Analysis on Approaches and Structures of Automated Machine Learning Frameworks

Peixuan Ge
Department of Computer Science
Virginia Tech
Blacksburg, VA, 24061
gpeixuan@vt.edu

***Abstract*—Due to the explosive and increasing demands of machine learning, automated machine learning is developed to handle machine learning tasks for non-experts. Lots of AutoML frameworks are introduced in past decades. Each of them has their unique contributions towards AutoML field. In this paper, four popular open source AutoML frameworks are selected and reviewed to show current development directions and common features for frameworks. This paper also analyzes innovative structures and designs of selected frameworks. The result shows that one of the newest frameworks, AutoGluon, has extraordinary performance and innovative structures when compared to others.**

Keywords-Automated Machine Learning; Bayesian Optimization; Algorithm Selection and Hyperparameter Optimization; AutoML; Auto-sklearn; TPOT; Auto-Keras; AutoGluon-Tabular

I. INTRODUCTION

With the growing needs on machine learning (ML) in industrials and companies, automated machine learning (AutoML) frameworks are created to provide more effective and convenient solutions for non-machine learning experts. Current AutoML frameworks are based on the existing ML modules and algorithms. Rather than traditional ML approach, AutoML greatly simplified the complexity by providing much simpler approach on hyperparameter optimization and algorithm selection with integrated ML libraries. There are a lot of AutoML frameworks implemented by different structures to address problems in various fields. This review will look over approaches and structures of some well-known open source AutoML frameworks and provide suggestions for real life applications by reviewing the official documents, peer reviews and benchmarks in recent years.

II. SELECTED AUTOML FRAMEWORKS

A. Auto-sklearn

Auto-sklearn is an AutoML framework built on the scikit-learn. It is one of the earliest successful AutoML frameworks that creates a ML pipeline and provides directions for future studies. Its pipeline structure is showing as the figure 1 [1].

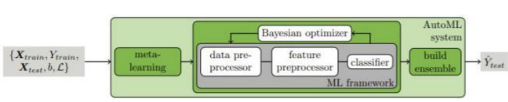

Figure 1. Structure of Auto-sklearn [1]

The core part is Bayesian optimization procedure and the development team makes two major improvements for it. First, the pipeline integrates a meta-learning module as a complementary to warm up for Bayesian optimizer and improve the performance when choosing selected ML framework. Second, there is an ensemble construction module served as a post-procedure to improve hyperparameter optimization and make the whole framework more robust [1].

Auto-sklearn can free people from tedious hyperparameter optimization and algorithm selections. Because it is focusing on scikit-learn, and it can have good compatibility with datasets that have already tested on scikit-learn. This is also a disadvantage for it, however, since scikit-learn focuses on small and medium datasets, Auto-sklearn is not well performed on large datasets and modern deep learning systems as well.

B. TPOT

Tree-based Pipeline Optimization Tool (TPOT) uses genetic programming to build and optimize AutoML pipelines. It is designed to make ML tools to be more accessible and automated. Their developers call it as a "Data Science Assistant" [2].

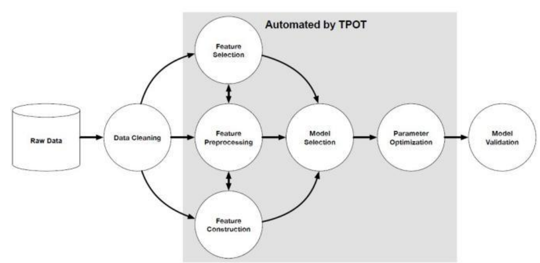

Figure 2. Structure of TPOT [2]

Figure 2 shows the structure and modules of TPOT. TPOT uses existing scikit-learn implementations to build pipeline operators, including a series of preprocessing, feature selection and classification methods. It uses those operators to build tree-based pipelines and uses genetic programming to optimize and improve classification accuracy. According to the creators of TPOT, the work of TPOT is essential to fully establish automating machine learning pipeline design [2].

TPOT is a continuous evolving framework. Its developers have done some great work to improve it. Paper by Olson's group has proved that Pareto optimization can be used in TPOT to avoid classification accuracy loss [2]. In the paper by Le's group, the author introduced Feature Set Selector and Template to TPOT. Feature Set Selector can significantly improve the efficiency of TPOT on big datasets, which makes up for the integrated scikit-learn tool. Template can help

978-1-7281-9761-6/20/$31.00 ©2020 IEEE
DOI 10.1109/CISCE50729.2020.00106

TPOT to use Feature Set Selector. Those new features have improved scalability and reduced running time for TPOT [3].

TPOT is designed to help users find a decent pipeline for a dataset. Users can configure pipeline searching by limiting the range of hyperparameters and algorithms. Users can also set the population size and generations to limit the searching time. Since TPOT needs long time to run for larger datasets, it provides interrupt and warm start functions to allow users to pick up at where it was paused. TPOT also has an important feature that allows users to export the pipeline and cooperate with their own ideas and implementations.

C. *Auto-Keras*

Auto-Keras is an AutoML framework built on Keras with the implementation of neural architecture search. It is also designed to simplify the process to use ML techniques for non-experts as other AutoML frameworks, and it is more focused on deep learning tasks. It is also specially optimized for local computing, because the development team believes this would be helpful for users who are not comfort with cloud programming. It has a well-designed parallel workflow that using both CPU and GPU at the same time to provide a decent performance.

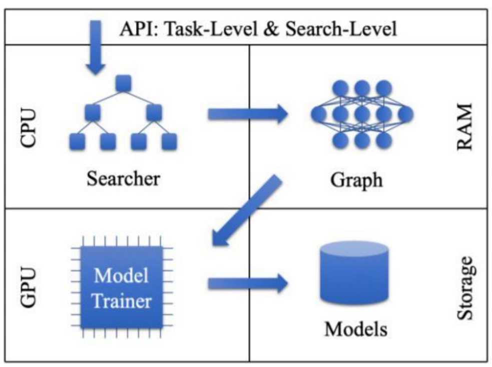

Figure 3. Structure of Auto-Keras [4]

Figure 3 shows the structure of Auto-Keras. After users called the API, CPU would run the Searcher module which uses Bayesian Optimizer to generate new architecture. At the same time, GPU runs the model trainer to train dataset with that architecture. CPU will transform the architecture into a neural network and put that into the RAM for faster speed. When that training is done, the model will be saved in storage and performance data is sent back to CPU to update Gaussian process. This kind of parallel workflow can significantly reduce the idle time and improve the efficiency. The development team states Auto-Keras uses Bayesian optimization to search for neural networks and has better performance than traditional hyperparameter-tuning and neural architecture search methods. They also state the development directions for Auto-Keras are expanding search space with recurrent neural networks, combining tuning process, and design task-oriented neural architecture search system [4].

Auto-Keras is designed to have a simple and non-expert friendly interface. Compared to other AutoML frameworks, it is more focused on deep learning. It can automatically search for a good neural architecture for dataset with optimizations for parallel workflow. It also supports restoring a previous search and exporting trained model as a Keras Model. For flexibility, it allows users to customize model and search space. It now only has a few supported tasks, but it is continuous growing. There is a limitation about GPU memory management for this framework. It has a declining upper limit for size of generated networks as more networks generated, in that the framework would crash if the GPU memory is running out.

D. *AutoGluon-Tabular*

AutoGluon-Tabular (AutoGluon) is one of the most recent released AutoML frameworks. It is a revolutionary framework, initiating a direction towards AutoML rather than traditional algorithm selection and hyperparameter optimization (CASH) problem[1]. AutoGluon is using tabular data to build a more efficient framework and to solve problems that other AutoML frameworks cannot handle. According to Erickson's team, AutoGluon follows simplicity, robustness, fault tolerance and predictable timing principles. Those principles are achieved by different modules. AutoGluon provides the fit API that users can just use three lines of code to launch a training task. It has a data preprocessing module with model-agnostic preprocessing and model-specific preprocessing stages. This module will help AutoGluon to predict the problem and clean the raw data. AutoGluon uses a few ordered preset models to ensure models with more reliability are trained first and users can also add their own models. The development team also claim that the neural network of AutoGluon is the first AutoML framework that uses per-variable embeddings to improve their resulting quality [5].

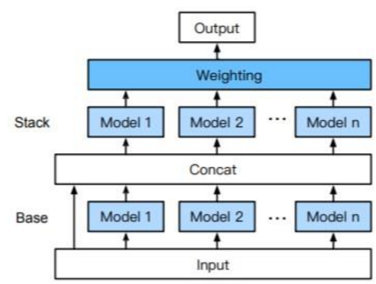

Figure 4. AutoGluon's multi-layer stacking strategy [5]

AutoGluon uses a multi-layer stacking strategy shown by figure 4. In this strategy, the stack has multiple layers. The first layer has some models and their outputs are concatenated as the base for next layer. AutoGluon reuses all of those model types layers by layers which can allow higher layers to access the original data features. The final layers in the stack innovatively applies ensemble selection which restrains over-fitting. AutoGluon also uses k-fold bagging method to further improve the stacking performance and use repeated bagging process to ease over-fitting. AutoGluon uses a different strategy to train models. It will save trained models immediately to disk that allows it to train within time limits and predict if a model may fail and skip that model by result. It also uses a different workflow strategy that trains models in order and relies on models' multi-task optimization to prevent potential memory management errors [5].

AutoGluon uses its innovative design to prove that there are more focuses rather than CASH problem for AutoML frameworks. It has many features that other frameworks do not implement. Its preprocessing model can handle heterogeneous datasets. It has a high-performance multi-layer stacking design combined with advanced neural network structure, comparing with the traditional Bayesian optimization design. At the same time, it keeps flexibility and not increasing complexity to use. It allows users to import their own model and provides simple API that also allows users to resume training.

III. BENCH-MARKING RESULTS

One important evaluation for AutoML frameworks is the performance. For Auto-sklearn, TPOT and AutoGluon, they are beating their previous frameworks on the performance part in their own released paper. There are no benchmarks for Auto-Keras comparing to other AutoML frameworks released in their paper, but they proved their method was effective through their experiment.

Here are two results of the most popular AutoML benchmarks and the benchmark data provided in the AutoGluon paper. Those benchmark results can provide an overview of which frameworks are better at which datasets.

Following figures are benchmark results provided by A. Balaji and A. Allen's team [6].

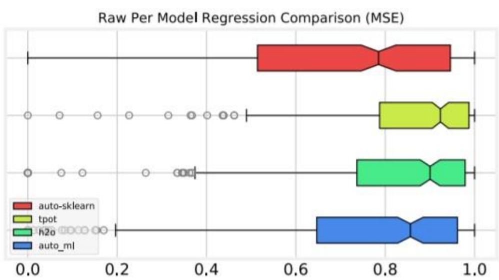

Figure 5. A. Balaji and A. Allen's benchmarking for classification datasets [6].

Figure 6. A. Balaji and A. Allen's benchmarking for regression datasets [6].

From those two graphs, auto-sklearn has the best classification score and TPOT has the best regression score comparing with auto_ml and H2O AutoML frameworks.

There is also an open source benchmark platform called AutoML Benchmarking developed by P. Gijsbers's team [7]. Figure 7 shows the benchmark results that they tested on various AutoML frameworks. Those results indicate that there is no framework can always have the best performance score across all the datasets.

Figure 8 is a part of AutoML benchmark results, relative to AutoGluon provided by AutoGluon's development team. The result indicates that AutoGluon is beating other frameworks.

Because the focus of this paper is to review approaches of frameworks, and their approaches are continuously improving, benchmarks here can only be a simple view for performance status of those frameworks.

Framework: Binary tasks:	auto-sklearn	Auto-WEKA	H2O AutoML	RandomForest	TPOT
adult	1.045	1.000	1.049	1.000	1.048
airlines	1.403	1.016	1.435	0.997	1.343
albert	1.009		1.115	1.001	0.981
amazon_employee...	0.972*	0.886	1.048	1.003	1.012
apsfailure	1.000	0.985	1.001	1.000	1.001
australian	1.010	1.015	0.909	1.010	1.011
bank-marketing	1.012	0.950	1.015	1.000	1.008
blood-transfusion	1.495	1.379	1.532	0.985	1.149
christine	1.072	0.998	1.048	0.988	1.029
credit-g	0.970*	0.829	0.991	1.004	0.924
guiellermo	1.004	0.934	1.024	0.999	0.878
higgs	1.018*	0.845	1.041	0.999	1.005
jasmine	0.987	0.939	1.001	0.998	1.004
kc1	0.999*	0.934	0.992	0.987	1.013
kddcup09_appetency	1.181*	1.043	1.176	1.016	1.134
kr-vs-kp	1.000*	0.959	1.000	0.999	0.999
miniboone	1.008	0.957	1.010	0.999	1.001
nomao	1.002	0.973	1.002	1.000	1.001
numerai28.6	1.679	1.544	1.730	1.042	1.428
phoneme	0.993*	0.998	1.005	1.000	1.015
riccardo	1.000	0.996	1.000	0.999	0.992
sylvine	1.013	0.985	1.011	0.999	1.023
Multi-class tasks:					
car	1.030	0.906	1.060	0.878	1.060
cnae-9	1.069	0.541	1.076	0.999	1.057
connect-4	1.184	-1.565	1.409	0.954	1.276
covertype	0.976	-0.361	0.856	0.944	0.933
dilbert	1.182	0.459	1.205	0.979	1.111
dionis	0.580	0.590		1.002	
fabert	1.026	-5.235	1.049	1.004	1.005
fashion-mnist	0.995	0.717	1.052	0.993	0.841
helena	0.660	-18.420	1.905	0.970	1.676
jannis	1.083	-1.989	1.065	0.973	0.987
jungle_chess...	1.299	-3.309	1.235	0.933	1.459
mfeat-factors	1.059*	0.789	1.053	0.992	1.018
robert	-0.001		1.545	1.000	0.640
segment	1.004	0.808	1.012	0.992	1.008
shuttle	1.000	0.979	1.000	1.000	1.000
vehicle	1.102	-4.630	1.166	0.986	1.099
volkert	1.002	-5.585	1.111	0.954	0.945

Figure 7. Benchmark provided by tool developed by Pieter Gijsbers's team. Scores are scaled "between a constant class prior predictor (=0) and a tuned random forest (= 1). Missing values mean that no results were returned in time. * the task was also included in meta-learning models [7]."

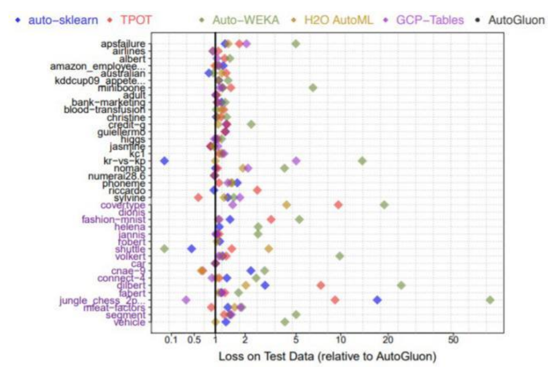

Figure 8. AutoML frameworks benchmark results relative to AutoGluon provided by AutoGluon's development team [5].

IV. DISCUSSION

AutoML is becoming popular in recent years due to the growing needs on ML and this field has started developing since several years ago. There are many fundamental AutoML frameworks which have made great contributions to this field like Auto-WEKA. They introduced basic ideas and direction of AutoML. The classic approach for AutoML focuses on algorithm selection and hyperparameter optimization which is the CASH problem. AutoML frameworks usually apply Bayesian optimization to implement the selection process and add their new features to improve the efficiency and extend functions. At the same time, every AutoML framework tries to simplify their API as much as possible, because they target on non-machine learning experts. Some of frameworks can also provide flexibility that allows advanced users to set their own models and algorithms to select.

Each of the selected framework shows their contributions to AutoML field. For Auto-sklearn, it has introduced ML pipeline design with meta learning and ensemble constructions to improve the efficiency and robustness of AutoML. According to the official document of Auto-sklearn, it only requires a simple process to install and start training.

However, it still needs adjustments if someone wants to use it on various datasets. In addition, it is not effective when training big datasets. TPOT introduces generic programming and tree structure to build ML pipelines and can search for a large range of hyperparameters and ML algorithms. Users can pause and resume a search process then export the tuned pipeline. Many developers are improving TPOT like recent research indicates the Feature Set Selector can help TPOT on big dataset training. Auto-Keras has introduced a new algorithm based on network morphism to search for the machine learning models. It is specialized for the parallel workflow of CPU and GPU to improve the efficiency. AutoGluon is one of the latest and well-developed frameworks that has open sourced. Instead of focusing on traditional CASH problem, it has innovatively introduced tabular data processing and multi-layer stacking structure for ensembling. It also combines with advanced neural network and k-fold bagging. Those innovations allow it to have great performance when comparing with other frameworks without losing robustness and flexibility or increasing using complexity.

By looking through these frameworks, it is not hard to see that they are all sharing the same development directions which is to improve the effectiveness and accuracy of searching algorithms, to keep the system robustness, to increase more user-friendly features and to simplify user interface. Popular and practical features that those frameworks have implemented include pause/resume for searching, export the search result or pipeline built, allow users to set their own searching range including hyperparameter, algorithms and models, preprocessing data for error tolerance, determine task type based on input data. Currently some of frameworks only have limited task class on classification and regression, but selected frameworks are still having an active development group. Therefore, it is hopeful that they could have more task classes and practical features supported in the future.

This paper has some shortages. As many great AutoML frameworks like H2O did not provide detailed papers or explanations for their approaches yet and some frameworks are not open sourced, this paper only selected a few frameworks with detailed papers published.

V. CONCLUSIONS

AutoML frameworks are designed to provide an easy way to use interface and high-performance machine learning pipeline for non-machine learning experts. This paper provides a review of current popular open source AutoML frameworks. Four outstanding frameworks are selected to show current direction and popular features of AutoML frameworks which are Auto-sklearn, TPOT, Auto-Keras and AutoGluon. Each of them has distinct and innovative contributions to this field. To review them, it can be found that they are sharing common ideas and objectives that make their approach be effective, robust, and accuracy. At the same time, they are also designed with user-friendly features to encourage more people to participate in the field of machine learning. In addition, improvements of those frameworks also represent the evolution of AutoML frameworks design focus that is from traditional CASH problem to each of detailed AutoML pipeline components. This paper has shortages that it only includes four AutoML frameworks and covers their recent breakthroughs. Because many great frameworks have not published their research results yet, the future research will cover more frameworks and breakthroughs.

ACKNOWLEDGMENT

First and foremost, I would like to show my deepest gratitude to my teachers and professors in my university, who have provided me with valuable guidance in every stage of the writing of this thesis. Further, I would like to thank my friends and parents for their encouragement and support. Without all their enlightening instruction and impressive kindness, I could not have completed my thesis.

REFERENCES

[1] M. Feurer, A. Klein, K. Eggensperger, J. Springenberg, M. Blum and F. Hutter, "Efficient and Robust Automated Machine Learning," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama and R. Garnett, Eds., Curran Associates, Inc., 2015, p. 2962–2970.

[2] R. S. Olson, N. Bartley, R. J. Urbanowicz and J. H. Moore, "Evaluation of a Tree-based Pipeline Optimization Tool for Automating Data Science," in *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, New York, NY, USA, 2016.

[3] T. T. Le, W. Fu and J. H. Moore, "Scaling tree-based automated machine learning to biomedical big data with a feature set selector," *Bioinformatics,* vol. 36, p. 250–256, 2020.

[4] H. Jin, Q. Song and X. Hu, "Auto-Keras: An Efficient Neural Architecture Search System," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019.

[5] N. Erickson, J. Mueller, A. Shirkov, H. Zhang, P. Larroy, M. Li and A. Smola, "AutoGluon-Tabular: Robust and Accurate AutoML for Structured Data," *arXiv preprint arXiv:2003.06505,* 2020.

[6] A. Balaji and A. Allen, "Benchmarking Automatic Machine Learning Frameworks," *CoRR,* vol. abs/1808.06492, 2018.

[7] P. Gijsbers, E. LeDell, S. Poirier, J. Thomas, B. Bischl and J. Vanschoren, "An Open Source AutoML Benchmark," *arXiv preprint arXiv:1907.00909 [cs.LG],* 2019.