World Journal of Advanced Research and Reviews
eISSN: 2581-9615 CODEN (USA): WJARAI
Cross Ref DOI: 10.30574/wjarr
Journal homepage: https://wjarr.com/

(RESEARCH ARTICLE)

Check for updates

Assessment of Winograd-based processor element for CNN processor and ASIC implementation of convolution processor element

Vaddiraju Venkata Satya Sreya *, Lanka Mohan Satish, Velaga Hemu Sai and Sasanapuri Sai Vijaya Krishna

Electronics and Communication Engineering, GMRIT, Rajam, Andhra Pradesh, India.

World Journal of Advanced Research and Reviews, 2022, 16(01), 761–766

Publication history: Received on 21 September 2022; revised on 25 October 2022; accepted on 27 October 2022

Article DOI: https://doi.org/10.30574/wjarr.2022.16.1.1101

Abstract

The convolutional neural network (CNN) is the most widely used machine learning technique within the fields of image and video processing. It is primarily used to categorize images using vast datasets. This require a lot of calculations. The effectiveness of a field-programmable gate array (FPGA) as a hardware accelerator for CNNs will give excellent performance at low power budgets. The employment of the Winograd algorithm can reduce the number of processing stages in CNN. 2-D convolution is employed for the bulk of calculations in CNNs. The tactic for computing convolution for smaller filter sizes that uses Winograd minimum filtering is the handiest. The comparison of computation complexity for multiplications will be performed using Matlab. The architecture of the Winograd-based processing element, RTL coding in Verilog HDL, and the test bench are designed to examine the performance. The Xilinx Vivado / Cadence tool will be used to implement the processing element in the convolution unit on an FPGA or ASIC.

Keywords: Convolutional Neural Network (CNN); Filter; field-programmable gate array (FPGA); Winograd; Vivado; Matlab

1 Introduction and Literature Survey

Convolutional neural networks (CNNs), the most popular deep learning model, are employed in a variety of fields including speech and picture recognition [1],[2]. CNN is the most extensively used image classification method, and it's employed in a lot of applications. CNN's require a lot of computing and memory [3]. Convolution layers (CONV) and FC layers are two of them that are memory-bound and compute-bound, respectively [9]. Convolution layer computations have been optimized using a variety of techniques and architectures.

Convolutions are reduced to generic element-wise matrix multiplications (GEMMs) in the usual method [7]. Convolution based on the Fast Fourier transform (FFT) is less computationally complex than the traditional technique. Unfortunately, it only works well with big filter sizes or when the size of the filter and the data are the same. In normal CNNs, many of the convolution layers employ minimal filter sizes. With the help of the Winograd algorithm, these layers can be calculated. In contrast to the traditional CNN, the Winograd algorithm is used in this study. It is derived based on Chinese Remainder Theorem [3]. Since using traditional CNN is more expensive and takes several multiplications.

The input values are typically mapped memoryless to a single output value in processing elements. The processing elements (PE) frequently conduct multiplication, accumulation, subtraction, and shift operations [8]. Since all of the processing components operate on the same input, it could be desirable to combine them into a single PE with numerous inputs and outputs. In this study, we applied 1-D and 2-D strides for various kernel sizes for Winograd-based convolution to the input matrices and kernels. These are given to the PE that performs Multiply and accumulate operation (Mac).

* Corresponding author: V.V.S.Sreya
Electronics and Communication Engineering, GMRIT, Rajam, Andhra Pradesh, India.

The 1-D, 2-D and 3-D Winograd Minimal Filtering Algorithms (WMFA) of stride 2 CNNs are the foundation for this research. These algorithms divide the input and kernel into groups based on odd and even positions. Greater effectiveness and benefits are made possible by this concept for CNN designs that make use of stride 2. J. Yepez and S.B. Ko proposed the model and the VGG architecture on various Xilinx boards are contrasted in this article. The boards in question are the Arria-10, MPSOC ZCU102, and Zynq XC7Z045. The accuracies reported by the mentioned boards were 86.66, 84.66, and 91.4. With efficiencies of 0.18, 1.32, and 1.33 (GOPs/DSP), the performances were 137, 3044, and 1788 (GOPs) respectively [3].

L. Du et al. suggested that the hardware IP is used to create a traffic-sign net, which is then tested on the FPGA. Convolution in a neural network is localized by creating a regional filter window in each input channel, as opposed to normal convolution, which requires the entire input data to generate one output data. Its energy efficiency of 434GOPS/W enables it to be integrated with IoT devices. They proposed the use of an accelerator built using TSMC's 65-nm technology and having various core sizes. The sizes of the cores superscripted are 〖5mm〗^2, 〖12mm〗^2a 〖16mm〗^2. The respective power usage is 0.3, 0.5, and 0.05 watts. For the specified cores, the efficiency is 434, 166, and 1400 (GOPs/W). The specified cores' performances can be given as 152, 84, and 64 (GOPs) respectively [5].

X. Wang et al., focused on lessening the encoding overhead brought on by activation sparsity, they suggested an algorithm optimization strategy for the Winograd method, and an encoding format termed MBM in this work. In hardware design, they created a unique Scatter-Compute-Gather approach to deal with the inconsistencies of sparse data. They proposed WINONA, an FPGA accelerator that can support the Winograd algorithm, weight sparsity, and activation sparsity all at once based on the methods. The proposed model and VGG Architecture are contrasted on various types of boards in this article. The boards used are the Zed board, ZC706, and VCU108, which operate at power consumptions of 2.1, 10.1, and 21.6 watts with execution times of 80.9, 12.2, and 8.9 milliseconds, respectively. 5.89, 8.03, and 5.2 frames per joule are attained in terms of efficiency [6].

FPGAs being the most impressive choice in executing the complicated multiplications between abundant input matrices, multiple processing array architectures evolved in which Junzhong Shen et. Al [8] focussed on implementing linear parallel arrays along with the combinational approaches. Dense This also includes the workload queue management (WQM) of the PE array. The dense matrix multiplications are performed easily by sub-block matrix multiplications, storing inputs and the acquired results in the off-chip memory. The process of attainment of sub-block matrices is duly noticed by the Memory Access Controller (MAC). Implementation of MAC, WQM, with off-chip DDR memory access shows that the optimal design gives better performance and improved computational efficiency.

2 Methodology

2.1 Striding in Convolutional Neural Network (CNN)

CNN is the most efficient for classifying images. They produce accurate results compared to other architecture. The CNNs have greater abilities that they are trained to recognize and able to extract the features of the image. In CNN, the term convolution defines the multiplication of two functions that produce the third function. It means the images are represented in matrix form and multiplied to get the output. This output is useful for the extraction of the images.

Stride is a part of the convolution neural network which is used to convolve the kernel with the input matrix. The Fig-1 represents the example for stride-1 and stride-2 convolution for the 5 x 5 input matrix with a 2 x 2 kernel matrix.

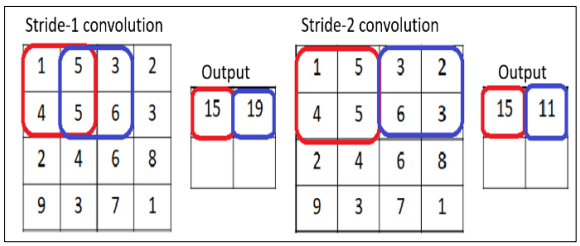

Figure 1 Stride-1 and Stride-2 Convolutions

In Fig-1, The first part represents the convolution with stride-1. In stride-1, the kernel will move one unit at a time. The second part represents the convolution under stride 2 the kernel will move two units at a time. The convolved result under a single stride unit is stored separately as output.

2.2 The Winograd technique

The 1-D convolution with stride 2 is depicted in the fig-2. Here, even input locations are multiplied by even kernel positions, while odd input positions are multiplied by odd kernel positions. Here, we divided the data into two groups, one that is even and the other that is odd. We may do the convolutions with stride 2 using these groups. The example for the input size 5 and filter with 3 is shown in the fig-2.

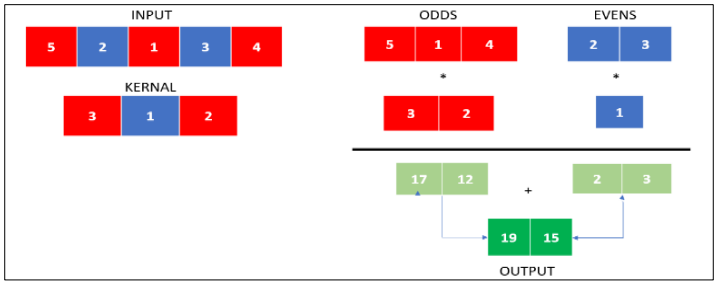

Figure 2 1-D Convolution with stride 2 and kernel size of 3 Example

The odd group contains 3 elements in the input and 2 elements for the kernel. The even groups 2 input components and one filter element. The even input is convolved with the even filter and the odd input is convolved with the odd filter. The outputs of the two groups are added to get the final output. This example uses a total of 3 multiplications. Two multiplications for even and one multiplication for odd groups.

The algorithm uses three additions of involving data, two multiplications, and two additions for the involvement of the kernel. For normal convolution, it uses six multiplications. The same can be applied to a 2-D convolution as shown in Fig 3, here we considered a 3 x 3 kernel with stride 2 and with input data as a 5 x 5 matrix. The input data and the kernel is divided into four different groups. The first group contains row and column elements that are odd for both the input and kernel. The second group contains the elements with an odd row and a column as even for the kernel and input. The third group contains the elements with the row as an even index and the column as an odd index for both kernel and input. The fourth group contains elements as both rows and columns as even.

For the input of 5 x 5 and kernel of 3 x 3, the first group has a total of 9 elements for input and 4 elements for the filter and the second group contains 6 elements for input and 2 elements for the kernel. 6 input components and 2 kernel elements are in the third category. The fourth group contains 4 elements for input and only one element for the filter group one takes 9 multiplications, group two takes six multiplications. Groups three and four take single multiplication. By this method, we get a total of 25 multiplications whereas the traditional requires 36 multiplications.

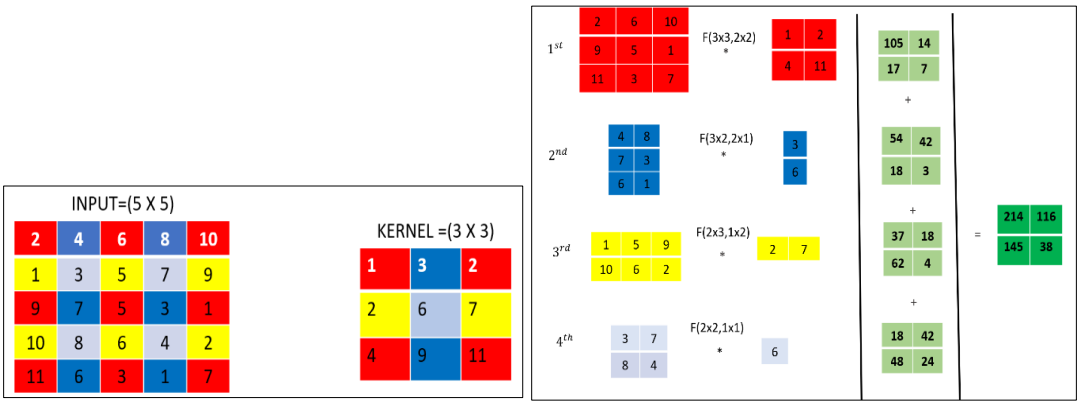

Figure 3 Example of Convolution by kernel 3 x 3 with stride 2

2.3 Validation methodology

The validation of the methodology is done by taking the examples of input and kernel matrices having different sizes. The calculation of the Winograd based convolution and conventional convolution are done using Matlab. The

computation complexities are also compared and mentioned in Table 1. So, there is decrease in number of multiplications. It is clear that utilized architecture gives better performance than the standard CNN architecture. Thus, this provides more accuracy and computational complexity is reduced to the factor 2.25 [9].

2.4 Hardware Implementation

Field-programmable gate array (FPGA) is a hardware accelerator for CNNs that can give a great overall performance at minimal strength budgets, as demonstrated by recent research. 2-D convolution is used in most CNN calculations. For smaller sizes, the most effective method for computing convolution is a set of rules based on Winograd minimum filtering. Completely linked layers that are generated using general element-wise matrix multiplication are also included in CNN's (GEMM).

In this paper, we advocated a paradigm in which an equal number of processing elements might be used to boost both Winograd-based convolution and GEMM [7]. The PE contains a multipliers adder, register, and FIFO'S. The accelerator's computing heart is shaped by PE arrays. One Mac operation can be executed by PEs every cycle. Such PEs can work together as a single systolic processing unit when they are coupled.

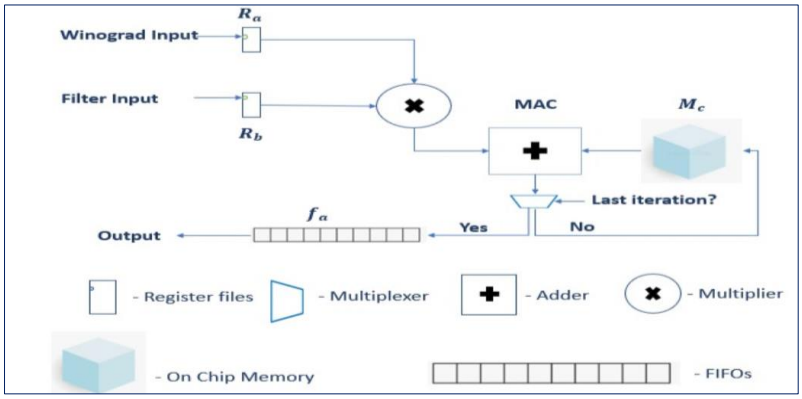

Figure 4 PE Architecture

The operation of PE can be done in two ways: independent mode and connected mode [9]. In the independent mode of functioning, there will be intercommunication between the multiple PE blocks. whereas in fully connected mode the PEs altogether work as a single workload and the complete work is distributed among the PEs without latencies. The architecture of one such PE is shown in Fig-4. In this, the input is given to capping, R_A and filter input is given to the R_B registers.

The Winograd algorithm shown below has already used the inputs. The step involved in the algorithm are:

- Step 1: Separate the given input matrix based on the odd indices and even indices in rows and columns.
- Step 2: Separate the kernel elements based on the odd indices and even indices in rows and columns.
- Step 3: Then group the input and kernel based on the index values separately.
- Step 4: Now perform the stride 2 convolution with the filter on the input element-wise which are grouped.
- Step 5: Repeat step 4 for all the grouped matrices for every element in the matrices.
- Step 6**:** Now add the corresponding elements of the resulted matrices into a single output matrix.

After Performing Winograd-based separation for both the input and kernel. The input is stored in RA and the kernel is stored in RB registers. Now the input and kernel matrix are applied to the Mac unit in the PE architecture. The Mac unit performs the Multiplication and Addition to the given input and kernel.

The Multiplication is performed based on stride 2. A controller is employed after the Mac unit to check the last iteration. If it is the last iteration the resultant value is stored in a queue register. The queue register follows the First In First Out pattern. If it is not the last iteration the value is feedback to the Mac unit. Followed by Fig-3, The inputs and kernel which are grouped are applied to the PE architecture. Let us consider 1st group from Fig-3, Winograd input is of size 3 x3 and kernel size is 2 x 2 is applied to the PE block.

In this block, the input is convolved with the filter by stride 2. The striding is performed element-wise on the input data under the Mac unit. The process is repeated till the last iteration is performed. The resultant is stored in the queue register. The same process is repeated for all the groups in the Fig-3. Finally, all the elements of the respective groups are added to get the output matrix.

3 Results and discussion

We obtained fewer multiplications on each operation of the Winograd-based convolution as compared to traditional convolution after running convolutions based on the Winograd technique on various kernel sizes. Table -1 compares the multiplicities needed for utilized architecture against standard convolution. The implementation of the basic architecture of the processing element as shown in fig-4 is designed and simulated using the Xilinx Vivado tool where ra, and rb are the 4-bit binary inputs and results include output waveforms in fig-5 along with the simulation summary of the PE architecture. Table 2 shows the post-synthesis results describing the availability and usage of Look Up Tables (LUT), Flip flops (FF), Input-Output (IO) and Buffer (BUFG).

Table 1 Comparison of Standard convolution and Utilized Architecture Using Matlab

Algorithms	Standard Convolution (Mul)	Utilized Architecture (Mul)
F (2,2)	6	5
F (2 x 2,3 x 3)	36	25
F (2 x 2,5 x 5)	100	49

Table 2 Post-synthesis results

Resource	Utilization	Available	Utilization %
LUT	34	53200	0.06
FF	25	106400	0.02
IO	414	200	207.00
BUFG	1	32	3.13

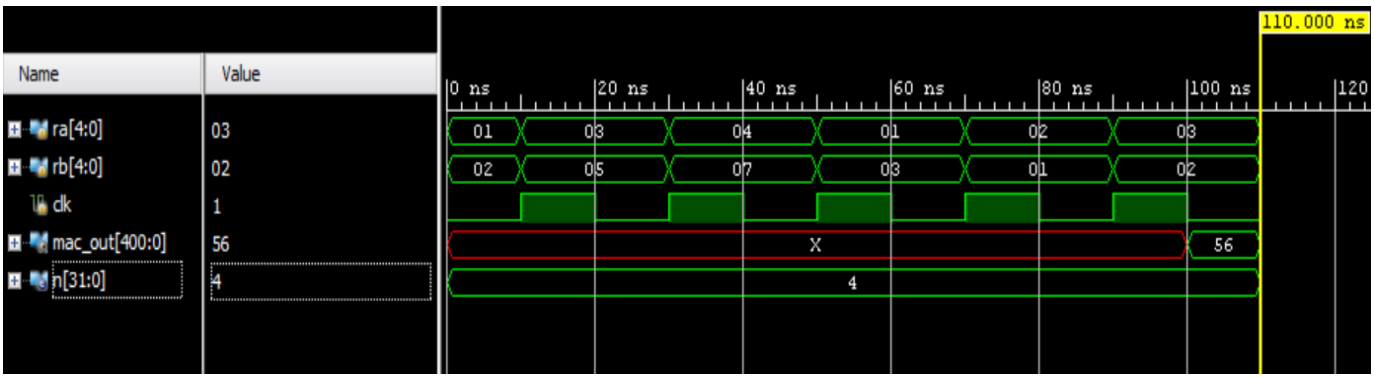

Figure 5 Simulation result of PE architecture

4 Conclusion

This work shows the brief importance of FPGA as a hardware accelerator for CNNs by employing the Winograd algorithm. The basic functionality of the PE architecture is specified by implementing the Verilog HDL and observing the simulation waveforms with the determined inputs. This also reduces the complexity by utilizing smaller filter sizes which are observed through the MATLAB simulations. Hence the employment of the Winograd algorithm can reduce the number of processing elements in CNN with reduced computational complexity.

Compliance with ethical standards

Acknowledgments

This project is done in the VLSI lab of the Department of Electronics and Communication Engineering of GMR Institute of Technology. We are grateful to our project guide Dr. L. Govinda Rao, Associate Professor at GMR Institute of Technology, and to our college management for providing us with the required software tools for this project.

Disclosure of conflict of interest

Hereby, the authors declare no conflicts of interest.

References

[1] H. Qian, J. Xu and J. Zhou, "Object Detection Using Deep Convolutional Neural Networks," 2018 Chinese Automation Congress (CAC), 2018.

[2] X. Xi, Z. Yu, Z. Zhan, Y. Yin and C. Tian, "Multi-Task Cost-Sensitive-Convolutional Neural Network for Car Detection," in IEEE Access, vol. 7, pp. 98061-98068, 2019.

[3] J. Yepez and S. -B. Ko, "Stride 2 1-D, 2-D, and 3-D Winograd for Convolutional Neural Networks," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 28, no. 4, pp. 853-863, April 2020.

[4] L. Lu, J. Xie, R. Huang, J. Zhang, W. Lin, and Y. Liang, "An Efficient Hardware Accelerator for Sparse Convolutional Neural Networks on FPGAs," 2019 IEEE 27th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), 2019.

[5] L. Du et al., "A Reconfigurable Streaming Deep Convolutional Neural Network Accelerator for Internet of Things," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 65, no. 1, pp. 198-208, Jan. 2018.

[6] X. Wang, C. Wang, J. Cao, L. Gong, and X. Zhou, "WinoNN: Optimizing FPGA-Based Convolutional Neural Network Accelerators Using Sparse Winograd Algorithm," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 39, no. 11, pp. 4290-4302, Nov.2020.

[7] S. Kala, J. Mathew, B. R. Jose and S. Nalesh, "UniWiG: Unified Winograd-GEMM Architecture for Accelerating CNN on FPGAs," 2019 32nd International Conference on VLSI Design and 2019 18th International Conference on Embedded Systems (VLSID), 2019.

[8] J. Shen, Y. Qiao, Y. Huang, M. Wen and C. Zhang, "Towards a Multi-array Architecture for Accelerating Large-scale Matrix Multiplication on FPGAs," 2018 IEEE International Symposium on Circuits and Systems (ISCAS), 2018, pp. 1-5.

[9] S. Kala, B. R. Jose, J. Mathew and S. Nalesh, "High-Performance CNN Accelerator on FPGA Using Unified Winograd-GEMM Architecture," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 27, no. 12, pp. 2816-2828, Dec. 2019.