The Journal of Open Source Software

AMLTK: A Modular AutoML Toolkit in Python

Edward Bergman [1¶], **Matthias Feurer** [2,3¶], **Aron Bahram** [1], **Amir Rezaei Balef** [4], **Lennart Purucker** [1], **Sarah Segel** [5], **Marius Lindauer** [5,6¶], **Frank Hutter** [1¶], and **Katharina Eggensperger** [4¶]

1 University of Freiburg, Germany **2** LMU Munich, Germany **3** Munich Center for Machine Learning **4** University of Tübingen, Germany **5** Leibniz University Hannover, Germany **6** L3S Research Center, Germany ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Josh Borrow

Reviewers:

- @gomezzz
- @woznicak
- @hirzel

Submitted: 07 December 2023
Published: unpublished

Summary

Machine Learning is a core building block in novel data-driven applications. Practitioners face many ambiguous design decisions while developing practical machine learning (ML) solutions. Automated machine learning (AutoML) facilitates the development of machine learning applications by providing efficient methods for optimizing hyperparameters, searching for neural architectures, or constructing whole ML pipelines (Hutter et al., 2019). Thereby, design decisions such as the choice of modelling, pre-processing, and training algorithm are crucial to obtaining well-performing solutions. By automatically obtaining ML solutions, AutoML aims to lower the barrier to leveraging machine learning and reduce the time needed to develop or adapt ML solutions for new domains or data.

Highly performant software packages for automatically building ML pipelines given data, so-called AutoML systems, are available and can be used off-the-shelf. Typically, AutoML systems evaluate ML models sequentially to return a well-performing single best model or multiple models combined into an ensemble. Existing AutoML systems are typically highly engineered monolithic software developed for specific use cases to perform well and robustly under various conditions.

With the growing amount of data and design decisions for ML, there is also a growing need to improve our understanding of the design decisions of AutoML systems. Current state-of-the-art systems vary in implemented paradigms (stacking (Erickson et al., 2020) vs CASH (Thornton et al., 2013), optimizing a pre-defined pipeline structure (Thornton et al., 2013) vs evolving open-ended pipelines (Olson et al., 2016)) and also use different methods within one paradigm (i.e. Bayesian optimization (Feurer et al., 2015; Thornton et al., 2013) or Genetic Programming (Gijsbers & Vanschoren, 2019; Olson et al., 2016) as the optimization algorithm, different search spaces for the same machine learning algorithm cf. (Feurer et al., 2015; Gijsbers & Vanschoren, 2019; Olson et al., 2016; Thornton et al., 2013), different post-hoc ensemble methods or even no post-hoc ensembling at all cf. (Feurer et al., 2015; Imrie et al., 2022; Wang et al., 2021)), raising many research questions and opportunities to study improved algorithms and novel applications.

AMLTK (Automated Machine Learning ToolKit) is a collection of components that enable researchers and developers to easily implement AutoML systems without the need for common boilerplate code. AMLTK addresses this with a modular perspective on AutoML systems, aiming to cover various existing AutoML system paradigms in principle. It contributes to the field three-fold: (a) Enabling systematic comparison of AutoML design decisions with a higher level of reproducibility, (b) fast prototyping and evaluation of new AutoML methods, and (c) easy adaptation of developed solutions to new tasks.

In addition, it also facilitates easy integration and swapping of components from various

AutoML tools, for example, an optimizer from SMAC (Lindauer et al., 2022) or Optuna (Akiba et al., 2019), a search space from ConfigSpace (Lindauer et al., 2019) or Optuna, as well as the integration with additional tools such as the visualization and analysis tool DeepCAVE (Sass et al., 2022). These provided integrations are done without the need to modify AMLTK's source code, enabling users to extend the framework as their needs require. Overall, AMLTK lowers the barrier to engaging with AutoML research and, thus, opens up the opportunity to bundle research efforts towards flexible and effective AutoML systems.

AMLTK is designed for AutoML researchers to develop and study novel AutoML systems and domain experts to adapt these AutoML systems for novel use cases. AMLTK is based on the experience of a subset of the authors in developing AutoML systems (Auto-sklearn (Feurer et al., 2015, 2022) and Auto-PyTorch (Zimmer et al., 2021)) and their effort to unify their code bases. Last but not least, we also believe that this toolkit will help educate students and support ML practitioners in engaging with AutoML systems.

Statement of Need

Current AutoML systems are monolithic and provide little opportunity for customization. As a result, researchers often build new AutoML systems to implement a new methodology. This results in two issues: (1) it creates a barrier to research on AutoML systems, and (2) it hinders the fair comparison of new components in AutoML systems. Recent examples of open source AutoML systems are AutoGluon (Erickson et al., 2020), GAMA (Gijsbers & Vanschoren, 2021, 2019), and Auto-Sklearn (Feurer et al., 2015, 2022).

To give an example for Issue (1), a researcher working on new optimization methods for AutoML would need to develop all components of an AutoML system in order to evaluate their method because current systems do not allow for easy replacement of the optimization method, as pointed out by Mohr & Wever (2023). Also, a researcher wanting to study a variation of an existing system would need to go through an extensive, potentially undocumented codebase to find the right place to apply their variation. The tight integration of components allows for highly efficient systems but poses a high barrier to new research and novel, innovative AutoML systems.

Issue (2) is also a huge problem. A recent benchmark study (Gijsbers et al., 2023) extensively compared multiple AutoML systems on a common set of ML tasks. While such benchmarking efforts are necessary to assess the current state of the art, we note that each system uses its own implementation of the search space, optimization, evaluation and ensembling, making a principled comparison and ablation study virtually impossible and leaving potential performance gains by combining solutions unnoticed. Instead of comparing different methods, researchers are actually comparing the implementations. By providing a unified toolkit for AutoML, researchers can focus on comparing the changes they have made while leaving all other parts of the AutoML system as they were.

Acknowledgements

Edward Bergman was partially supported by TAILOR, a project funded by EU Horizon 2020 research and innovation programme under GA No 952215. Katharina Eggensperger and Amir Balef acknowledge funding by the German Research Foundation under Germany's Excellence Strategy - ECX number 2064/1 - Project number 390727645. Marius Lindauer acknowledges support by the Federal Ministry of Education and Research (BMBF) under the project AI service center KISSKI (grantno. 01IS22093C). Lennart Purucker acknowledges funding by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – SFB 1597 – 499552394. Sarah Segel acknowledges funding by the European Union (ERC, "ixAutoML", grant no. 101041029). Frank Hutter acknowledges funding by the European Union (ERC Consolidator Grant "DeepLearning 2.0", grant no. 101045765) Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency. Neither the European Union nor the granting authority can be held responsible for them.

References

Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). Optuna: A next-generation Hyperparameter Optimization framework. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD'19)*, 2623–2631.

Erickson, N., Mueller, J., Shirkov, A., Zhang, H., Larroy, P., Li, M., & Smola, A. (2020). AutoGluon-tabular: Robust and accurate AutoML for structured data. *arXiv:2003.06505 [Stat.ML]*.

Feurer, M., Eggensperger, K., Falkner, S., Lindauer, M., & Hutter, F. (2022). Auto-Sklearn 2.0: Hands-free AutoML via meta-learning. *Journal of Machine Learning Research*, *23*(261), 1–61.

Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., & Hutter, F. (2015). Efficient and robust automated machine learning. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, & R. Garnett (Eds.), *Proceedings of the 29th international conference on advances in neural information processing systems (NeurIPS'15)* (pp. 2962–2970). Curran Associates.

Gijsbers, P., Bueno, M., Coors, S., LeDell, E., Poirier, S., Thomas, J., Bischl, B., & Vanschoren, J. (2023). AMLB: An AutoML benchmark. *arXiv:2207.12560 [Stat.ML]*.

Gijsbers, P., & Vanschoren, J. (2021). GAMA: A general automated machine learning assistant. *Machine Learning and Knowledge Discovery in Databases. Applied Data Science and Demo Track*, 560–564.

Gijsbers, P., & Vanschoren, J. (2019). GAMA: Genetic automated machine learning assistant. *Journal of Open Source Software*, *4*.

Hutter, F., Kotthoff, L., & Vanschoren, J. (Eds.). (2019). *Automated machine learning: Methods, systems, challenges*. Springer.

Imrie, F., Cebere, B., McKinney, E., & Schaar, M. van der. (2022). AutoPrognosis 2.0: Democratizing diagnostic and prognostic modeling in healthcare with automated machine learning. *arXiv:2210.12090 [Cs.LG]*.

Lindauer, M., Eggensperger, K., Feurer, M., Biedenkapp, A., Deng, D., Benjamins, C., Ruhkopf, T., Sass, R., & Hutter, F. (2022). SMAC3: A versatile bayesian optimization package for Hyperparameter Optimization. *Journal of Machine Learning Research*, *23*(54), 1–9.

Lindauer, M., Eggensperger, K., Feurer, M., Biedenkapp, A., Marben, J., Müller, P., & Hutter, F. (2019). BOAH: A tool suite for multi-fidelity bayesian optimization & analysis of hyperparameters. *arXiv:1908.06756 [Cs.LG]*.

Mohr, F., & Wever, M. (2023). Naive automated machine learning. *Machine Learning*, *112*(4), 1131–1170.

Olson, R., Bartley, N., Urbanowicz, R., & Moore, J. (2016). Evaluation of a tree-based pipeline optimization tool for automating data science. *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, 485–492.

Sass, R., Bergman, E., Biedenkapp, A., Hutter, F., & Lindauer, M. (2022). DeepCAVE: An interactive analysis tool for automated machine learning. *ICML Adaptive Experimental Design and Active Learning in the Real World (ReALML Workshop 2022)*.

Thornton, C., Hutter, F., Hoos, H., & Leyton-Brown, K. (2013). Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms. *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD'13)*, 847–855.

Wang, C., Wu, Q., Weimer, M., & Zhu, E. (2021). FLAML: A fast and lightweight automl library. *Proceedings of Machine Learning and Systems*, *3*, 434–447.

Zimmer, L., Lindauer, M., & Hutter, F. (2021). Auto-Pytorch: Multi-fidelity MetaLearning for efficient and robust AutoDL. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *43*, 3079–3090.