Accelerating Conv2D and DepthwiseConv2D Tensor Operations for TensorFlow Lite on Embedded FPGA with Hybrid Custom Floating-Point Approximation

1st Yarib Nevarez
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

2nd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

3rd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

***Abstract*—Convolutional neural networks (CNNs) have become ubiquitous in the field of image processing, computer vision, and artificial intelligence (AI). Given the high computational demands of CNNs, dedicated hardware accelerators have been implemented to improve compute efficiency in FPGAs and ASICs. However, most commercial general-purpose deep learning processing units (DPUs) struggle with support for low-power, resource-limited devices. In this publication, we present a dedicated hardware accelerator for TensorFlow (TF) Lite on embedded FPGA emulating an Edge TPU coprocessor to delegate Conv2D and DepthwiseConv2D tensor operations. The hardware design is implemented with high-level synthesis (HLS). This accelerator incorporates the support for TF Lite quantization for fixed-point and floating-point. The proposed optimization decomposes floating-point calculation for the dot-product. This approach accelerates computation, reduces energy consumption and resource utilization. To demonstrate the potential of the proposed accelerator, we address a design exploration with custom-built CNNs covering fixed-point quantization, floating-point single precision, half-precision, brain floating-point, TensorFloat, and custom reduced formats for approximate processing, including logarithmic computation. A single accelerator running at 150 MHz on a Xilinx Zynq-7020 achieves 45X runtime acceleration on Conv2D tensor operation compared with ARM Cortex-A9 at 666MHz, and 5X compared with the equivalent standard floating-point implementation in HLS with Xilinx LogiCORE IP. This accelerator yields a peak performance of 1.6 TFLOPS/watt and 152 MFLOP/s. The entire hardware design and the implemented TF Lite software extensions are available as an open-source project.**

***Index Terms*—Artificial intelligence, convolutional neural networks, depthwise separable convolution, hardware accelerator, TensorFlow Lite, embedded systems, FPGA, custom floating-point, logarithmic computation, approximate computing**

I. Introduction

THE constant research and the rapid evolution of artificial neural network (ANN) architectures are driving the transition to smarter and more powerful AI applications, where CNN-based models represent the essential building blocks of deep learning algorithms in computer vision tasks [1]. Applications such as smart surveillance, medical imaging, natural language processing, robotics, and autonomous navigation have been powered by CNN-based models in industry and academia [2]. Nonetheless, dedicated hardware is often a requirement to accelerate execution due to the high computational demands of CNNs. In terms of pure computational throughput, graphics processing units (GPUs) offer the best performance [3]. In terms of power consumption, FPGA solutions are well known to be more energy efficient (vs GPUs). As a result, numerous FPGA accelerators have been proposed, targeting both high performance computing (HPC) for data-centers and embedded systems applications [4]. However, most commercial deep learning processing units (DPUs) are not designed for low-power, resource-limited embedded FPGAs.

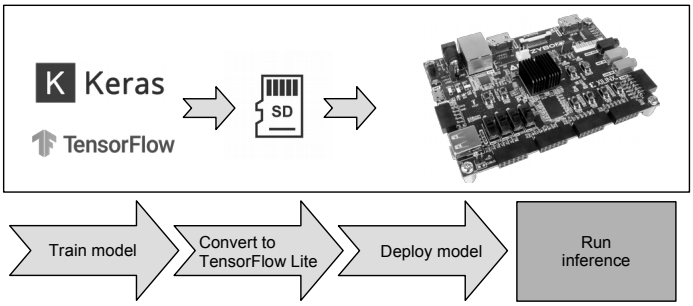

Fig. 1. Deployment workflow.

In this paper, we present a hardware/software co-design framework to accelerate tensor operators for TF Lite Micro on resource-constrained and low-power embedded FPGAs. Based on their high computational cost, in this paper, we accelerate Conv2D and DepthwiseConv2D operators. The proposed architecture supports TF Lite quantized models with int8 and standard float32. To enhance resource utilization and energy consumption, the tensor operators are designed as separate hardware engines, where they are optionally instantiated in the FPGA fabric. To accelerate floating-point computation, we propose a decomposed floating-point calculation for the vector dot-product. To enhance on-chip memory usage and for quality reconfigurability, the proposed design employs hybrid custom floating-point formats, where filter and bias tensors are quantized and stored with a reduced format on the hardware accelerator. The hardware design is implemented with high-level synthesis (HLS).

To operate the proposed accelerator framework, we train a CNN model on TensorFlow or Keras, then this is converted into a TensorFlow Lite model with either float32 or int8 quantization, then the model is stored in a micro SD card along with the software and the FPGA configuration bitstream for deployment. See **Fig.** 1.

Our main contributions are as follows:

- We develop a dedicated hardware accelerator for TensorFlow Lite on embedded FPGA emulating an Edge TPU coprocessor to delegate Conv2D and DepthwiseConv2D tensor operations.
- We develop a hardware/software co-design framework ideal for low-power and resource-constrained AI applications. The parameterized design of the proposed hardware accelerator enables exploring various architecture configurations for different target applications.
- We demonstrate the potential of the proposed framework addressing a design exploration with two custom-built CNN architectures trained for CIFAR-10 image classification. We demonstrate the performance of the Conv2D and DepthwiseConv2D operator engines with fixed-pint quantization, floating-point with Xilinx LogiCORE IP, decomposed floating-point with single precision, half-precision, brain floating-point, TensorFloat, and custom reduced formats for approximate processing, including logarithmic computation. We present a detailed runtime schedule and an exhaustive accuracy performance.

To promote the research in this field, our entire work is made available to the public as an open-source project at .

II. Related work

A. TensorFlow models on the Google's Edge TPU

The Edge Tensor Processing Unit (TPU) is an ASIC designed by Google that provides high performance machine learning (ML) inference for TensorFlow Lite models [5]. This implementation uses PCIe and I2C/GPIO to interface with an iMX 8M SoC. The reported throughput and power efficiency are 4 trillion operations per second (TOPS) and 2 TOPS per watt, respectively [6]. The Edge TPU supports 40 tensor operators including Conv2d and DepthwiseConv2d [7].

However, the Edge TPU has disadvantages.

- Power dissipation. The Edge TPU System-on-Module (SoM) requires 2 to 3A at 5V DC power supply [6], which can be unsuitable for very low-power applications.
- Model compatibility. It supports only TensorFlow Lite models that are fully 8-bit quantized and then compiled specifically for the Edge TPU [8]. The 8-bit quantization method requires a representative dataset that can be inaccessible, which limits its usability.

III. Background

IV. System Design

A. Embedded system architecture

The embedded system architecture is implemented on Xilinx Zynq devices. In this paper, we present a use case with Zynq-7020. processing system (PS), we use a single CPU core ARM Cortex-A9 with NEON floating point unit (FPU) running at 666MHz. The proposed hardware accelerator is the CPU though AXI-Lite interface, and connected with a direct memory access (DMA) t the high-performance to an off-chip memory RAM with 1 GB. communication

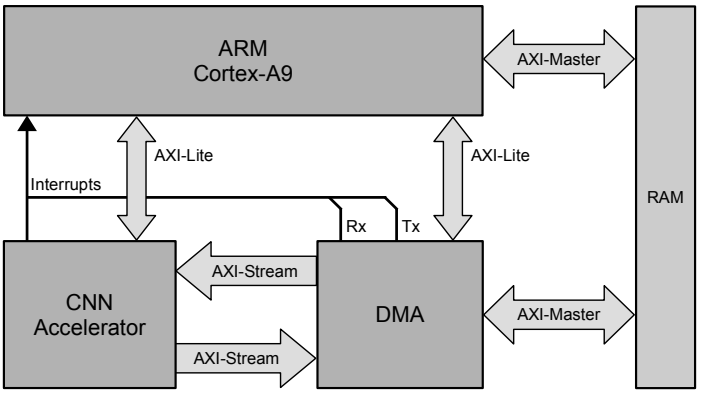

Fig. 2. System-level architecture of the proposed embedded platform.

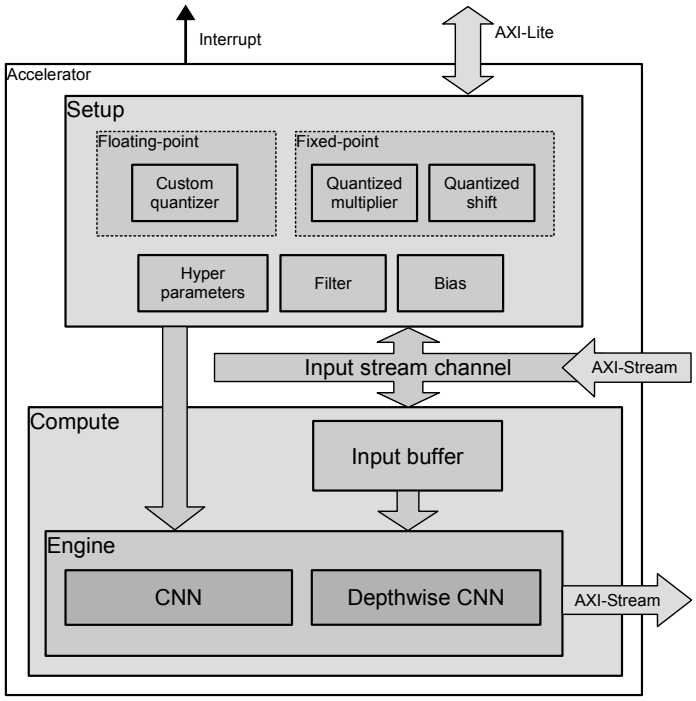

Fig. 3. Hardware architecture of the proposed accelerator.

Regarding the software architecture, this is structured as a layered object-oriented application framework written in the C programming language. This offers a comprehensive high level embedded software application programming interface (API) that allows the construction of scalable sequential SbS networks with configurable hardware acceleration. Conceptually this design is modular, reusable, and extensible. The overall structure is depicted in **Fig.** 5.

In this section, we exploit the hybrid custom floating-point and logarithmic dot-product approximation presented in [9].

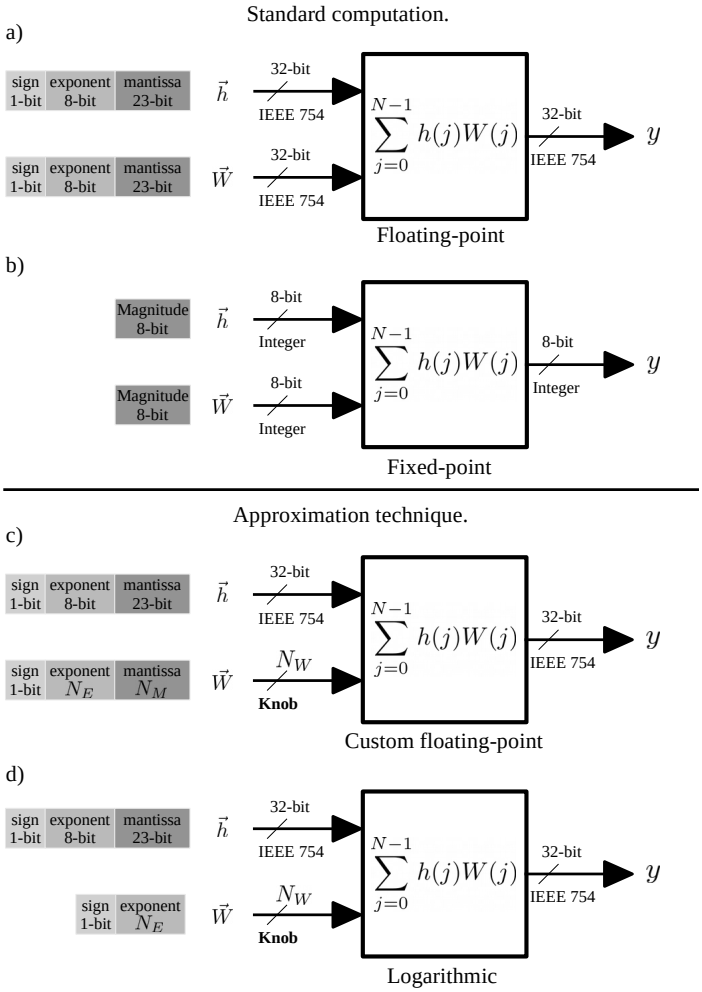

Fig. 4. Proposed hardware modules for vector dot-product.

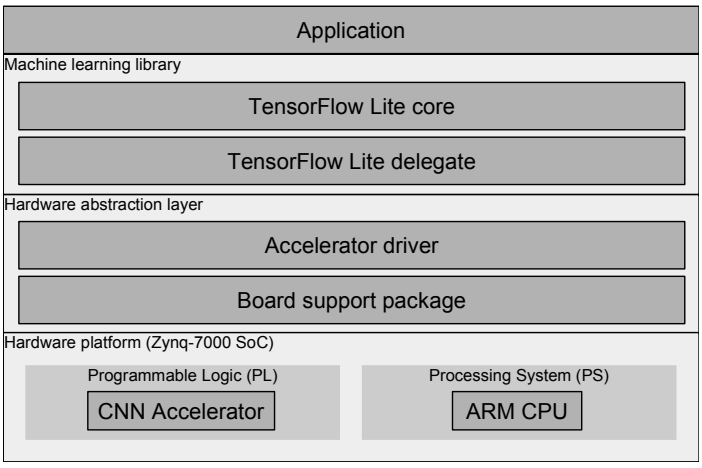

Fig. 5. System-level overview of the embedded software architecture.

V. Experimental results

VI. Conclusions

Acknowledgments

This work is funded by the *Consejo Nacional de Ciencia y Tecnologia – CONACYT* (the Mexican National Council for Science and Technology).

References

[1] M. Hassaballah and A. I. Awad, *Deep learning in computer vision: principles and applications*. CRC Press, 2020.

[2] A. Dhillon and G. K. Verma, "Convolutional neural network: a review of models, methodologies and applications to object detection," *Progress in Artificial Intelligence*, vol. 9, no. 2, pp. 85–112, 2020.

[3] E. Nurvitadhi, G. Venkatesh, J. Sim, D. Marr, R. Huang, J. Ong Gee Hock, Y. T. Liew, K. Srivatsan, D. Moss, S. Subhaschandra *et al.*, "Can fpgas beat gpus in accelerating next-generation deep neural networks?" in *Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, 2017, pp. 5–14.

[4] K. Abdelouahab, M. Pelcat, J. Serot, and F. Berry, "Accelerating cnn inference on fpgas: A survey," *arXiv preprint arXiv:1806.01683*, 2018.

[5] A. Yazdanbakhsh, K. Seshadri, B. Akin, J. Laudon, and R. Narayanaswami, "An evaluation of edge tpu accelerators for convolutional neural networks," *arXiv preprint arXiv:2102.10423*, 2021.

[6] "Coral. dev board datasheet.," https://coral.ai/docs/dev-board/datasheet/., accessed September 15, 2021.

[7] "Coral. tensorflow models on the edge tpu.," https://coral.ai/docs/edgetpu/models-intro/., accessed September 15, 2021.

[8] S. Cass, "Taking ai to the edge: Google's tpu now comes in a maker-friendly package," *IEEE Spectrum*, vol. 56, no. 5, pp. 16–17, 2019.

[9] Y. Nevarez, D. Rotermund, K. R. Pawelzik, and A. Garcia-Ortiz, "Accelerating spike-by-spike neural networks on fpga with hybrid custom floating-point and logarithmic dot-product approximation," *IEEE Access*, 2021.