base models and hyper-parameters. It should be noticed that most existent research on the AutoML belongs to the class of Narrow AutoML[2] and the possible benefits obtained are at a cost of increase in computing burdens. Therefore many techniques such as meature learning and transfer learning are also proposed to mitigate this new problem.

Most research upon AutoML are focused on the model generation part, which could be divided into two groups according to whether they adopt NAS. Among the approaches withoud NAS, grid search and random search are the most basic methods without using any search heuristics, which means that it's very difficult for them to find a good result. Stacked ensemble and Bayesian optimization both use search heuristics. Stacked ensemble implements a meta model to learn how to best combine the models in an emsemble. Bayesian optimization uses an acuisition function to guide the sampling from the area which is most likely to paf off in the search space. Evolutionary Algorithm uses mechanisms inspired by biological evolution to gradually generate a better pipeline from the initial configuration. It could be used in combination with random search and grid search. The Automation of NAS is achieved by implementing reinforcement learning or evolutionary algorithms. Reinforcement learning learns by interacting with the enviroment. The network architecture could be optimized by maximizing cumulative rewards of the agent. Evolutionary algorithms could also be used to optimize the network architecture, even when the search space is generalized to contain only math operations.

However, as full automation seems still a dream too far away, the researches are moving towards two different directions. That is, besides the initial direction where human intervene is totally excluded, many researches are focusing on how to facilitate the cooperation between human and machines. Different levels of automation have also been proposed. Generalized AutoML has a strong tie in spirit with Artificial General Intelligence, which means that there is still a long way ahead of us.

As mentioned before, much fewer researches on AutoML are about data pre-processing and feature enginnering. But these two parts are also of great importance because this is where domain experts utilize their knowledge. Besides, although many techniques and methods are proposed to solve the efficiency problem of AutoML systems, there is still much room to progress. Funthermore, the usablity and interpretability of many AutoML tools are also need to be improved.

Evolutionary Algorithms show great potential in AutoML. It could even build effective neural networks from a search space where only math operations instead of human-designed building blocks are available. How to explore its full potential to construct more complex and effective network architectures could be a promising future research.

How to implement Multi Agent System theories and techniques in the field of AutoML is also a very insteresting research topic. For instance, the similarity between the search space in AutoML and the Tuple Centre in MAS sheds light on this possibility. Specifically, whether agents in a tuple centre could be used to represent base models in an AutoML search space, where each agent takes care of its hyper-parameters and decides whether and how it would join the final pipeline? If so, how behaviour specification could be utilized to measure the quality of the pipelines? These are among the questions which could be explored.

References

[1] Doris Jung-Lin Lee, Stephen Marke, Doris Xin, Angela Lee, Silu Huang and Aditya Parameswaran. A human-in-the-loop perspective on AutoML. 2019. Bulletin of the IEEE Computer Society Technical Committee on Data Engineering.

[2] Bin Liu. A very brief and critical discussion on AutoML. 2018. arXiv:1811.03822.

[3] Nick Erickson, Jones Mueller, Alexander Shirkov, Hang Zhang, Pedro Larroy, Mu Li, Alexander Smola. AutoGluon-Tabular: Robust and Accurate AutoML for Structured Data. 2020. arXiv:2003.06505.

[4] Matthias Feurer, Katharina Eggensperger, Setfan Falkner, Marius Lindauer, Frank Hutter. Auto-Sklearn 2.0: Hands-free Automl via Meta-Learning. 2021. arXiv:2007.04074.

[5] Matthias Feurer, Katharina Eggensperger, Setfan Falkner, Marius Lindauer, Frank Hutter. Practical Automated Machine Learning for the AutoML Challenge 2018. 2018. ICML 2018 AutoML Workshop.

[6] Marc-andre Zoller, Marco F. Huber. Benchmark and Survey of AutoML Frameworks. 2019. Journal of Artificial Intelligence Reserch 70.

[7] Yang Li, Yu Shen, Wentao Zhang, Jiawei Jiang, Bolin Ding, Yaling Li, Jingren Zhou, Zhi Yang, Wentao Wu, Ce Zhang, Bin Cui. VolcanoML: Speeding up End-to-End AutoML via Scalable Search Space Decomposition. 2021. arXiv:2107.08861.

[8] Doris Xin, Eva Yiwei Wu, Doris Jung-lin Lee. Whither AutoML? Understanding the Role of Automation in Machine Learning Workflows. 2021. CHI '21, May 8-13, 2021.

[9] Fatjon Zogaj, Jose Pablo Cambronero, Martin C. Rinard, Jurgen Cito. Doing More with Less : Characterizing Dataset Downsampling for AutoML. 2021. ETH Library.

[10] Isabelle Guyon, Imad Chaabane, Hugo Jair Escalante, Sergio Escalera, Damir Jajetic, James Robert Lloyd, Nuria Macia, Bisakha Ray, Lukasz Romaszko, Michiele Sebag, Alexander Statnikov, Sebastien Treguer, Evelyne Viegas. A brief Review of the ChaLearn AutoML Challenge. 2016. JMLR: Workshop and Conference Proceedings 64:21-30.

[11] Joseph Giovanelli, Sesim Bilalli, Alberto Abello. Effective data pre-processing for AutoML. 2021. CEUR-WS.org/vol-2840/paper1.pdf.

[12] Chao Xue, Mengting Hu, Xueqi Huang, Chun-Guang Li. Automated search space and search strategy selection for AutoML. 2022. Pattern Recognition 124(2022) 108474.

[13] Catherine Wong, Neil Houlsby, Yifeng Lu, Andrea Gesmundo. Transfer Learning with Neural AutoML. 2018. NeurIPS 2018.

[14] Chi Wang, Qingyun Wu, Markus Weimer, Erkang Zhu. FLAML: A FAST AND LIGHTWEIGHT AUTOML LIBRARY. 2021. Proceedings of the 4th MLSys Conference.

[15] Chao Xue, Junchi Yan, Rong Yan, Stephen M. Chu, Yonggang Hu, Yonghua Lin. Transferable AutoML by Model Sharing Over Grouped Datasets. CVPR.

[16] Bochao Wang, Hang Xu, Jiajin Zhang, Chen Chen, Xiaozhi Fang, Yixing Xu, Ning Kang, Lanqing Hong, Chenhan Jiang, Xinyue Cai, Jiawei Li, Fengwei Zhou, Yong Li, Zhicheng Liu, Xinghao Chen, Kai Han, Han Shu, Dehua Song, Yunhe Wang, Wei Zhang, Chunjing Xu, Zhenguo Li, Wenzhi Liu, Tong Zhang. VEGA- TOWARDS AN END-TO-END CONFIGURABLE AUTOML PIPELINE. 2020. arXiv:2011.01507

[17] E.LeDell, S.Poirier. H2O AutoML: Scalable Automatic Machine Learning. 2020. 7th ICML Workshop on Automated Machine Learning.

[18] Herlalaina Rakotoarison, Louisot Milijaona, Andry Rasoanaivo, Michele Sebag, Marc Schoenauer. Learning Meta Features for AutoML. 2022. ICLR

[19] Jose P. Cambronero, Jurgen Cito, Jurgen Cito. AMS: Generating AutoML Search Spaces from Weak Specifications. 2020. ESEC/FSE '20.

[20] Piali Das, Nikita Ivkin, Tanya Bansal, Laurence Rouesnel, Philip Gautier, Zohar Karnin, Leo Dirac, Lakshmi Ramakrishnan, Andre Perunicic, Iaroslav Shcherbatyi, Wilton Wu, Aida Zolic, Huibin Shen, Amr Ahmed, Fela Winkelmolen, Miroslav Miladinovic, Cedric Archembeau, Alex Tang, Bhaskar Dutt, Patricia Grao, Kumar Venkateswar. Amazon SageMaker Autopilot: a white box AutoML solution at scale. 2020. DEEM '20.

[21] Sijia Liu, Parikshit Ram, Deepak Vijaykeerthy, Djallel Bouneffouf, Gregory Bramble, Horst Samulowitz, Dakuo Wang, Andrew Conn, Alexander Gray. An ADMM Based Framework for AutoML Pipeline Configuration. 2020. AAAI-20.

[22] Pieter Gijsbers, Erin LeDell, Janek Thomas, Sebastien Poirier, Bernd Bischl, Joaquin Vanschoren. An Open Source AutoML Benchmark. 2019. arXiv:1907.00909.

[23] Chengrun Yang, Jicong Fan, Ziyang Wu, Madeleine Udell. AutoML Pipeline Selection: Efficiently Navigating the Combinatorial Space. 2020. KDD '20.

[24] Iddo Drori, Lu Liu, Yi Nian, Sharath C. Koorathota, Jie S. Li, Antonio Khalil Moretti, Juliana Freire, Madeleine Udell. AutoML using Metadata Language Embeddings. 2019. arXiv:1910.03698.

[25] Qingyun Wu, Chi Wang, John Langford, Paul Mineiro, Marco Rossi. ChaCha for Online AutoML. 2021. arXiv:2106.04815

[26] Esteban Real, Chen Liang, David R. So, Quoc V. L. AutoML-Zero: Evolving Machine Learning Algorithms From Scratch. 2020. Proceedings of the 37th International Conference on Machine Learning.

[27] Jason Yoo, Tony Joseph, Dylan Yung, S. Ali Nasseri, Frank Wood. Ensemble Squared: A Meta AutoML System. 2021. arXiv:2012.05390.

[28] Jason Liang, Elliot Meyerson, Babak Hodjat, Dan Fink, Karl Mutch, and Risto Miikkulainen. Evolutionary Neural AutoML for Deep Learning. 2019. arXiv:1902.06827.

[29] Ziqiao Weng. From Conventional Machine Learning to AutoML. 2019. CCEAI 2019.

[30] Chengrun Yang, Yuji Akimoto, Dae Won Kim, Madeleine Udell. Oboe: Collaborative Filtering for AutoML Model Selection. 2019. KDD '19.

[31] Anatoly Yakovlev, Hesam Fathi Moghadam, Ali Moharrer, Jingxiao Cai, Nikan Chavoshi, Venkatanathan Varadarajan, Sandeep R. Agrawal, Sam Idicula, Tomas Karnagel, Sanjay Jinturkar, Nipun Agarwal. Oracle AutoML: a fast and predictive automl pipeline. 2020. Proceedings of the VLDB Endowment, Vol. 13, No. 12.

[32] Koen van der Blom, Alex Serban, Holger Hoos, Joost Virser. AutoML adoption in ML software. 2021. 8Th Icml Workshop On Automated Machine Learning.

[33] Ren Pang, Zhaohan Xi, Shouling Ji, Xiapu Luo, Ting Wang. On the Security Risks of AutoML. 2021. arXiv:2110.06018.

[34] Iordanis Xanthopoulos, Ioannis Tsamardinos, Vassilis Christophides, Eric Simon, Alejandro Salinger. Putting the Human Back in the AutoML Loop. 2020. Workshop Proceed- ings of the EDBT/ICDT 2020 Joint Conference.

[35] Tuomas Halvari Jukka K. Nurminen, Tommi Mikkonen. Testing the Robustness of AutoML Systems. 2020. First Workshop on Agents and Robots for reliable Engineered Autonomy 2020.

[36] Oleg Bezrukavnikov, Rhema Linder. Evaluating the Promises of Automatic Machine Learning Tools. 2021. arXiv:2101.05840.

[37] Mitar Milutinovic, Brandon Schoenfeld, Diego Martinez-Garcia, Saswati Ray, Sujen Shah, David Yan. On Evaluation of AutoML Systems. 2020. 7th ICML Workshop on Automated Machine Learning.

[38] Guillaume Baudart, Martin Hirzel, Kiran Kate, Parikshit Ram, Avraham Shinnar, Jason Tsay. Pipeline Combinators for Gradual AutoML. 2021. 35th Conference on Neural Information Processing Systems.

[39] Matthias Feurer, Frank Hutter. Towards Further Automation in AutoML. 2018. ICML 2018 AutoML Workshop.

[40] Florian Pfisterer, Janek Thomas, Bernd Bischl. Towards Human-Centered AutoML. 2019. arXiv:1911.02391.