2024 47th International Conference on Telecommunications and Signal Processing (TSP) | 979-8-3503-6559-7/24/$31.00 ©2024 IEEE | DOI: 10.1109/TSP63128.2024.10605959

Hardware Parallel Structure for Convolution Computing in Image Processing

Reyes Perez Melesio*, Moises Arredondo-Velázquez†, Javier Diaz-Carmona*

*Electronics Engineering Department, Technological Institute of Celaya, Celaya, Gto, Mexico

† Faculty of Physical and Mathematical Sciences, Meritorious Autonomous University of Puebla, Puebla, Puebla, Mexico javier.diaz@itcelaya.edu.mx

***Abstract*—A proposed hardware structure for convolution computing with application in image processing is described in this paper. The solution is based on using parallel processing in the computing of the filtering and MAC operation between the convolution mask and one or various filters, as well as on a proposed structured and defragmented input data. According to the described proposal, a flexible feature state of the proposed structure is possible with the combination of different parallelism computing types. Such feature can be extrapolated to different types of hardware. The parallel computing allows reduced computing time mainly in the filtering and convolution computing stages. The obtained structure is suitable to be implemented in convolutional neuronal networks applications.**

***Keywords*—Hardware design, image processing, parallel convolution; MAC; Line buffer**

I. Introduction

Nowadays, the convolution operation is a topic that has taken on great relevance due to the areas where it is used, such as image processing, neural networks, and even artificial intelligence. Due to its key relevance in data processing, several computing methods to optimize and efficiently perform convolution algorithm are being sought.

In recent years, research and proposals have been made in this regard to achieve efficient data processing when running a convolution neural network.

Advances have been made through different proposed approaches particularly in convolution parallel processing. Some research has been focused on algorithm optimization. This includes techniques such as Winograd convolution, Fast Fourier Transform (FFT)-based convolution and depth separable convolution [1]. Another research approach is based on specialized acceleration hardware, where specialized processing units are being designed and improved to accelerate parallel convolution. This includes developments in sensory processing units (TPUs) and neural processing units (NPUs) [2].

Although the convolution algorithm optimization is where some researches are mainly focused on, hardware energy consumption optimization is an important design aspect to take into account mainly in parallel convolution computing. Strategies such as model quantization, neural network compression and low power hardware design are being explored to improve the energy efficiency of hardware systems running parallel convolution operations [3]. Many other researchers focus on general applications, or more specifically using reconfigurable hardware (FPGA) implementations [4,5] for efficiently computing the convolution algorithm, adding and modifying the same algorithm stages in different ways, modifying the max pooling, adding hardware elements, and even optimizing the memory resources of the algorithm and memory elements of the hardware component.

For all the above mentioned, it can be considered that the topic of parallel convolution is a very current topic and that it is worth developing more optimization proposals for the existing wide convolution operation scope and applications. Being of them the convolution computing in image processing, which is the application that the proposed structured is focused on.

II. Data Form

The convolution input and output image data form, as well as the parallelism computing concept are highlighted in this section.

A. Matrix form of an image

Before further explaining the proposed convolution structure, it is important to define how the image input data is considered, or more specifically, how the information of an image or series of images is considered. Currently it is common knowledge that images can have different dimension values in height and width, being the most common multimedia resolution 1920 x 1080, which simply refers to the image size in pixels and can be represented by a matrix with 1920 columns and 1080 rows.

In the image convolution computing process, the convolution operation is done between the input image convolution mask and a filter, which is a small dimension filter that is swept along the input image area. The convolution mask is a square matrix with commonly dimensions of 3x3, 5x5, 7x7 and 9x9. Larger masks could be used but those dimensions are the most used for optimization and parallelism purposes.

The convolution mask matrix swept is done from left to right column by column and up to down row by row. Such

Authorized licensed use limited to: VIT University. Downloaded on September 10,2024 at 16:07:17 UTC from IEEE Xplore. Restrictions apply.

swept path may be modified depending on what is requested to the convolution.

B. *Parallel Processing*

The concept of performing parallel operations is just to perform multiple operations of the same or different nature at the same time. However, the implied concept operations are limited by the available resources in a hardware implementation. In order to achieve efficient parallel processing implementation, efficient hardware solutions are currently used. For instance, optimization techniques in computing processes and logical operations have been used in DSP and FPGA architectures. One of the most known is the pipeline technique, which focuses its parallelism on different computing stages.

One of the key features in the pipeline technique is to free valuable resources for starting a new computing process stage when the current one is finished. An important question in the pipeline processing is: What would happen if the stages parallelism were changed to free hardware resources and parallel perform operations of different magnitudes at the same time?

III. Proposed Structure

The first step in the convolution structure is its data input of one or multiple images. The whole input image pixels are defined as a matrix of a given dimension. In an image processing hardware, the image pixels data are introduced as memory blocks implemented by Flip-Flop sets. The memory blocks are defined through shift registers with the purpose of taking every data package from the input image. Some resources as well as a processing time are required by the entire process, which depends on the clock speed. Three clock cycles for executing a process command or action were considered in the developed analysis.

A. *Line Buffer*

The line buffer has the function of storing all the information of an input image, but only in sectors or fractions. The image data input process in the proposed structure, each data package is defined by storing the data in the flip- flops and shifted through shift registers (SR), once all the input image information is stored the convolution mask is defined (Fig. 1). It should be noted that this convolution mask has the characteristic of being flexible, which means that it can be extended to be used in a convolution layer of any dimension. This flexibility is possible due to the existence of some connections, denoted as dotted lines, that are derivations placed in the inputs of the odd flip-flops and in the shift resisters outputs. In this way the filter mask dimension can be easily modified.

The convolution mask is the image data matrix to which all available filters are applied to. Once the convolution mask has been filtered its values are updated with the next input image data and same filters are applied again. The process is repeated until the complete image information is covered.

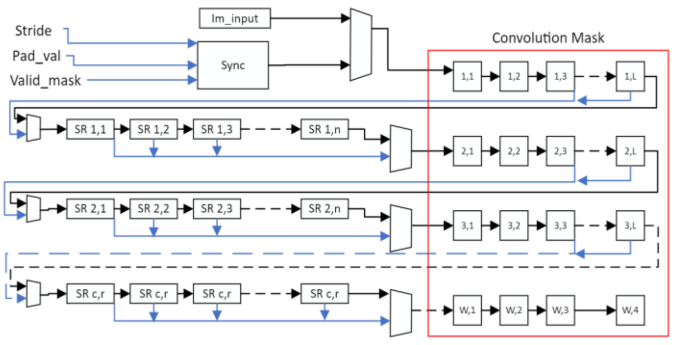

Fig. 1. Structure for filling input image data (Line buffer)

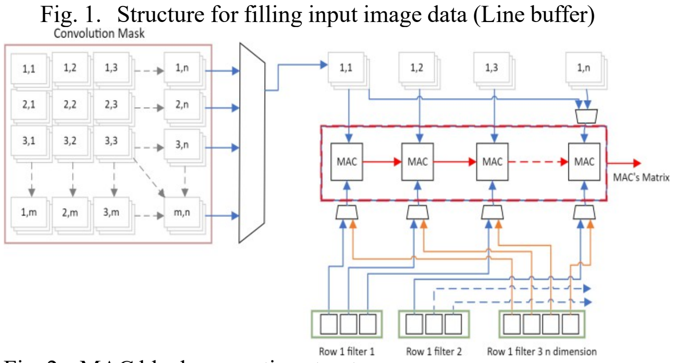

Fig. 2. MAC block computing structure.

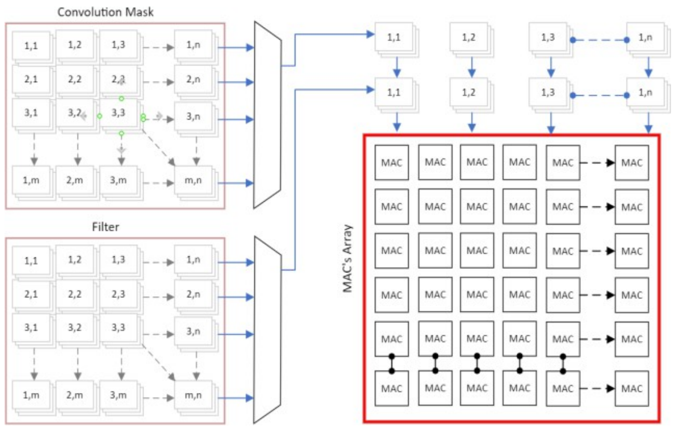

Fig. 3. MAC block matrix definition.

B. *MAC blocks*

The MAC blocks have the function of executing multiplications of their respective inputs and accumulating or adding the multiplied values in each clock cycle, or each time they are used. After the image data is updated in the convolution masks, their filtering is done through Multiply and Accumulate (MAC) blocks (Fig 2). The matrix data is separately selected row by row, and each row is taken data by data. Then the ith convolution mask data is multiplied with the ith filter matrix value and the results are accumulated to define the MAC matrix data (Fig 3). The filter matrix elements are constant coefficient values. Therefore, the total number of filters requires hardware resources. The filter matrix dimension must fit the convolution mask matrix dimension and, according to Fig. 2, multiple filters can simultaneously be applied.

Authorized licensed use limited to: VIT University. Downloaded on September 10,2024 at 16:07:17 UTC from IEEE Xplore. Restrictions apply.

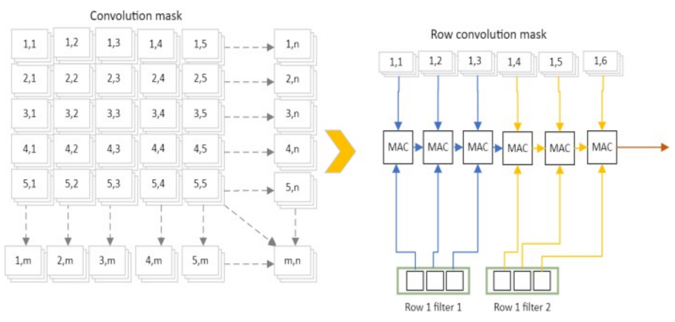

Fig. 4. Example how to multiply 2 filter of the same size at the same time

IV. Parallelism types analysis

The flexibility feature of the proposed structure is thanks to the added line buffer derivations allowing convolution mask expansion. Multiple multiplications applied between the convolution mask and the filter are also included. As can be observed such multiplication operation path can be configured according to the required. The filters dimensions can be resized allowing even the use of multiple filters with different dimensions for parallelism processing. This would yield to different values in the MAC blocks and therefore multiple result matrices.

A. Filter Parallelism

This type of parallelism refers to the number of filters that can be applied to the convolution mask at the same time. The layer of the proposed convolution mask structure can be either very large or very small, due to its configurability, which allows the option of being able to apply filters not only of the same dimension, but also apply filters with different dimensions for example 3x3 and 5x5 if the convolution mask is at least 5x5 (fig 4), this can be extrapolated to more and more filters. The convolution mask filling stage is strongly related to the following MAC blocks computing stage because if a parallel filter computing is designed simultaneously multiplication must be done between the filter row with the convolution mask row. A parallel filtering can be extended depending on the still available hardware elements.

This way of performing filter multiplication can be extrapolated to higher dimension and to the mixing of multiple filter sizes, this represents a reduction in the application of multiple filters on a convolution mask, however it should be noted that it is not the only form of parallelism in the convolution computing stage.

B. Operations Parallelism

Another possible form of parallelism in the convolution computing stage is when MAC operations are performed. If we consider that there is a convolution mask of any size and several filters to be applied, then the parallelism to be focused on is the multiplications between the elements of a single filter and a convolution mask. As such matrices elements can be considered independent, then the row multiplications of the filter and the convolution mask can be done at the same time (Fig 5), covering all the information of the input image multiplied be the filter according to the available hardware elements.

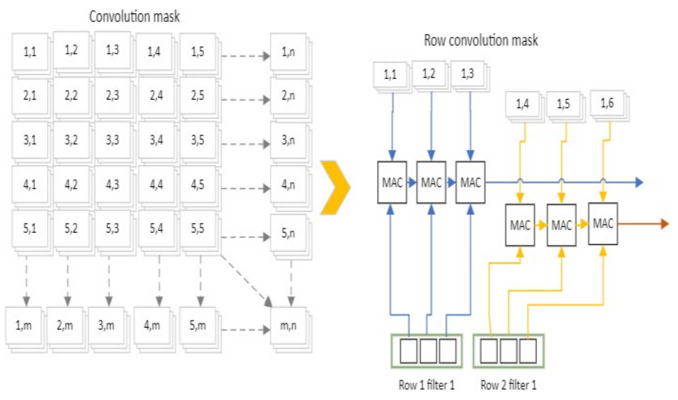

Fig. 5. Example of a simultaneously two row multiplication.

Similarly, this parallelism in operations can be extrapolated to a higher-dimensional filter and convolution mask, to cover as much information from the input image in the shortest possible time for the filter being applied.

C. Processing time and usable resources

The parallelism use in convolution computing hardware is where advantages are clearly observed. Operation required execution time and hardware resources are aspects commonly sought.

Considering a specific case where the input image has a dimension of 28 x 28 pixels and two 3x3 filters are applied.(In case you want to process a much larger image, the process is the same as described above, only that it is extrapolated to a longer processing time and a greater number of resources used, because as the information is much larger, the Line Buffer cannot contain all of it at the same time and must be sectioned to process the entire set.)

Two distinct stages can be distinguished in the convolution computing process. The first one where the convolution mask is filled, since at this stage the parallelism is not yet really applied, the time and resources of this stage can be considered fixed. The second stage is convolution computing process through the MAC blocks.

As the average operating clock frequency range of FPGAs is between 50 MHz and 500 MHz, in the following timing analysis a clock frequency value of 50 MHz is considered. An instruction should be ideally executed in one clock cycle, in order to have a practical approach analysis three clock cycles are taken as the execution time of one operation. Hence the one operation execution time in the developed timing analysis is 60ns.

In the convolution mask filling stage, the input image is read pixel by pixel until the filling is completed, then an execution time of 47040 ns or about 47 ms is required.

In the multiplication of the filters with the convolution mask, within the convolution computing stage, the following three cases are compared:

- In the first case the filter multiplications are sequentially done, this is one filter is applied after the previous one have been processed, so the process would take an execution time of 1080 ms.

Authorized licensed use limited to: VIT University. Downloaded on September 10,2024 at 16:07:17 UTC from IEEE Xplore. Restrictions apply.

TABLE 1. PROCESSING TIME AND REQUIRED RESOURCES.

Method	***Conv. Mask time***	***Resources***	***Filtering time***	***Resources***	***MAC Filter time***	***Resources***
Sequential	47ms	3 Shift Register and 9 memory elements	1.080ms	3 MAC Blocks	1.080ms	18 memory elements
Filter Parallelism	47ms	3 Shift Register and 9 memory elements	0.54ms	6 MAC Blocks	0.54ms	18 memory elements
Operation parallelism	47ms	3 Shift Register and 9 memory elements	0.54ms	6 MAC Blocks	0.54ms	18 memory elements
			0.36ms	9 MAC Blocks	0.36ms	18 memory elements

- In the second case the filter parallelism is used, so the process would take about 0.54 ms, since the number of resources can be used to multiply the convolution mask by the two filters at the same time.
- In the third case the operation parallelism is applied, where more resources are dedicated to make the multiplication faster, so the time is about 0.54 ms but if at this stage more resources are released, the time can be reduced progressively.

Each stage consumes a greater number of resources; however, we can see that the trend is the higher number of used resources, the smaller processing time, as shown in Table 1.

D. Parallelism benefits

- Processing speed: By distributing the workload across multiple cores or processing units, convolution can be performed faster, especially on large data sets.
- Hardware efficiency: When performing a task in parallel, it performs more operations using as many resources as possible to reduce operation time.
- Scalability: Parallelism allows scaling the convolution process according to the available hardware capabilities. This means that the number of cores or processing units can be increased to handle even larger data sets without sacrificing processing speed.
- Resource optimization: By distributing the workload efficiently among multiple processing units, the use of resources such as memory and compute capacity can be optimized.

When comparing the results obtained with other methods and algorithms, we found it difficult to compare with complete and designed structures, since they are used and developed with specific hardware and not in a conceptual and general way, so we could only compare with a general concept of conventional convolution.

V. CONCLUSIONS

The described hardware structure for image convolution computing is based on two proposed types of parallelism in the computing of the filtering and MAC operation stages, as well as a proposed structured and defragmented image input data. One of the main characteristics of the proposal is its flexible feature to easily include different dimensions of one same or multiple different filters. According to the reported performance analysis the use of the parallelism allows a reduced computing time compared to a completely sequentially computing mainly in the filtering and MAC blocks stages. The hardware structure features represent a great advantage when extrapolating this architecture to an FPGA or DPS implementation. The physical implementation of the described hardware structure in convolutional neural networks applications is left as future work.

REFERENCES

[1] An, F.; Wang, L.; Zhou, X. A High-Performance Reconfigurable Hardware Architecture for Lightweight Convolutional Neural Network. Electronics 2023, 12, 2847. https://doi.org/10.3390/electronics12132847

[2] Huang, J.; Liu, X.; Guo, T.; Zhao, Z. A High-Performance FPGA-Based Depthwise Separable Convolution Accelerator. Electronics 2023, 12, 1571. https://doi.org/10.3390/electronics12071571

[3] Hong, E.; Choi, K.-A.; Joo, J. Efficient Two-Stage Max-Pooling Engines for an FPGA-Based Convolutional Neural Network. Electronics 2023, 12, 4043. https://doi.org/10.3390/electronics12194043

[4] Ma, Y.; Xu, Q.; Song, Z. Resource-Efficient Optimization for FPGA-Based Convolution Accelerator. Electronics 2023, 12, 4333. https://doi.org/10.3390/electronics12204333

[5] P. Mousouliotis et al., Exploiting Vitis Framework for Accelerating Sobel Algorithm, 2021 10th Mediterranean Conference on Embedded Computing (MECO), Budva, Montenegro, 2021, pp. 1-5, doi: 10.1109/MECO52532.2021.9460221.

Authorized licensed use limited to: VIT University. Downloaded on September 10,2024 at 16:07:17 UTC from IEEE Xplore. Restrictions apply.