A Depthwise Separable Convolution Hardware Accelerator for ShuffleNetV2

Linshuang Li[1], Dihu Chen[1], and Tao Su[1]

[1]Sun Yat-Sen University

April 27, 2024

Abstract

Convolutional neural networks (CNNs) have been widely applied in the field of computer vision with the development of artificial intelligence. MobileNet and ShuffleNet, among other depthwise separable convolutional neural networks, have gained significant advantages in deploying on resource-constrained embedded devices due to their characteristics such as fewer parameters and higher computational efficiency compared to previous networks. In this paper, we focus on the hardware implementation of ShuffleNetV2. We optimized the network structure. Feature channel numbers, pooling modes, and channel shuffle modes are modified, resulting in a 1.09% increase in accuracy while reducing the parameter count by 0.18M. Additionally, we implement a highly parallel hardware accelerator on the Xillinx xczu9eg FPGA, which supports both standard convolution and depthwise convolution. The power consumption of this accelerator is only 7.3W while achieving an energy efficiency of 13.45 GOPS/W. The running frame rate achieves 675.7 fps.

A Depthwise Separable Convolution Hardware Accelerator for ShuffleNetV2

Linshuang Li, Dihu Chen, Tao Su

School of Electronics and Information Technology, Sun Yat-Sen Univer-sity, Guangzhou, Guangdong, People's Republic of China

Email: stscdh@mail.sysu.edu.cn

Convolutional neural networks (CNNs) have been widely applied in the field of computer vision with the development of artificial intelligence. MobileNet and ShuffleNet, among other depthwise separable convolutional neural networks, have gained significant advantages in deploying on resource-constrained embedded devices due to their characteristics such as fewer parameters and higher computational efficiency compared to previous networks. In this paper, we focus on the hardware implementation of ShuffleNetV2. We optimized the network structure. Feature channel numbers, pooling modes, and channel shuffle modes are modified, resulting in a 1.09% increase in accuracy while reducing the parameter count by 0.18M. Additionally, we implement a highly parallel hardware accelerator on the Xillinx xczu9eg FPGA, which supports both standard convolution and depthwise convolution. The power consumption of this accelerator is only 7.3W while achieving an energy efficiency of 13.45 GOPS/W. The running frame rate achieves 675.7 fps.

Introduction: Nowadays, with the development of artificial intelligence, Convolutional Neural Networks (CNNs) as one of its representative algorithms have received increasing attention. Due to its fast speed and small model size, depthwise separable convolutional neural networks have gained significant advantages deploying on embedded terminals.

However, depthwise separable convolutions decouple traditional convolutions into depthwise convolution (DwC) and pointwise convolution (PwC) [1]. Therefore, conventional CNN accelerators are no longer suitable for performing computations in depthwise separable convolutional neural networks. Based on current

research, the pipelined computing architecture introduced an additional feature bank to prefetch data from off-chip memory [2], the introduction of additional storage units leads to increased hardware resources and data read/write time consumptions. The reconfigurable architecture considered the combination of different computation modes in the network model that supports both PwC and DwC calculations [3]. This structure cannot guarantee that all modules are in an operational state during computation, leading to inefficient utilization of computing resources and resulting in wastage. It does not support complex operations such as grouped convolution and network shuffle operations as well. Although traditional channel shuffle mode separately handled the shortcut branch and concating the results after PwC [4], it still resulted in a significant amount of memory read and write operations, leading to high latency.

Based on the above observations, the main contributions of this paper are as follows. We redesigned the structure of the depthwise separable convolution ShuffleNetV2. The network channels, pooling mode and channel shuffle mode are optimized, resulting in a 12.9% decrease in the parameter count and 1.09% accuracy increase. We also proposed a hardware accelerator that supports both DwC and PwC, allowing them to fully utilize and share the hardware resources of the computing array. Achieving Energy efficiency ratio with minimal FPGA hardware resource utilization, resulting in a frame rate of 675.7fps for image processing.

Design details: The purpose of this paper is to implement the computation of depthwise separable convolution on resource-constrained embedded terminals, enabling fast and efficient image classification. Firstly, the pooling layer after the first convolution layer is removed to ensure that more effective feature information enters the construction block for feature extraction. Secondly, to ensure the utilization of processing elements (PEs) in the convolution computing array, the number of channels after each down sampling unit is changed from 29× to 27×. This operation allows for the full utilization and sharing of resources on the same computational array by both DwC and PwC, without wasting hardware resources. The simplified ShuffleNetV2 structure is shown in Table 1. In addition, to further compress the network, we quantized the weights by converting them from 32-bit floating-point numbers to 8-bit fixed-point numbers, with 3 bits for the integer part and 5 bits for the fractional part.

Table 1. Simplified ShuffleNetV2 network structure

Layer	Output size	Repeat	Stride	Output Channel
Image	224×224			3
Conv1	112×112	1	2	27
Stage2	112×112	1	2	108
	112×112	3	1	108
Stage3	56×56	1	2	216
	56×56	7	1	216
Stage4	28×28	1	2	432
	28×28	3	1	432
Conv5	7×7	1	1	1024
Maxpool	1×1			
FC				100

The core of the accelerator is a highly parallel array that supports both standard convolution and DwC. For standard convolution, we employed channel-level parallelism in the convolution. In one clock cycle, each row of the convolution array reads the convolution window at the same position from l input channels, while each column reads the convolution kernel weights from m output feature maps. The array uses a total of$l \times m$ PEs to perform the element-wise multiplication. The input feature maps and weights are accessed in address order, a parallel approach suitable for natural data storage patterns. Moreover, only a single read operation is required for the same data within the same clock cycle, resulting in reduced bandwidth requirements. After$D_k \times D_k$ (kernel size) cycles, a set of convolution results is obtained. Then, the sliding window moves to the next position to traverse the entire feature map, and the computation continues for the next input

feature map channel. This process repeats until all m output feature maps have completed the convolution, and then the calculation starts for the next group of$l \times m$ channel dimensions of the feature maps.

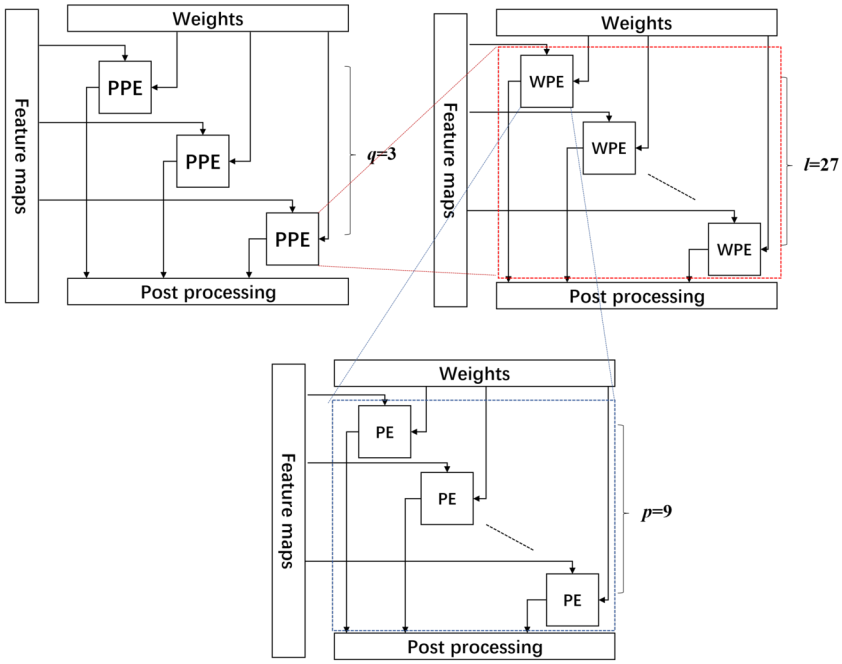

Fig 1 *Convolution computing array for DwC mode.*

For a DwC with both input and output channel parallelism of l , only l groups of PEs can be used simultaneously when implemented on traditional CNN accelerators, while the rest of the PEs in the computation array will be idle. Therefore, to ensure that the computation engine shares the same computation array for both DwC and standard convolution modes, and to achieve high resource utilization efficiency during computation, an additional control module is used to manage the depth convolution mode. As shown in Fig 1, the standard convolution computation array is divided into q image processing units (PPE). Each PPE reads l different channels of input data and the corresponding convolution kernel weights from the buffers. Each PPE also contains l groups of window processing units (WPE) for multiplication. Each WPE performs parallel computation on a single sliding window. Since the depth convolution layer has a uniform 3×3 kernel size, the parallelism of PEs in each WPE is p =9.

In each clock cycle, the computation array first reads $l \times p$convolution weights from the weight buffer for each column. When the input feature map channels remain the same, the same batch of convolution kernel weights can be used for each convolution operation, so the weights can be loaded only once. Then, p pixels of a single convolution window from l input channels of $q$different images are read from the input buffer and loaded to different PEs for convolution. Therefore, the number of parallelizable multiplications is $q \times l \times p$. The convolution window is traversed by prioritizing the entire feature map before moving on to the next set of channel feature map computations, until all $q$different input images have completed the convolution. Besides, the size of the convolution array is determined to be 27×27. The parallelism for both standard convolution and DwC is equal, ensuring that both convolution modes can fully utilize the hardware resources of the computation module without wasting.

After the computation is completed in the convolution array, the feature map data enters the post-processing module for further processing, including the addition tree module, activation module, pooling module, and channel shuffle module. The hardware accelerator architecture is shown in Fig 2. After all the computation

is finished, the output feature map data would be sent to the output buffer and then returned to the BRAM for the next round of computation.

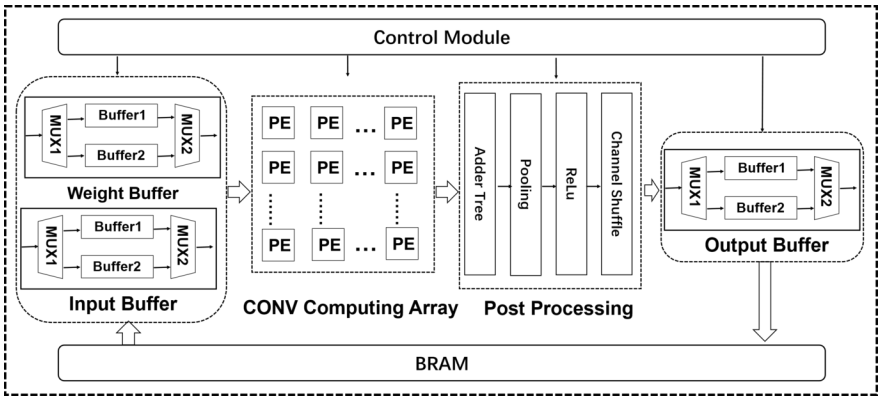

Fig 2 *Architecture of the accelerator.*

For the traditional channel shuffle operation, channels are selected alternately from two groups of feature maps and recombined into a new output feature map, which is then transferred to BRAM as the input for the next convolution. In this paper, the channel shuffle method is modified by partitioning the output feature maps internally into groups of 4 channels. Channels are then selected alternately from the two groups and recombined into a new output feature map, as illustrated in Fig 3. This approach maintained the advantage of increasing inter-channel information exchange while reducing the number of memory read/write operations by 75%, significantly reducing memory access time.

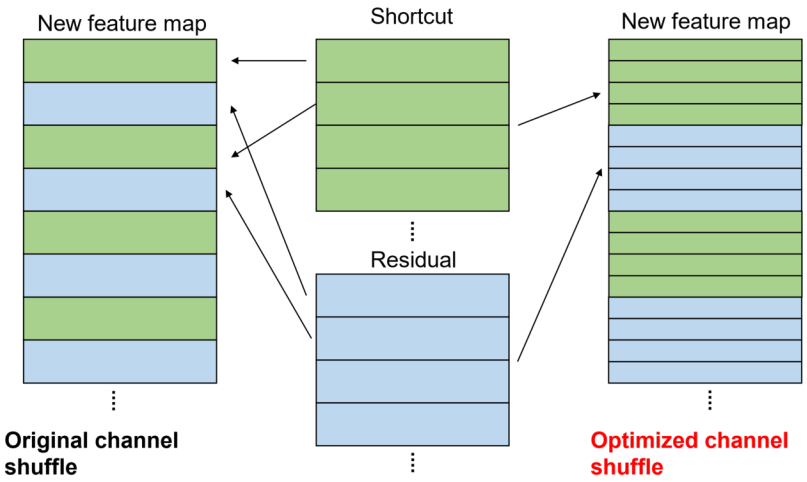

Fig 3 *Comparison between original and optimized channel shuffle mode.*

Experiments and results: The network we proposed is trained and tested using the PyTorch deep learning framework. The test dataset we used is CIFAR100. The simplified ShuffleNetV2 network proposed in this paper is compared with traditional networks, and the comparison results are shown in the table I. In the case of small input feature map size, the network achieves a recognition accuracy improvement of 1.33% and 1.09% compared to the original network. At the same time, the number of parameters is reduced from 1.4M to 1.22M, effectively reducing the storage resources and data access on the hardware.

Table 2. Comparison with original network

	Original network	Simplified network	Simplified network
Data format	float32	float32	int8
Accuracy(%)	69.40	**70.73**	**70.49**
Parameters(M)	1.40	**1.22**	**1.22**

The hardware acceleration system was deployed on Xilinx FPGA chip xczu9eg, with a clock frequency of 180 MHz. The GPU used was NVIDIA GeForce GTX 1080 Ti. The execution times of ShuffleNetV2 on FPGA and GPU are shown in the table II, representing the average time for classifying 1250 images. The speed on FPGA is 1.12 times faster than that on GPU, with a decrease in recognition accuracy of only 0.66%. Since it takes 146M FLOPS to perform the whole network, we can calculate the GPU's energy efficiency as 0.44. Our work has achieved an energy efficiency 30.57 times higher than that of the GPU.

Table 3. Comparison with GPU

Device	FPGA xczu9eg	GPU
Power(W)	**7.3**	220
Power efficiency (GOPS/W)	**13.45**	0.44
Execution time(ms)	**1.48**	1.66
Accuracy(%)	69.83	70.49

We also compared our work with others work [2][4][5][6]. We achieved a computational efficiency that is 1.75 times higher than [4]. Compared to [5], we achieved a throughput and frame rate that are 2.08 times and 11.5 times higher respectively, even at a lower clock frequency. These comparisons demonstrate significant advantages of our work in terms of computational speed and hardware resource consumption.

Table 4. Comparison with other work

	[2]	[4]	[5]	[6]	ours
Device	zu2eg	7z045	zu3eg	7z045	zu9eg
BRAM	145	213	170	64	190
LUT	31198	105000	24130	69666	**18573**
DSP	212	-	-	385	729
Throughput(GOPS)	-	56.1	47.1	33.6	**98.2**
Power (W)	-	–	5.5	-	7.3
Energy efficiency (GOPS/W)	-	-	8.56	-	**13.45**
Frame rate(fps)	205.3	291.5	96.5	240	**675.7**

Conclusion: This paper presents a deep separable convolutional neural network accelerator designed specifically for ShuffleNetV2. Based on the features of ShuffleNetV2, optimizations are made to the network structure, achieving a 1.09% increase in accuracy while reducing the parameters by 0.18M. The paper also proposes a reconfigurable hardware accelerator that supports both PwC and DwC. The power consumption of this accelerator is only 7.3W while achieving a power efficiency of 13.45 GOPS/W. The running frame rate achieves 675.7 fps.

Acknowledgments: The authors thank to the support by the Science and Technology Program of Guangdong Province under Grant 2022B0701180001.

References

1. N. Ma *et al* , "ShuffleNet V2: Practical Guidelines for Efficient CNN Architecture Design," in Proceedings of the European conference on computer vision (ECCV) pp. 116-131, doi: *arXiv:1807.11164v1*.
2. D. Wu *et al* ., "A High-Performance CNN Processor Based on FPGA for MobileNets," 2019 29th International Conference on Field Programmable Logic and Applications (FPL), Barcelona, Spain, 2019, pp. 136-143, doi:*10.1109/FPL.2019.00030*.
3. Y. -G. Chen, H. -Y. Chiang, C. -W. Hsu, T. -H. Hsieh and J. -Y. Jou, "A Reconfigurable Accelerator Design for Quantized Depthwise Separable Convolutions," 2021 18th International SoC Design Conference (ISOCC), Jeju Island, Korea, Republic of, 2021, pp. 290-291, doi:*10.1109/ISOCC53507.2021.9613976*.
4. Z. Fan, W. Hu, H. Guo, F. Liu and D. Xu, "Hardware and Algorithm Co-Optimization for pointwise convolution and channel shuffle in ShuffleNet V2," 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Melbourne, Australia, 2021, pp. 3212-3217, doi:*10.1109/SMC52423.2021.9659057*.
5. Y. Yang *et al* , Synetgy: "Algorithm-hardware co-design for convnet accelerators on embedded fpgas," Proceedings of the 2019 ACM/SIGDA international symposium on field-programmable gate arrays. 2019, pp. 23-32, doi:*10.1145/3289602.3293902*.
6. Y. Lin *et al* , "A High-speed Low-cost CNN Inference Accelerator for Depthwise Separable Convolution," 2020 IEEE International Conference on Integrated Circuits, Technologies and Applications (ICTA), Nanjing, China, 2020, pp. 63-64, doi:*10.1109/ICTA50426.2020.9332057*.