Tiny On-Chip Memory Realization of Weight Sparseness Split-CNNs on Low-end FPGAs

Akira Jinguji
Tokyo Institute of Technology, Japan
jinguji@reconf.ict.e.titech.ac.jp

Shimpei Sato
Tokyo Institute of Technology, Japan
satos@ict.e.titech.ac.jp

Hiroki Nakahara
Tokyo Institute of Technology, Japan
nakahara@ict.e.titech.ac.jp

I. FPGA on-chip memory and CNN memory usage

Considering implementation in a low-end FPGA with more restrictions on on-chip memory resources and external memory bandwidth, existing methods are limited by communication with external memory. The on-chip memory in a low-end FPGA is small, hence cannot store an entire feature map. It is imperative to rely on external memory for buffering. Although an on-chip memory in an FPGA is fast, implementing sparse CNN in low-end FPGAs is hindered by their limited memory size. To address the limitations of external memory bandwidth and its size, we employ a split-CNN [1] that splits an input image into small spatial patches and tests each patch using a CNN model as shown in Fig. 1. Since the feature-map to be processed at one time is reduced by the splitting, the amount of memory required for buffering is reduced.

II. Overall architecture

Fig. 2 shows the overall architecture. It consists of a convolution unit that performs convolution operation, a controller, and a DMA controller. The convolution unit includes several convolution cores and a write-back module. The number of cores depends on available FPGA resources. Each convolution core consists of an address controller, buffer memory, and 2D SIMD connected in a two-dimensional grid. Each PE computes spatially different outputs at the same time, realizing match parallel computation. for the number of input channels.

III. Experimental Results

We conduct experiments on two tasks: Classification and object detection. In object classification, we experiment with the VGG which learned ImageNet. For comparison, we select the NVIDIA Jetson Nano embedded GPU board, existing FPGA implementation and Google Edge TPU board for ASIC. Our FPGA implementation applied the weight-sparseness techniques, with a ratio of 90% as shown in Table I.

TABLE I
Comparison of CNN inference performance.

Task	ImageNet Classification				VOC Object Detection		
Device	GPU[2]	FPGA[3]	ASIC[4]	Ours	GPU	FPGA[3]	Ours
FPS	10	5.7	2.8	**33.7**	30	8	**80.6**
Clock (MHz)	850	214	-	200	850	150	200
Power (W)	10	3	2	3	10	-	3
FPS/W	1	1.9	1.4	**11.2**	3	-	**26.9**

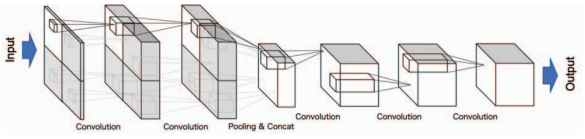

Fig. 1. Split-CNN [1] which splits into 4-patches

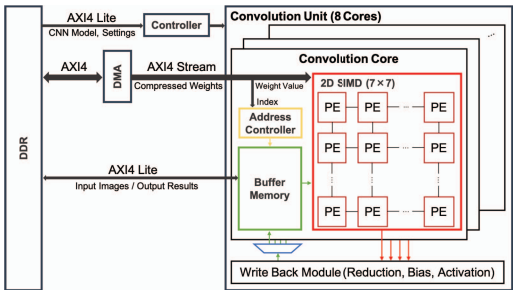

Fig. 2. Overall Architecture

IV. Conclusion

We proposed an architecture and scheduling using weight sparseness split-CNN. Based on the splitting feature maps and scheduling on-chip memory buffering, our implementation resolved the external memory bottleneck on low-end FPGAs and achieved high computational efficiency at high sparseness. Experiments have shown that the proposed architecture is faster than GPUs, ASICs, and existing FPGAs.

References

[1] T. Jin and S. Hong, "Split-cnn: Splitting window-based operations in convolutional neural networks for memory system optimization," in *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*. ACM, 2019, pp. 835–847.

[2] Nvidia, "Jetson Nano: Deep Learning Inference Benchmarks," https://developer.nvidia.com/embedded/jetson-nano-dl-inference-benchmarks/, [Online; accessed 13-January-2020].

[3] K. Guo, L. Sui, J. Qiu, S. Yao, S. Han, Y. Wang, and H. Yang, "From model to fpga: Software-hardware co-design for efficient neural network acceleration," in *2016 IEEE Hot Chips 28 Symposium (HCS)*. IEEE, 2016, pp. 1–27.

[4] Google, "Edge TPU performance benchmarks," https://coral.ai/docs/edgetpu/benchmarks/, [Online; accessed 13-January-2020].

2576-2621/20/$31.00 ©2020 IEEE
DOI 10.1109/FCCM48280.2020.00061

IEEE computer society