Poster Abstract: TensorBind: Unifying On-device Tensor Program Optimization through Foundation Model

Zhihe Zhao[1], Neiwen Ling[1], Kaiwei Liu[1], Nan Guan[2], and Guoliang Xing[1,*]
[1]The Chinese University of Hong Kong, Hong Kong SAR, China
[2]City University of Hong Kong, Hong Kong SAR, China

ABSTRACT

We present TensorBind, a novel approach aimed at unifying different hardware architectures for compilation optimization. Our proposed framework establishes an embedding space to seamlessly bind diverse hardware platforms together. By leveraging this unified representation, TensorBind enables efficient tensor program optimization techniques across a wide range of hardware platforms. We provide experimental results demonstrating the essentiality and adaptability of TensorBind in translating tensor program optimization records across multiple hardware architectures, thus revolutionizing compilation optimization strategies and facilitating the development of high-performance compilation systems over heterogeneous devices.

CCS CONCEPTS

• **Computer systems organization** → *Real-time System*;

KEYWORDS

Efficient DNN Processing, DNN Compiler, Mobile Computing

1 INTRODUCTION

The deployment of high-performance machine learning models has become a significant challenge in various domains like image recognition, natural language processing, and virtual reality [7]. At the same time, there is an increasing demand to deploy intelligent applications on a wide range of mobile and edge devices, catering to diverse users. DNN compilers serve as intermediaries between AI development frameworks (e.g., PyTorch) and the execution hardware, reducing the model deployment cycle and enhancing the performance of deployed models on these devices [5]. However, as model complexity and computational requirements grow, the compilation overhead of DNN compilation also increases. For example, compiling Llamma2 on an NVIDIA Orin may take up to three days [1]. Thus, it is essential to investigate novel optimization techniques to reduce the compilation overhead without compromising the performance of the deployed models [6].

*Corresponding email: glxing@cuhk.edu.hk.

SenSys '23, November 12–17, 2023, Istanbul, Turkiye

ACM ISBN 979-8-4007-0414-7/23/11…$15.00
https://doi.org/10.1145/3625687.3628378

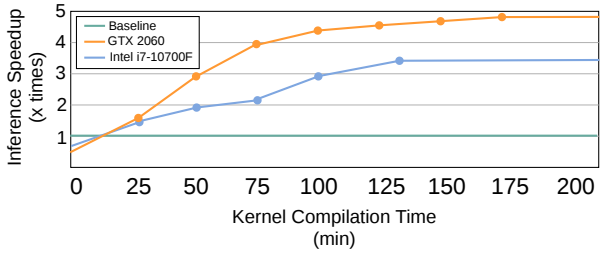

Figure 1: A compiled transformer encoder kernel performance and generation time of TVM Unity [1] on NVIDIA GTX 2060 and Intel i7-10700F CPU. The baseline is ONNX-runtime performance of the model on the two hardware.

We observe a similar phenomenon of cross-device compilation in the emerging field of cross-modality embedding techniques. Recently, there has emerged a framework called ImageBind [3], which is the first framework that utilizes a shared set of parameters to encode data from multiple modalities (e.g., audio and IMU), creating a unified embedding space for all modalities. This enables various compositional multimodal tasks across different modalities, such as evaluating pretrained vision models for non-vision tasks. **We propose TensorBind as a general-purpose solution for unifying cross-device tensor program optimizations.** Our approach leverages foundation models based on transformers to achieve this goal. The transformer-based foundation model serves as a powerful representation learning tool, capturing both hardware architecture and DNN model graph properties jointly.

2 MOTIVATION STUDY

We measured the time required for TVM Unity to generate optimized kernels for a transformer encoder on both a GPU and a CPU device, with a batch size of 2 and a sequence length of 32. As is shown in Fig. 1, it took approximately 2 hours on the GPU and 3 hours on the CPU to identify the kernel with the highest performance. A DNN model typically involves multiple kernels. This incurs significant compilation overhead, especially in the context of IoT devices where client diversity is prevalent. The long optimization time for kernel generation is attributed to the large optimization space and the complex tensor compilation process. Instead of focusing solely on improving tuning efficiency for individual devices, we explore the possibility of transferring tuning records across platforms with different ISA (Instruction set architecture) for the same kernel, thereby addressing the compilation overhead more effectively.

To address this problem, we provide a summary of previous research endeavors aimed at unifying the compilation process, considering two distinct perspectives: the *"top-down"* approach,

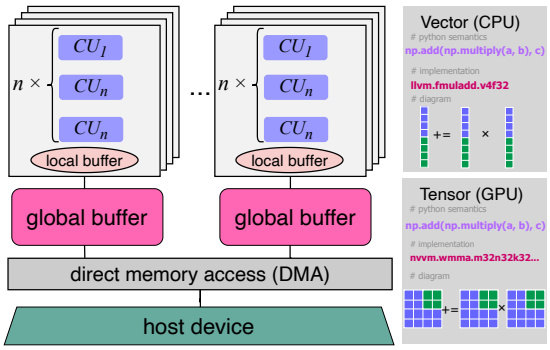

Figure 2: Abstractions for DNN domain-specific accelerators and tensor intermediate representation abstractions for CPU and GPU.

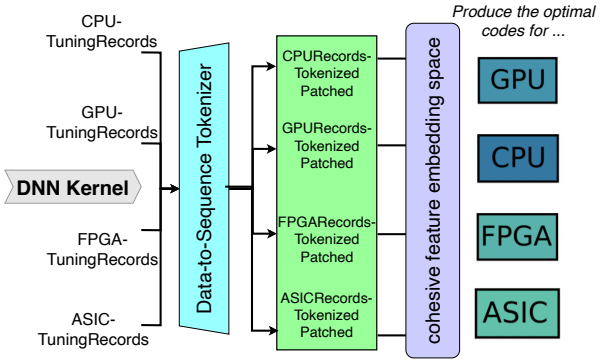

Figure 3: TensorBind comprises sequence-based data tokenization and cohesive feature embedding space. These components are utilized to handle different representations of $< ISA, TuningRecords >$ pairs generated during the compilation process. The framework is demonstrated using various hardware platforms, including CPU, GPU, FPGA, and ASIC.

focusing on the DNN model-to-intermediate representation perspective, such as TensorIR (tensor intermediate representation) and the *"bottom-up"* approach, concentrating on the hardware architecture perspective [2, 4]. In Fig. 2, we present a visual representation of the key insights derived from these works. The efficient scheduling of DNNs based on either of these abstractions is crucial for harnessing the computational capabilities of DNN accelerators in DNN compilers. **However, previous research cannot jointly consider and optimize the compilation process from both perspectives**, primarily due to the absence of representation capabilities to integrate across heterogeneous devices.

3 PROBLEM FORMULATION AND DESIGN

The objective of TensorBind is to develop a foundation model that unifies the DNN compilation process across diverse hardware platforms, which can automatically translate the optimal DNN computational graphs compilation records to target multiple platforms, enabling efficient cross-platform compilation.

Given the DNN computational graph G, the set of hardware platforms P. A unified mapping function F aims to transform the optimized code representations for DNN computational graphs across diverse hardware platforms. With the optimal trace of compilation record T_m on the hardware platform P_m. The function F produces an optimized code representation, denoted as C_n, on the target hardware platform P_n. Mathematically, the formulation is as follows:

$$C_n = F(G, P_n; T_m) \tag{1}$$

TensorBind utilizes a foundation model built upon the transformer architecture, which possesses strong representation capabilities, to create an embedding that facilitates the unification of tensor-program optimization. The transformer architecture is known for its ability to capture and encode complex patterns and relationships within data, making it an ideal choice for generating comprehensive and effective embeddings that can be leveraged in the optimization of tensor programs. The design is shown in Fig. 3. TensorBind aligns the embeddings of compilation tuning records from multiple hardware platforms across different DNN models, ultimately merging them into a cohesive and unified representation embedding space.

4 CONCLUSION AND FUTURE WORK

In conclusion, we propose TensorBind, a foundation model to unify tensor-program optimization and enable cross-device DNN compilation optimization. By embedding DNN graphs and hardware device information into a unified representation space, the model can generate optimal tuning records on one hardware platform and translate them to another with a different ISA. Future work includes extending the model to support hardware platforms with unknown ISAs, such as NPUs.

ACKNOWLEDGEMENT

This work is supported in part by the Research Grants Council (RGC) of Hong Kong under Collaborative Research Fund (CRF) grants C4072-21G and C4034-21G

REFERENCES

[1] Tianqi Chen et al. 2018. TVM: An Automated End-to-End Optimizing Compiler for Deep Learning. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*. USENIX Association, Carlsbad, CA, 578–594.

[2] Siyuan Feng, Bohan Hou, Hongyi Jin, Wuwei Lin, Junru Shao, Ruihang Lai, Zihao Ye, Lianmin Zheng, Cody Hao Yu, Yong Yu, et al. 2023. Tensorir: An abstraction for automatic tensorized program optimization. In *Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*. 804–817.

[3] Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. 2023. Imagebind: One embedding space to bind them all. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 15180–15190.

[4] Jie Zhao, Siyuan Feng, Xiaoqiang Dan, Fei Liu, Chengke Wang, Sheng Yuan, Wenyuan Lv, and Qikai Xie. 2023. Effectively Scheduling Computational Graphs of Deep Neural Networks toward Their {Domain-Specific} Accelerators. In *17th USENIX Symposium on Operating Systems Design and Implementation (OSDI 23)*. 719–737.

[5] Zhihe Zhao, Neiwen Ling, Nan Guan, and Guoliang Xing. 2022. Aaron: Compile-time Kernel Adaptation for Multi-DNN Inference Acceleration on Edge GPU. In *Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems*. 802–803.

[6] Zhihe Zhao, Neiwen Ling, Nan Guan, and Guoliang Xing. 2023. Miriam: Exploiting Elastic Kernels for Real-time Multi-DNN Inference on Edge GPU. *arXiv preprint arXiv:2307.04339* (2023).

[7] Zhihe Zhao, Kai Wang, Neiwen Ling, and Guoliang Xing. 2021. EdgeML: An AutoML Framework for Real-Time Deep Learning on the Edge. In *Proceedings of the International Conference on Internet-of-Things Design and Implementation (IoTDI '21)*. Association for Computing Machinery, New York, NY, USA, 133–144. https://doi.org/10.1145/3450268.3453520