Machine Learning Symbolic Integration Algorithm Selection

Matthew England

Coventry University, Coventry, United Kingdom
Matthew.England@coventry.ac.uk

This talk will present joint work with Rashid Barket, Jüergen Gerhard, and Uzma Shafiq, on the topic of Machine Learning and Symbolic Integration. The work is either already published, or submitted for review, as referenced below.

Introduction

We will summarise a recently concluded PhD project at Coventry University that was sponsored by the software company Maplesoft. The overall task was to provide guidance to the user-level `int` command for symbolic integration in the Maple computer algebra system. That command is actually a meta-algorithm which can call from a range of sub-algorithms. At the start of the project Maple would try these sub-algorithms in turn until one succeeded (in some cases checking guard code that could quickly conclude if an algorithm would fail). The hypothesis of the project was that Machine Learning (ML) could better guide the choice of sub-algorithm for a given instance. Following the priorities of Maplesoft, we optimised for the size of the output rather than the time taken to produce it. We summarise some of the key lessons learned.

The importance & challenge of datasets for ML training in symbolic computation

We started using the data generation methods of [7], but quickly found these to contain hidden biases and shortcomings: much effort was spent reverse engineering established symbolic computation methods to form new data generators, as summarised in [2, 3].

The importance of independent datasets for validating the quality of ML models

It is well-established ML practice to have different datasets for testing and training; but in the case where data is being generated synthetically this is not sufficient – to build trust in the outputs we must validate them on data other than from those generators. We used an independent test set from Maplesoft to uncover in [1] and then fix in [4] shortcomings in our models that would otherwise have not been detected.

Appropriate embeddings for mathematical expressions

Our initial work in [5] used sequential transformer models to process mathematical expressions, similar to [7]. While these showed strong performance on the synthetic data their generalisation to independent data was weaker than hoped for. We discovered that using tree embeddings offered superior results and generalisation for both LSTM [1] and transformer models [4].

Experimenting to find the optimal architecture

We experiments in framing the problem using a range of paradigms: e.g. classification on whether a method is optimal; regression to predict output size. Such traditional paradigms

were challenged by the imputation problem: how to treat data instances were a particular sub-algorithm failed (which is a common occurence in the dataset).

We discovered that the optimal architecture was a two stage approach: we first trained classifiers on the simpler problem of predicting whether a given method could tackle a problem (effectively replacing the guard code for the methods which had it in Maple, and providing guard code for those which did not [5]); and then trained ranking models to order the methods on output size, but enforcing the ranking of methods predicted admissible above those not [4].

Summary

Taken together, these lessons led us to propose in [4] a two-stage architecture tree-transformer solution as the state-of-the-art for the problem. We hypothesise that many of these lessons above may be useful for similar problems of ML optimisation in symbolic computation.

We finish by noting recent work on an adjacent but different problem: the formation of a sequence of integration rules to evaluate an integral similar to how this would be tackled by hand. The problem was previously approached with supervised learning in [8]. In contrast to the work surveyed here, this problem involves iterative decisions on rules and so we proposes the first reinforcement learning approach in [6]. It remains to be seen whether the results in [6] can be improved by utilising some of the lessons surveyed here.

References

[1] Barket, R., England, M., Gerhard, J.: Symbolic integration algorithm selection with machine learning: LSTMs vs tree LSTMs. In: Buzzard, K., Dickenstein, A., Eick, B., Leykin, A., Ren, Y. (eds.) Mathematical Software – ICMS 2024. Lecture Notes in Computer Science, vol. 14749, pp. 167–175. Springer Nature Switzerland (2024), https://doi.org/10.1007/978-3-031-64529-7_18

[2] Barket, R., England, M., Gerhard, J.: Generating elementary integrable expressions. In: Boulier, F., England, M., Kotsireas, I., Sadykov, T.M., Vorozhtsov, E.V. (eds.) Computer Algebra in Scientific Computing, Lecture Notes in Computer Science, vol. 14139, pp. 21–38. Springer Nature Switzerland (2023), https://doi.org/10.1007/978-3-031-41724-5_2

[3] Barket, R., England, M., Gerhard, J.: The Liouville generator for producing integrable expressions. In: Boulier, F., Mou, C., Sadykov, T.M., Vorozhtsov, E.V. (eds.) Computer Algebra in Scientific Computing (Proc. CASC 2024). Lecture Notes in Computer Science, vol. 14938, pp. 47–62. Springer Nature Switzerland (2024), https://doi.org/10.1007/978-3-031-69070-9_4

[4] Barket, R., England, M., Gerhard, J.: Tree-based deep learning for ranking symbolic integration algorithms. Submitted, Preprint: arXiv:2508.06383 (2025), https://doi.org/10.48550/arXiv.2508.06383

[5] Barket, R., Shafiq, U., England, M., Gerhard, J.: Transformers to predict the applicability of symbolic integration routines. In: The 4th Workshop on Mathematical Reasoning and AI (MATH-AI) at NeurIPS'24 (2024), https://openreview.net/forum?id=b2Ni828As7

[6] John, R., Barket, R., England, M.: Replacing heuristic rule ordering in symbolic integration with learned policies. Submitted (2026)

[7] Lample, G., Charton, D.: Deep learning for symbolic mathematics. In: Mohamed, S., White, M., Cho, K., Song, D. (eds.) Eighth International Conference on Learning Representations (ICLR 2020) (2020), https://iclr.cc/virtual_2020/poster_S1eZYeHFDS.html

[8] Sharma, V., Nagpal, A., Balin, M.: SIRD: Symbolic integration rules dataset. In: Proceedings of the 3rd Workshop on Mathematical Reasoning and AI (MATH-AI 2023) at NIPS 2023 (2023), https://mathai2023.github.io/papers/39.pdf