The Journal of Open Source Software

LCPP: Learning Curve Plus Plus

Ozgur Taylan Turan [1¶] **and David M. J. Tax** [1]

1 Delft University of Technology, The Netherlands ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Johan Larsson

Reviewers:

- @rcurtin
- @zoq
- @robcaulk

Submitted: 30 September 2025
Published: unpublished

Summary

A learning algorithm is said to learn if its performance on a given task improves with experience (Mitchell, 2013). This fundamental definition links the size of training data to the generalization performance of the model. In supervised learning, a **learning curve** depicts how generalization performance evolves as a function of the training set size. Collections of such data, known as **learning curve databases**, track the performance of diverse machine learning algorithms (learners) across multiple tasks as they observe increasing amounts of training data.

Learning curve databases are valuable for **model selection** and for **estimating the amount of data needed** to achieve a target performance. These applications typically assume learning curves are monotonic and convex. However, findings of (Yan et al., 2025), (Mohr et al., 2023) and (Viering & Loog, 2022) suggest that learning curves often exhibit more complex and irregular behavior. Sparse sampling of training sizes limits the ability to fully characterize these behaviors, highlighting the need for **high-fidelity learning curves** to investigate them.

LCPP (Learning Curve Plus Plus) is a C++ library that allows for learning curve creation of machine learning models. `LCPP` enables its users to obtain learning curves for variety of learners on any supervised learning dataset with/out hyper-parameter tuning, enabling model selection and training data requirement determination.

Statement of Need

Generally, creating learning curves is computationally expensive because it requires repeatedly training algorithms on many subsets of varying training sizes. Consequently, learning curves are often computed for a limited number of training set sizes. For example, while creating learning curve databases (Mohr et al., 2023) and (Yan et al., 2025) limited number of training set-sizes are investigated, moreover, these generations are done only for fixed learners without hyper-parameter tuning.

To empower the machine learning community to generate richer, more detailed learning curves, we propose `LCPP`, a C++ library for scalable learning curve generation. `LCPP` offers several features; first several approaches for splitting a given dataset into training and test sets of varying sizes (where training sets can be drawn randomly or incrementally, where test sets can be fixed or vary in size). Next, unlike most existing tools that fix hyper-parameters during learning curve creation, `LCPP` integrates hyperparameter optimization routines from `mlpack` (Curtin et al., 2023), enabling optimized learner evaluations.

`LCPP` also includes a simple dataset container for access to `OpenML` datasets (Vanschoren et al., 2013), with built-in support for complete dataset transformations and train/test splits, allowing users to directly measure the generalization performance of models available in `mlpack` (Curtin et al., 2023) and some other learning algorithms included in itself, such as kernel ridge regression, discriminant classifiers, multi-class classification extensions of binary classifiers.

State of the Field

Several tools are available in the Python ecosystem for learning curve generation. `scikit-learn` (Pedregosa et al., 2011) provides a flexible interface for constructing learning curves, allowing cross-validation strategies to be combined with a learner. However, its extensibility and suitability for constructing high-fidelity learning curves remain limited. `LCDB 1.0` and `1.1` (Yan et al., 2025), (Mohr et al., 2023) primarily serve as wrappers around existing learning curve databases. While learning curve generation is possible, it requires additional modification of the provided scripts and is not a central design focus.

In the C++ domain, we are not aware of any tool explicitly designed for learning curve generation. `LCPP` addresses this gap. It is modular by design, can be extended to support a wide range of learning curve research workflows, and can be deployed in high-performance computing environments with minimal overhead. In addition, similar to `mlpack` (Curtin et al., 2023), it can be valuable for embedded and low-resource environments, for model-selection and hyper-parameter tuning purposes.

Software Design

`LCPP` is designed for easy deployment on high-performance computing (HPC) environments. With little effort it can efficiently run large-scale experiments in parallel, ensuring reproducibility and scalability. Moreover, it supports easy and light-weight check-pointing, allowing high-fidelity (both in terms of the training set size resolution and also the times the training set is resampled) learning curves to be created in multiple sessions. This structure also enables the missing experiments to be investigated easily.

It is also designed with future-proofing in mind. Adoption of the `mlpack` (Curtin et al., 2023) conventions `LCPP` has access to wide range of learning algorithm access. As `mlpack` (Curtin et al., 2023) continues to expand the number of supported models will also increases. In addition, `LCPP` is not restricted by this, any model that is using the same conventions as `mlpack` (Curtin et al., 2023) and relies on `armadillo` (Sanderson & Curtin, 2016) and `ensmallen` (Curtin et al., 2021) can also be used without an effort.

Research Impact Statement

`LCPP` is used in (Turan et al., 2025) and (Turan et al., 2026) to generate large-scale learning curve databases by considering many degrees of freedom involved in this process. By enabling tracking of the generalization performance across machine learning models, `LCPP` facilitates a systematic learning curve creation with a fast development and deployment cycle. We hope that it serves as a foundation for future learning curve research.

AI Usage Disclosure

We did not use AI for the development of this software and the writing.

Acknowledgements

We thank Marco Loog, and Tom Viering for discussions and collaborations that helped shape the broader context of this work, and Gijs van Tulder for his help with documentation and rebasing the repository.

References

Curtin, R. R., Edel, M., Prabhu, R. G., Basak, S., Lou, Z., & Sanderson, C. (2021). The ensmallen library for flexible numerical optimization. *Journal of Machine Learning Research*, *22*(166), 1–6. http://jmlr.org/papers/v22/20-416.html

Curtin, R. R., Edel, M., Shrit, O., Agrawal, S., Basak, S., Balamuta, J. J., Birmingham, R., Dutt, K., Eddelbuettel, D., Garg, R., Jaiswal, S., Kaushik, A., Kim, S., Mukherjee, A., Sai, N. G., Sharma, N., Parihar, Y. S., Swain, R., & Sanderson, C. (2023). Mlpack 4: A fast, header-only C++ machine learning library. *Journal of Open Source Software*, *8*(82), 5026. https://doi.org/10.21105/joss.05026

Mitchell, T. M. (2013). *Machine learning* (Nachdr.). McGraw-Hill. ISBN: 978-0-07-115467-3

Mohr, F., Viering, T. J., Loog, M., & van Rijn, J. N. (2023). LCDB 1.0: An extensive learning curves database for classification tasks. In M.-R. Amini, S. Canu, A. Fischer, T. Guns, P. Kralj Novak, & G. Tsoumakas (Eds.), *Machine learning and knowledge discovery in databases* (pp. 3–19). Springer Nature Switzerland. https://doi.org/10.1007/978-3-031-26419-1_1

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Sanderson, C., & Curtin, R. (2016). Armadillo: A template-based C++ library for linear algebra. *The Journal of Open Source Software*, *1*(2), 26. https://doi.org/10.21105/joss.00026

Turan, O. T., Loog, M., & Tax, D. M. J. (2026). Generalization performance distributions along learning curves. *Pattern Recognition Letters*. https://doi.org/10.1016/j.patrec.2026.01.003

Turan, O. T., Tax, D. M. J., Viering, T. J., & Loog, M. (2025). Learning Learning Curves. *Pattern Analysis and Applications*, *28*(1), 15. https://doi.org/10.1007/s10044-024-01394-6

Vanschoren, J., Rijn, J. N. van, Bischl, B., & Torgo, L. (2013). OpenML: Networked science in machine learning. *SIGKDD Explorations*, *15*(2), 49–60. https://doi.org/10.1145/2641190.2641198

Viering, T., & Loog, M. (2022). *The shape of learning curves: A review*. https://arxiv.org/abs/2103.10948

Yan, C., Mohr, F., & Viering, T. (2025). *LCDB 1.1: A database illustrating learning curves are more ill-behaved than previously thought*. https://arxiv.org/abs/2505.15657