Master Internship Proposal

Sketch-Guided Optimization of Whole-LLM Tensor Equations

Location	Team Inria CAMUS, ICube Laboratory, Strasbourg (Illkirch campus) or Team Inria CASH, LIP Laboratory, Lyon, France
Advisors	Thomas Kœhler, CNRS Researcher Cedric Bastoul, University of Strasbourg Professor Christophe Alias, Inria Researcher

1 Context

The field of Machine Learning (ML) keeps delivering breakthroughs, revolutionizing how our society interacts with computers. Early successes with models like ResNet-50 (2015) for image processing have been eclipsed by a wave of more advanced Large Language Models (LLMs) that keep improving between models like ChatGPT-4 (2023) and ChatGPT-6 (2026).

However, this success comes with significant training and inference cost. According to NVIDIA CEO Jensen Huang, ChatGPT-6 was trained on more than 100,000 GPUs, with plans to scale up to 400,000 GPUs for future training [1]. The energy consumption of 100,000 such GPUs is estimated at 187 megawatts, which is roughly equivalent to 80% of Strasbourg's overall electricity footprint. Improving the efficiency of artificial intelligence computations is therefore critical.

In practice, this requires diverse optimizations at diverse abstraction levels, many of them starting at the level of mathematical tensor equations. Due to the complexity of such optimizations, ML kernels whose performance is critical are currently hand-optimized: a time-consuming and error-prone process. For example, the popular FlashAttention kernel [2] requires leveraging the mathematical properties of softmax computations in order to correctly and efficiently tile its computations. Even modern domain-specific compilers like Halide [3], TVM [4], or TorchInductor [5] cannot currently exploit such advanced mathematical properties.

Rather than aiming to fully automatically optimize tensor equations in elusively smart compilers, we aim to combine the strengths of manual and automatic approaches through *user-guided* (i.e. semi-automatic) techniques. More concretely, we intend to build on top of two prior techniques that enable programmers to optimize their code by providing *sketches*, incomplete programs that focus on the desired structure of the optimized code, without worrying about further details. The first technique is guided equality saturation [6], a semi-automatic term rewriting technique that allows human insight to guide the optimization process at key points. The second technique is sketch-guided polyhedral compilation [7], which extends this principle to optimizing imperative programs. Neither technique currently scales to optimizing whole LLMs, instead focusing on relatively small kernels such as matrix multiplication.

The goal of this internship is to enable sketch-guided optimization of tensor equations that represent the end-to-end computations of an LLM.

2 Objectives

We envision the following steps for the internship:

1. Design a methodology to decompose whole LLM tensor equations into smaller kernels. This will enable optimizing each kernel independently using techniques like sketch-guided polyhedral compilation, while generating the necessary orchestration code.

2. Use this tool to optimize whole LLMs and benchmark the resulting code against reference PyTorch compilers. Ideally, demonstrate that sketch-guidance enables generating faster code thanks to programmer insight, possibly disabling the PyTorch compiler from targeting optimized libraries whose code was optimized by hand.

3. If time allows, explore how to extend the scope of the tool to more advanced optimizations. For example, exploiting mathematical properties to derive the FlashAttention kernel; applying tensor decomposition, quantization or precision tuning optimizations.

3 Required and Acquired Skills

The intern should come with:

- solid programming skills, familiarity with imperative and functional paradigms
- interest in (or experience with) building programming languages and compilers
- interest in (or experience with) mathematical reasoning and LLM architectures
- willingness to learn (or knowledge of) Python and possibly Rust

The intern is expected to acquire:

- experience in basic academic skills: reviewing literature, conducting novel research, collaborating with other researchers, presenting research and communicating ideas
- greater understanding of LLM computations, compilers and program optimization
- the ability to apply theoretical techniques to practical benchmarks

4 Research Environment

The members of the Inria CAMUS and CASH teams are involved in European and international collaborations on scientific projects both with academia and industry. In particular, this work will take place within the broader French PEPR project called CAMELIA, that aims to develop AI hardware acceleration components and their optimised software stack. Within CAMELIA, Thomas Kœhler coordinates the development of new compilation techniques that facilitate prototyping AI code optimisations at all abstraction levels, from tensor expressions down to hardware ISAs.

The intern will be advised by Thomas Kœhler, Cedric Bastoul and Christophe Alias, and is likely to interact with Valéran Maytie, the PhD researcher who is currently developing the sketch-guided polyhedral compilation technique.

We believe that our success as researchers depends not only on discovery, but also on the integrity, wellbeing, and solidarity of those who make discovery possible. Check out Thomas' manifesto called People Make Academia, detailing what early stage researchers should expect from supervisors, what is expected from them, and more broadly the values that we stand for.

5 Contact

If you are interested, send us an e-mail with a short statement of interest, questions you might have, the desired start date and duration of your internship (4-6 months), as well as a 1 page CV and your Master transcripts. If the internship goes well, there is potential to pursue a PhD in similar topics.

References

[1] *Jensen Huang Declares "AGI Has Arrived" — Powered by 100,000+ Nvidia Grace Blackwell GPUs.* URL: https://247wallst.com/investing/2026/09/07/jensen-huang-declares-agi-has-arrived-powered-by-100000-nvidia-grace-blackwell-gpus/ (visited on 09/16/2026).

[2] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. *FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness.* 2022. arXiv: 2205.14135 [cs.LG]. URL: https://arxiv.org/abs/2205.14135.

[3] Jonathan Ragan-Kelley, Andrew Adams, Sylvain Paris, Marc Levoy, Saman P. Amarasinghe, and Frédo Durand. "Decoupling algorithms from schedules for easy optimization of image processing pipelines". In: *ACM Trans. Graph.* 31.4 (2012), 32:1–32:12. URL: https://doi.org/10.1145/2185520.2185528.

[4] Tianqi Chen et al. "TVM: An Automated End-to-End Optimizing Compiler for Deep Learning". In: *13th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2018, Carlsbad, CA, USA, October 8-10, 2018.* Ed. by Andrea C. Arpaci-Dusseau and Geoff Voelker. USENIX Association, 2018, pp. 578–594. URL: https://www.usenix.org/conference/osdi18/presentation/chen.

[5] Peng Wu. "Pytorch 2.0: The journey to bringing compiler technologies to the core of pytorch (keynote)". In: *Proceedings of the 21st ACM/IEEE International Symposium on Code Generation and Optimization.* 2023, pp. 1–1.

[6] Thomas Koehler, Andrés Goens, Siddharth Bhat, Tobias Grosser, Phil Trinder, and Michel Steuwer. "Guided Equality Saturation". In: *Proc. ACM Program. Lang.* 8.POPL (2024), pp. 1727–1758. URL: https://doi.org/10.1145/3632900.

[7] Valeran Maytié, Reuben Carolan, Christophe Alias, Cedric Bastoul, and Thomas Koehler. "Towards Optimising Programs with Sketch-Guided Polyhedral Compilation". In: *IMPACT 2026 - International Workshop on Polyhedral Compilation Techniques.* Cracovie (PL), Poland, Jan. 2026. URL: https://hal.science/hal-05493794.