2022 IEEE 30th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM) | 978-1-6654-8332-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/FCCM53951.2022.9786198

A Versatile Systolic Array for Transposed and Dilated Convolution on FPGA

Suhail Basalama, Atefeh Sohrabizadeh, Jie Wang, Jason Cong
University of California, Los Angeles, USA
{basalama, atefehsz, jiewang, cong}@cs.ucla.edu

I. Extended Abstract

Many modern CNNs feature complex architecture topologies with different layer types. One of these special layers is a ***fractionally-strided or transposed convolution (T-CONV) layer*** [1], which is an up-sampling layer that uses trained weights to produce enlarged high-resolution feature maps. An ***atrous or dilated convolution (D-CONV) layer*** is another special layer that maintains the resolution and coverage of feature maps by expanding the receptive fields of convolution filters as discussed in [2]. Both T-CONV and D-CONV layers can be naïvely implemented as normal convolution (N-CONV) layers by inserting $S'-1$ zeros between adjacent pixels of the input feature maps (FMs) for T-CONV or $d-1$ zeros between adjacent values of the filters for D-CONV, where S' is T-CONV stride and d is D-CONV dilation rate. This approach, however, leads to a huge underutilization of computation resources due to the introduced zero MAC operations.

Previous FPGA works attempted to accelerate either T-CONV layers like [3]–[7] or D-CONV layers like [8], but not both. On the other hand, some ASIC works proposed versatile accelerators for T-CONV and D-CONV layers like [9]–[12] However, none of these works discussed the area overhead of supporting T-CONV and D-CONV layers efficiently.

We used the decomposition approach in [12], but we provide an optimized integration with the automated systolic array (SA) compiler PolySA [13], and the deep learning implementation framework FlexCNN [14] for T-CONV and D-CONV layers with arbitrary filter size, T-CONV stride (S'), and dilation rate (d).

The decomposition of the T-CONV operation gets rid of the non-effectual zero MAC operations by decomposing the convolution filters into S'^2 sub-filters that convolve over the dense input feature maps producing the same outputs as the naïve implementation.

Symmetrically, the input feature maps of D-CONV are decomposed into d^2 sub-feature maps. Each sub-feature map contains non-contiguous pixels separated by a distance $d-1$ which are then convolved by the dense filters.

To implement the decomposition approach in a systolic array, we used two open-source, Xilinx-HLS-based frameworks: FlexCNN [14] (an end-to-end CNN accelerator), and PolySA (an SA compiler) [13]. To the best of our knowledge, this is the first efficient FPGA implementation of N-CONV, T-CONV, and D-CONV in one systolic array.

For evaluation, we generated two designs: 1) FlexCNN with the versatile SA. 2) FlexCNN with a standard SA. We also used U-Net [15], which has irregular architecture topology and various layer types, including T-CONV layers, to show the effectiveness of our approach in a real-world application.

As shown in Table I, we compared the versatile SA against a standard SA using multiple layers with various filter sizes, T-CONV strides (S'), and dilation rates (d). We found that layers with small $I_c, O_c, or I_{h/w}$ have low computation-to-communication ratios making them communication-bound on both SAs, which explains the low DSP efficiency for these layers. In contrast, the last three layers are computation-bound, and the versatile SA can achieve the ideal speedups by removing the zero MAC operations with a DSP efficiency of around 98%, while the DSP efficiency of the standard SA is capped at $\frac{100}{IdealSpeedup}\%$.

When comparing U-Net performance against CPU and GPU (NVIDIA A100-PCIE-40GB), the versatile SA achieves speedups of $6.9\times$ and $1.3\times$ respectively. Moreover, U-Net on the versatile SA achieves $3.3\times$ speedup for the T-CONV layers, and $1.4\times$ for the whole network compared to a standard SA implementation.

Finally, such a performance improvement comes at a small area overhead compared to a standard SA with about 7% more LUTs, 3% more Flip Flops, and 3% more DSPs. For on-chip buffers, there is a 24% increase in BRAMs to store the enlarged output FMs of T-CONV but a 7% decrease in URAMs due to the removal of zeros from D-CONV filters.

TABLE I: Performance of different T-CONV and D-CONV layers.

Layer $(I_c, O_c, I_{h/w})$	T-CONV Results						D-CONV Results					
	k, S'	Standard SA		Versatile SA		Speedup (Ideal)	k, l	Standard SA		Versatile SA		Speedup (Ideal)
		Latency (ms)	DSP Eff.*	Latency (ms)	DSP Eff.*			Latency (ms)	DSP Eff.*	Latency (ms)	DSP Eff.*	
(256,16,16)	5,2	1.2	18.52%	0.6	37.23%	2.04 × (4×)	5,2	1.1	20.86%	0.7	29.23%	1.42 × (3.24×)
(16,256,16)	5,2	1.1	19.68%	0.6	34.64%	1.79 × (4×)	5,2	1.0	22.04%	0.5	41.77%	1.93 × (3.24×)
(16,16,256)	5,2	14.7	24.25%	5.0	69.50%	2.91 × (4×)	5,2	11.8	30.03%	5.2	67.84%	2.30 × (3.24×)
(256,256,256)	5,2	3653.0	24.94%	913.6	98.14%	4.00 × (4×)	5,2	2958.5	30.79%	913.6	98.14%	3.24 × (3.24×)
(256,256,256)	3,3	2960.7	11.08%	329.5	97.95%	8.98 × (9×)	4,3	3652.3	15.96%	584.9	98.10%	6.24 × (6.25×)
(256,256,256)	4,4	9352.8	6.23%	585.3	98.04%	15.98 × (16×)	3,5	4419.3	7.42%	329.4	98.00%	13.42 × (13.44×)

* DSP efficiency is measured as the actual performance using non-zero MAC operations divided by the peak performance (GFLOP/s) of the SA.

REFERENCES

[1] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," *arXiv preprint arXiv:1603.07285*, 2016.

[2] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.

[3] S. Liu and W. Luk, "Towards an efficient accelerator for DNN-based remote sensing image segmentation on FPGAs," in *2019 29th International Conference on Field Programmable Logic and Applications (FPL)*. IEEE, 2019, pp. 187–193.

[4] S. Liu, H. Fan, X. Niu, H.-c. Ng, Y. Chu, and W. Luk, "Optimizing CNN-based segmentation with deeply customized convolutional and deconvolutional architectures on FPGA," *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, vol. 11, no. 3, pp. 1–22, 2018.

[5] Y. Yu, T. Zhao, M. Wang, K. Wang, and L. He, "Uni-OPU: An FPGA-based uniform accelerator for convolutional and transposed convolutional networks," *IEEE transactions on very large scale integration (VLSI) systems*, vol. 28, no. 7, pp. 1545–1556, 2020.

[6] X. Di, H.-G. Yang, Y. Jia, Z. Huang, and N. Mao, "Exploring Efficient Acceleration Architecture for Winograd-Transformed Transposed Convolution of GANs on FPGAs," *Electronics*, vol. 9, no. 2, p. 286, 2020.

[7] A. Yazdanbakhsh, M. Brzozowski, B. Khaleghi, S. Ghodrati, K. Samadi, N. S. Kim, and H. Esmaeilzadeh, "Flexigan: An end-to-end solution for FPGA acceleration of generative adversarial networks," in *2018 IEEE 26th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*. IEEE, 2018, pp. 65–72.

[8] N. Zhang, X. Wei, H. Chen, and W. Liu, "FPGA implementation for CNN-based optical remote sensing object detection," *Electronics*, vol. 10, no. 3, p. 282, 2021.

[9] Q. Chen, Y. Huang, R. Sun, W. Song, Z. Lu, Y. Fu, and L. Li, "An efficient accelerator for multiple convolutions from the sparsity perspective," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 6, pp. 1540–1544, 2020.

[10] W. Liu, J. Lin, and Z. Wang, "USCA: A unified systolic convolution array architecture for accelerating sparse neural network," in *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2019, pp. 1–5.

[11] D. Im, D. Han, S. Choi, S. Kang, and H.-J. Yoo, "DT-CNN: Dilated and transposed convolution neural network accelerator for real-time image segmentation on mobile devices," in *2019 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2019, pp. 1–5.

[12] K.-W. Chang and T.-S. Chang, "Efficient accelerator for dilated and transposed convolution with decomposition," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2020, pp. 1–5.

[13] J. Cong and J. Wang, "PolySA: Polyhedral-based systolic array auto-compilation," in *2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. IEEE, 2018, pp. 1–8.

[14] A. Sohrabizadeh, J. Wang, and J. Cong, "End-to-end optimization of deep learning applications," in *Proceedings of the 2020 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays*, 2020, pp. 133–139.

[15] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.