Extend an ML Model Compiler for the Generation of Dedicated Hardware Accelerators

Description

Emerging workloads such as machine learning have gained significant importance. However, hardware acceleration is necessary to achieve the required performance and efficiency targets for such workloads. Consequently, a variety of hardware architectures for ML acceleration for application domains ranging from servers to IoT have been proposed by both academia and industry. This, however, open up another problem, which is using the newly developed architecture by the software. ML algorithms are a typical large complex set of functions that must be mapped to various hardware blocks to achieve the required performance. Large data size requires the computational problem to be broken down into smaller chunks while preserving functional correctness. Partitioning and hardware mapping tasks are typically handled by the ML compilers that are typically developed for particular hardware or framework such as NVIDIA CUDA Compile for Cuda-based devices, Accelerated Linear Algebra (XLA) for TensorRT. However, with the advent of new hardware accelerators, new compilers extension are required, thus requiring an open-source framework that allows such capabilities. TVM and MLIR are some of the popular alternatives to designing compilers to map software to execute on custom accelerators.

Goals

- Investigate popular ML compilers such as PyTorch, TensorFlow, TVM, MLIR, etc, to find a suitable candidate.
- Use either PyTorch, TensorFlow, or TVM to extract relevant information from the ML models and feed that to the existing HW generation framework.
- Designing a back-end that scales to generate code for a variety of hardware architectures
- Use the proposed design to generate code for already designed architectures such as FiBHA, VSA, and tiled architecture such as the Alice AI accelerator by ImSys [1]

Pre-requisites

For the success in this project the student needs to have basic knowledge of computer architecture. Basic knowledge of Machine-Learning is an added benefit.

[1] https://imsys.ai/wp-content/uploads/2022/01/Imsys-Alice-1000-Product-Brief.pdf

Targi groups

D, E

Selective Relevant References

- T. Chen, T. Moreau, Z. Jiang, L. Zheng, E. Yan, H. Shen, M. Cowan, L. Wang, Y. Hu, L. Ceze, et al., "TVM: An automated End-to-End optimizing compiler for deep learning," in 13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18), pp. 578–594, 2018.
- C. Lattner, M. Amini, U. Bondhugula, A. Cohen, A. Davis, J. Pienaar, R. Riddle, T. Shpeisman, N. Vasilache, and O. Zinenko, "MLIR: Scaling compiler infrastructure for domain specific computation," in 2021 IEEE/ACM International Symposium on Code Generation and Optimization (CGO), pp. 2–14, 2021.
- F. Qararyah, M. W. Azhar, and P. Trancoso, "FiBHA: Fixed Budget Hybrid CNN Accelerator," in IEEE 34th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD 2022), 2022.
- M. V. Maceiras, M. W. Azhar, and P. Trancoso, "VSA: A Hybrid Vector-Systolic Architecture," in 2022 IEEE International Conference on Computer Design (ICCD), 2022

Contact

Pedro Petersen Moura Trancoso
Computer Science and Engineering
ppedro@chalmers.se