PAIS 2022
A. Passerini and T. Schiex (Eds.)

doi:10.3233/FAIA220072

Constrained Hardware Dimensioning for AI Algorithms

Allegra De Filippo and Andrea Borghesi
Department of Computer Science and Engineering, University of Bologna

Abstract. Given the diffusion of Artificial Intelligence (AI) in numerous domains, experts and practitioners are faced with the challenge of finding the optimal hardware (HW) resources and configuration (*hardware dimensioning*) under different constraints and objectives (e.g., budget, time, solution quality). To tackle this challenge, we propose an automated tool for HArdware Dimensioning of (AI) Algorithms (HADA), an approach relying on the integration of Machine Learning (ML) models together into an optimization problem, where experts domain knowledge can be injected as well. The ML models encapsulate the data-driven knowledge about the relationships between HW requirements and AI algorithm performances. We show how HADA can be employed to find the best HW configuration that respects user-defined constraints in three different domains.

Keywords. Empirical Model Learning, Optimization, Machine Learning, Hardware Dimensioning

1. Introduction & Approach Description

In recent years, Artificial Intelligence (AI) has become widespread in a wide array of diverse domains. While the adoption of AI techniques is on the rise, a big challenge still has no simple answer: determining the right hardware (HW) architecture and configuration (e.g., HW on premises or cloud resources) – also referred to *hardware dimensioning*. This issue is significantly exacerbated by the difficulty of anticipating the behaviour of an AI algorithm on different HW architectures, and by potential constraints both on the available budget and on the quality of the solution. An automated way to match algorithms, user constraints and HW resources would be welcomed for AI practitioners.

We address this issue by proposing HADA[1], an automated approach for HW dimensioning based on the *Empirical Model Learning* (EML) framework [2], where ML models are embedded within an optimization problem to enable decision making over complex real-world systems. The idea is to *learn* the relationships between the AI algorithm performances and HW resources via ML models. In this way, complex aspects of the problem can be approximated with surrogate, data-driven models, rather than being explicitly or analytically expressed.

As shown in Fig. 1, HADA can be represented as a black-box that receives as input a set of features describing an AI algorithm and some user-defined constraints (e.g. budget limits, time constraints and required solution quality), and it produces as output the optimal HW resource dimensioning needed to run the algorithm. HADA surpasses standard ML as it enables *bidirectional interaction* between an AI algorithm, its performance, and

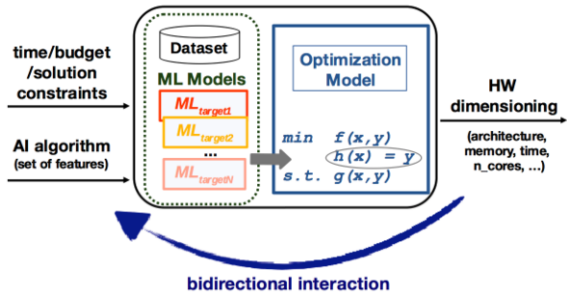

Figure 1. Scheme of HADA. x are the decision variables; y are the observed variables; $f(x,y)$ is the objective function; $g(x,y)$ is the set of user-defined and domain knowledge constraints; $h(x)$ is the approximation of the complex behaviour (the encoding of ML models)

HW dimensioning: given a specific HW architecture, it can be used to obtain the most suitable algorithm for a target task and its optimal parameters. Another key advantage of HADA is that, while an initial benchmarking phase is needed to build the data set, once the ML models are trained, *they can be reused* in the optimization phase on different data instances and different user-defined constraints. This represents a great advantage w.r.t. traditional black-box optimization methods such as surrogate-based methods.

2. Use-cases

The key assumption underlying HADA is that the target algorithms should be treatable as black-boxes exposing a set of configurable parameters, regardless of the application domain. With the aim of demonstrating the potential generalization of HADA, we are testing the approach on dramatically different real-world domains.

2.1. Energy Domain

In this application domain, we focus on online optimization algorithms for energy system domains, which have to take into account tight real-time constraints and uncertainty, e.g., renewable energy production and demand fluctuation. In more details, we consider two stochastic algorithms for energy systems from the literature [3]. Both are characterized by a single changeable parameter that impacts the algorithm runtime, memory consumption and solution quality, and both can be run with different level of parallelism.

In this use case, we focused on a single HW architecture since we mainly wanted to 1) demonstrate the mere feasibility of the approach (HW configuration under user-specified constraints with optimal selected algorithm) and 2) carefully validate it, that is to analyse the impact of ML model uncertainty and studying coping mechanisms. Moreover, as we want to highlight the interplay of data-driven and domain knowledge, we model the impact of the number of threads via an analytical model obtained from the literature for testing the algorithm under different degrees of parallelism.

2.2. Transprecision Computing

A second application domain is represented by transprecision computing [4]: a paradigm that allows users to trade the energy associated with computation in exchange for a

reduction in the quality of the computation results. In this complex domain, a typical target are Floating-point (FP) operations: transprecision techniques allow to specify the number of bits used to represent FP variables, and using a smaller number of bits decreases the precision, thus saving energy. To analytically calculate the impact of varying the number of bits on the computation results for programs with more than a couple of instructions is a crucial point. However, this relationship can be learned from data.

We tackle transprecision algorithms with HADA by 1) selecting the right HW platform (choosing among a High Performance Computing (HPC) system, cloud resources, and a consumer-grade laptop) and 2) specifying the number of bits for each FP variable in the transprecision algorithm. The goal is typically to reduce the runtime or the energy consumption (both affected mainly by the HW) while bounding the computation "error", i.e. the difference between the output computed at reduced FP precision and the one at full precision (the error is determined by the number of bits). These relationships are all approximated with ML models. In this domain, HADA can be used for addressing the challenging issue of selecting the optimal HW platform in a multi-HW setting for transprecision algorithms.

2.3. Adversarial Attacks Optimization

The third application domain is focused on adversarial attacks: algorithms capable of fooling otherwise robust neural networks. These algorithms have paved the way to further studies concerning the resilience of such models to arbitrary input perturbations. Rigorously evaluating their effectiveness, however, requires knowing the actual minimal perturbation to fool a specific model. Computing the exact minimal perturbation is an NP-hard problem, therefore failing to provide absolute measures regarding their effectiveness. It is also even harder to identify the best HW architecture to run the adversarial algorithm.

HADA can be a boon as it can provide the optimal HW configuration and attack method, exploiting the data-driven knowledge extracted from previously run benchmarks. In this case, the input instances are images; the quality of the adversarial attack method is measured via the MIPVerify tool [1]. With HADA we can impose bounds and goals on memory consumption and runtime of the adversarial algorithm, and its solution quality. We test it on multiple adversarial injection methods (e.g., [5]) run on different HW architectures (cloud and HPC resources). This domain provides a useful test-case for (1) testing HADA on HW dimensioning on multiple HW architectures and for (2) analyzing the effectiveness of adversarial attacks.

3. Discussion & Future Works

In this work, we propose an approach combining learning and optimization to automatically perform HW dimensioning and configuration for AI algorithms, under an heterogeneous set of constraints. However, there are a few key aspects that need some consideration when adopting HADA. (1) *Can the relationship between HW resources and algorithms performance be modelled via ML models?* This has been already demonstrated in some use cases, using decision trees and neural networks as ML models ([1,4], and it is an ongoing effort for other domains. The current limitations of the EML library must

[1] https://vtjeng.com/MIPVerify.jl/latest/

be taken into account as well: while arbitrarily complex ML models could, in principle, accurately approximate the target function, only a limited selection of ML models of can be encoded via EML. (2) *Is the optimization model capable of answering the desired queries while being flexible and reusable?* This has been satisfyingly answered in [1,4] by demonstrating how the optimization results directly answer different queries such as suggesting the best HW configuration needed to reach user-desired solution quality and finding the optimal algorithm configuration. (3) *Do the solutions actually satisfy the user-defined bounds?* The optimization model provides a solution which satisfies the given constraints according to the ML model, which might produce inaccurate estimates, and it can lead to solutions not respecting the desired constraints. This means that the robustness of the method must be validated. In HADA we adopted three solutions: i) increase robustness of the optimization model via stochastic considerations (e.g., chance constraints); ii) exploit ML models capable of providing prediction and confidence intervals; iii) iterative learning schemes to improve the ML models, if the solution of the optimization model does not respect the actual constraint. The obtained results are promising, since they show the effectiveness of our approach and its flexibility. However, many aspects are still under investigation and represent interesting open questions in this field. A crucial preliminary phase of HADA is the creation of data sets to train the ML models, obtained through the benchmarking of the target AI algorithms on diverse HW resources and under different configurations. In many domains, this benchmarking activity can be very costly and thus we will have to explore possible actions to mitigate this cost. Moreover, ML models performances are measurable: many models can naturally provide probabilistic, rather than deterministic predictions. These properties could be exploited to deal with optimisation under uncertainty. Importantly, in this way part of the complexity is moved in the training step (performed once), as opposed to directly addressing it in the optimization part.

When the data generation function is available (i.e., the target algorithms) we could collect additional data during the solution process, and this process could be focused on sampling regions of the ML input spaces that are promising and feasible. This could potentially allows the usage of smaller ML models without accuracy loss – and encoding of smaller models are easier to optimize. This iterative data collection approach is connected to surrogate-based optimisation (e.g., Bayesian optimization) and active learning, and it may benefit from some results in those areas, for instance by improving the sampling mechanism with the adoption of acquisition function to determining the next, most promising point to be sampled.

References

[1] De Filippo A, Borghesi A, Boscarino A, Milano M. HADA: an Automated Tool for Hardware Dimensioning of AI Applications. Knowledge-Based Systems. 2022.

[2] Lombardi M, Milano M, Bartolini A. Empirical decision model learning. Artificial Intelligence. 2017;244:343-67.

[3] De Filippo A, Lombardi M, Milano M. How to Tame Your Anticipatory Algorithm. In: Proceedings of IJCAI; 2019. p. 1071-7.

[4] Borghesi A, Tagliavini G, et al. Combining learning and optimization for transprecision computing. In: Proceedings of the 17th ACM International Conference on Computing Frontiers; 2020. p. 10-8.

[5] Madry A, Makelov A, Schmidt L, Tsipras D, Vladu A. Towards deep learning models resistant to adversarial attacks. arXiv preprint arXiv:170606083. 2017.