

# Visualization and Interaction with Automated Data Science Processes

Klaus Eckelt, Sheeba Samuel, David Koop, Kiran Gadhane, Dominik Moritz

September 10–15, 2023

Automated Machine Learning (AutoML) and Data Science (AutoDS) pipelines have gained significant attention in recent years. This working group focused on interactive systems to track and visualize the provenance of these pipelines and enable user interaction with automated algorithms as they are running. To gain an overview of the state-of-the-art, we reviewed related work that compares AutoML/DS libraries [1] and applied multiple of them to the same tabular data set. We analyzed the information they provide on the ongoing optimization process, the search space, and their final result (see Table 1). We also compared the resulting models by their complexity and accuracy. All the analyzed libraries provide logs in the console and optionally in a file. Our analysis included Auto-sklearn [2, 3], AutoGluon [4], TPOT [5], FLAML [6], and H2O AutoML [7]. These libraries were used within Jupyter Notebook using Python.<sup>1</sup>







Library	Downloads	Log	Search & Optimization Strategy	Accuracy
Auto-sklearn	25,000	>_ 	CASH* with Bayesian Optimization	95.35 %
Auto-sklearn 2.0	25,000	>_ 	PoSH† and Bayesian Optimization	93.07 %
AutoGluon	41,000	>_ 	Model Portfolio and Random Search	94.30 %
TPOT	35,000	>_ 	CASH with Genetic Programming	99.92 %
FLAML	189,000	>_ 	CASH with Cost-Frugal Optimization	98.89 %
H2O AutoML	340,000	>_	CASH with Random Search	94.44 %

Table 1: Overview of the considered AutoML libraries. Downloads were retrieved from PyPI Stats for the last 30 days [8]. The runtime limit was set to 30 minutes. All pipelines were trained with default settings, but maximized CPU utilization and logging outputs (in the console (>\_) and file ()). \* CASH... Combined Algorithm Selection and Hyperparameter Optimization; † PoSH... Portfolio Successive Halving

Although some AutoML/DS users have extensive domain knowledge, they may have limited knowledge in the realm of machine learning, and conversely, individuals with a strong machine learning background may lack expertise in the specific domain of the data [9]. As a result, different user groups require different levels of detail and information in visualizing the AutoML/DS process. Related work also differs in terms of detail presented and potential target users. While partial dependence or parallel coordinate plots are easily interpretable, tools such as PipelineProfiler [10], ATMSeer [11], or DeepCave [12] allow for a more detailed inspection, but also require technical understanding of the parts of the machine learning pipeline and its optimization.

AutoML/DS approaches and the tools to visualize them currently provide little room for human interaction. The only way to steer the algorithms is by setting parameters before starting an(other) optimization run. But for optimal performance, AutoGluon, for example, advises against human intervention in its documentation<sup>2</sup> and Auto-sklearn 2.0 also removed the human from the loop [3]. Even more steps of the AutoML/DS pipeline will be automated in the future [13].

However, recent research argues for the necessity of reintegrating users into the loop [14]. Given that AutoML/DS requires time to run, it is essential to allow adaptations to make efficient use of this

<sup>1</sup><https://github.com/keckelt/dagstuhl-23372-applications>

<sup>2</sup><https://auto.gluon.ai/stable/tutorials/tabular/tabular-essentials.html#maximizing-predictive-performance>

time. Interactive visualization systems can help identify issues early on. For instance, a label left in the training data could cause unusually high performance across all configurations, or poor performance could be due to insufficient data quality. These systems would also enable users to make adjustments to the performance metric, to trade-off between sensitivity and specificity, for example. Providing users with more control when necessary can increase their trust in the AutoML/DS system, often perceived as a ‘black box’, and could also speed up and improve the process by allowing users to contribute their knowledge more effectively. Meta-learning—i.e., learning from previous experiments—is currently only supported on the machine side of the loop. Auto-sklearn 2.0 [3], for example, looks for similar problems on OpenML [15] to learn from them. However, AutoML/DS systems do not allow users to provide information on similar problems they have encountered. Allowing users to provide their knowledge to the optimization process could guide the search throughout the optimization process.

Figure 1 shows our sketch to visualize AutoML/DS system processes, compare them, and interact with them. Multiple runs can be selected at the top. Their progress and performance metric is displayed on the left side. The large table on the right gives an overview of the configurations that were trained over time. The individual runs are distinguished through different colors (■, ■, ■). If a configuration fails, we use the negative of the run’s color (■, ■, ■). These negatives are less saturated to better differentiate them from successful runs. As there are more configurations than can be displayed on the available vertical screen space, an aggregation method can be selected using the radio buttons on the top right. The best overall configuration per run is additionally highlighted with a colored horizontal bar across the entire line and a trophy symbol next to it. For a detailed inspection, the progress bar, accuracy plot, and table can be vertically expanded using a switch button to show each tested configuration without aggregation. This table can also be used for interaction with the AutoML/DS process. Users should be able to prioritize or block elements to be explored in order to guide the search space. Using an interactive table like LineUp [16] would also allow to filter and rank the configurations, and update performance metrics through combination and weighting of recorded information. However, such an interaction is currently not possible in any of the AutoML/DS systems we reviewed.



Figure 1: Our sketched visual interface to visualize AutoML/DS systems. An ongoing Auto-sklearn optimization is visualized in green. Additional past runs can be selected at the top for comparison. The table on the right shows the best performing configurations in the specific time segment.

We also found that none of the AutoML libraries we tried support MLOps services like MLflow [17] or Weights and Biases [18], which are frequently used to track the training and optimization process of machine learning projects. A custom logging configuration can be passed to Auto-sklearn. All other libraries were only able to log into files instead. We wanted to visualize the AutoML/DS process in

real-time while it is running, and thus defined our own logger for Auto-sklearn that also sends all output to Weights and Biases.<sup>3</sup> As this approach heavily relies on log data and parsing string outputs it is limited and error prone. We also noted that these MLOps services do not support the process of tracking AutoML/DS optimizations well, due to the pipeline’s many different elements and their parameters.

MLflow or Weights and Biases do not store the recorded information in any standardized or interoperable format, such as PROV-ML [19], which is based on W3C PROV [20]. With *mlflow2prov* [21], data from MLflow and the versioned source code from which it originates can be combined into another provenance format based on W3C PROV.

In addition to the AutoML/DS provenance, a common format to describe the tracked data is necessary. Pipeline elements, their naming, and possible combinations vary between AutoML/DS tools. To ensure that interactive systems are interoperable between AutoML/DS tools, they require a common standard to communicate and store (intermediate) results [22]. This would allow users to visualize and compare the results of different AutoML/DS systems beyond the performance metrics. We argue that AutoML/DS systems require hooks with which intermediate results are communicated and APIs to steer the ongoing process. We believe the adoption of a standardized provenance format, which can be shared between different data science tools, would facilitate comparisons and allow better monitoring, debugging, interpretation, and explanation of the process.

## References

- [1] L. Ferreira, A. Pilastri, C. M. Martins, P. M. Pires, and P. Cortez, “A Comparison of AutoML Tools for Machine Learning, Deep Learning and XGBoost,” in *2021 International Joint Conference on Neural Networks*, pp. 1–8, July 2021.
- [2] M. Feurer, A. Klein, K. Eggenberger, J. Springenberg, M. Blum, and F. Hutter, “Efficient and Robust Automated Machine Learning,” in *Advances in Neural Information Processing Systems 28 (2015)*, pp. 2962–2970, 2015.
- [3] M. Feurer, K. Eggenberger, S. Falkner, M. Lindauer, and F. Hutter, “Auto-sklearn 2.0: hands-free AutoML via meta-learning,” *The Journal of Machine Learning Research*, vol. 23, pp. 261:11936–261:11996, Jan. 2022.
- [4] 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.
- [5] 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, no. 1, pp. 250–256, 2020.
- [6] C. Wang, Q. Wu, M. Weimer, and E. Zhu, “FLAML: A Fast and Lightweight AutoML Library,” *Proceedings of Machine Learning and Systems*, vol. 3, pp. 434–447, 2021.
- [7] E. LeDell and S. Poirier, “H2O AutoML: Scalable Automatic Machine Learning,” *7th ICML Workshop on Automated Machine Learning*, July 2020.
- [8] C. Flynn, “PyPI Download Stats.” <https://pypistats.org/>, 2023. Accessed: 2023-09-18.
- [9] A. Crisan, B. Fiore-Gartland, and M. Tory, “Passing the Data Baton: A Retrospective Analysis on Data Science Work and Workers,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, no. 2, pp. 1860–1870, 2021.
- [10] J. P. Ono, S. Castelo, R. Lopez, E. Bertini, J. Freire, and C. Silva, “PipelineProfiler: A Visual Analytics Tool for the Exploration of AutoML Pipelines,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, pp. 390–400, Feb. 2021.

---

<sup>3</sup><https://wandb.ai/dagstuhl-23372/automl>

- [11] Q. Wang, Y. Ming, Z. Jin, Q. Shen, D. Liu, M. J. Smith, K. Veeramachaneni, and H. Qu, “ATMSeer: Increasing Transparency and Controllability in Automated Machine Learning,” in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, (New York, USA), pp. 1–12, ACM, 2019.
- [12] R. Sass, E. Bergman, A. Biedenkapp, F. Hutter, and M. Lindauer, “DeepCAVE: An Interactive Analysis Tool for Automated Machine Learning,” in *ICML Workshop on Adaptive Experimental Design and Active Learning in the Real World*, June 2022.
- [13] S. K. Karmaker (“Santu”), M. M. Hassan, M. J. Smith, L. Xu, C. Zhai, and K. Veeramachaneni, “AutoML to Date and Beyond: Challenges and Opportunities,” *ACM Computing Surveys*, vol. 54, no. 8, pp. 175:1–175:36, 2021.
- [14] M. Lindauer and A. Tornede, “Rethinking AutoML: Advancing from a Machine-Centered to Human-Centered Paradigm.” <https://www.automl.org/rethinking-automl-advancing-from-a-machine-centered-to-human-centered-paradigm/>, 2022. Accessed: 2023-09-18.
- [15] J. Vanschoren, J. N. van Rijn, B. Bischl, and L. Torgo, “OpenML: networked science in machine learning,” *SIGKDD Explorations*, vol. 15, no. 2, pp. 49–60, 2013.
- [16] S. Gratzl, A. Lex, N. Gehlenborg, H. Pfister, and M. Streit, “LineUp: Visual Analysis of Multi-Attribute Rankings,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2277–2286, 2013.
- [17] M. Zaharia, A. Chen, A. Davidson, A. Ghodsi, S. A. Hong, A. Konwinski, S. Murching, T. Nykodym, P. Ogilvie, M. Parkhe, and others, “Accelerating the Machine Learning Lifecycle with MLflow,” *IEEE Data Engineering Bulletin*, vol. 41, no. 4, pp. 39–45, 2018.
- [18] L. Biewald, “Experiment Tracking with Weights and Biases.” <https://www.wandb.com/>, 2020. Accessed: 2023-09-18.
- [19] R. Souza, L. Azevedo, V. Lourenço, E. Soares, R. Thiago, R. Brandão, D. Civitarese, E. Brazil, M. Moreno, P. Valduriez, M. Mattoso, R. Cerqueira, and M. A. Netto, “Provenance Data in the Machine Learning Lifecycle in Computational Science and Engineering,” in *2019 IEEE/ACM Workflows in Support of Large-Scale Science*, pp. 1–10, Nov. 2019.
- [20] P. Missier, K. Belhajjame, and J. Cheney, “The W3C PROV family of specifications for modelling provenance metadata,” in *Proceedings of the 16th International Conference on Extending Database Technology*, (New York, USA), pp. 773–776, ACM, 2013.
- [21] M. Schlegel and K.-U. Sattler, “Extracting Provenance of Machine Learning Experiment Pipeline Artifacts,” in *Advances in Databases and Information Systems* (A. Abelló, P. Vassiliadis, O. Romero, and R. Wrembel, eds.), Lecture Notes in Computer Science, pp. 238–251, Springer Nature Switzerland, 2023.
- [22] M. Milutinovic, *Towards Automatic Machine Learning Pipeline Design*. PhD thesis, EECS Department, University of California, Berkeley, Aug 2019.