Bulletin of Electrical Engineering and Informatics
Vol. 15, No. 4, August 2026, pp. 3280~3290
ISSN: 2302-9285, DOI: 10.11591/eei.v15i4.11386

Hardware-level data layout approach to mitigate the memory row conflicts on FPGA-based CNN accelerators

Shalini Prasad[1], Suman Jayakumar[2], Bellary Kursheed[3], Aruna Mogarala Guruvaya[4], Rashmi Shivaswamy[5]

[1]Department of Electronics and Communication Engineering, City Engineering College, Bengaluru, India
[2]Department of Computer Science and Engineering, Sheshadripuram Institute of Technology, Mysuru, India
[3]Department of Computer Science and Engineering (Artificial Intelligence and Machine Learning), Don Bosco Institute of Technology, Bengaluru, India
[4]Department of Artificial Intelligence and Machine Learning, Dayananda Sagar College of Engineering, Bengaluru, India
[5]School of Computer Science and Engineering, RV University, Bengaluru, India

Article Info

Article history:

Received Sep 12, 2025
Revised May 6, 2026
Accepted Jul 9, 2026

Keywords:

Accelerator
Convolutional neural network
Field-programmable gate array
Pipeline mechanism
Resource optimization

ABSTRACT

Memory row conflicts (MRCs) continue to be a major bottleneck that results in higher latency, ineffective double data rate (DDR) usage, and decreased effective bandwidth in field-programmable gate array (FPGA-based) convolutional neural network (CNN) accelerators. The majority of current effort focuses on computational optimization, frequently ignoring inefficient memory access. In order to reduce memory reference codes (MRCs), this research suggests a hardware-level data layout technique using a memory-centric accelerator architecture. In order to improve hit rates and row buffer locality, the architecture incorporates a dynamic cursor-based address mapping method that adjusts to different feature map sizes across CNN layers and a dual-DDR setup for concurrent data access. The experimental results on VGG16, YOLOv2, and AlexNet show an 18% reduction in MRCs, a 40% increase in throughput, and a 23% decrease in latency compared to state-of-the-art techniques. The design uses a Xilinx Kintex-7 FPGA with low power usage of 1.52 W. The suggested method improves memory performance in FPGA-based CNN accelerators in a scalable and hardware-efficient manner without requiring a large computational burden.

Corresponding Author:

Shalini Prasad
Department of Electronics and Communication Engineering, City Engineering College
Kanakapura Main Rd, Doddakallasandra, Bengaluru, Karnataka 560062, India
Email: shaliniprasad5@gmail.com

1. INTRODUCTION

Convolutional neural networks (CNNs) are a key component of the advancements in deep learning (DL), which has attracted a lot of interest in the field of computation recently. CNNs have demonstrated impressive performance in a wide range of instances, such as semantic segmentation, recognition of objects, and image classification. Leaders in the sector have widely adopted CNNs as a result of their revolutionary influence, acknowledging them as an essential tool for development and superior competitiveness [1]. CNNs' advantage is their capacity for handling large amounts of data using non-linear processing, pooling, and convolutional processes, which makes computerized multi-level identification of features possible. Nevertheless, this ability entails higher resource requirements and complicated computations. CNNs have a difficult time performing well under resource limitations when they are used in portable and peripheral gadgets like handsets and aircraft [2]. Numerous studies have been done on hardware acceleration for CNNs

to tackle these issues. Although application-specific integrated circuits (ASICs), graphics processing units (GPUs), and central processing units (CPUs) are frequently used, their inability to manage the complexity of contemporary CNN structures and immediate processing demands has led to an increase in interest in substitute computing platforms. Field-programmable gate arrays (FPGAs), which provide energy savings, parallel computing, and flexible configuration, have become a viable option. Highly effective, resource-saving accelerators designed for DL algorithms are made possible by their capacity to modify hardware for particular workloads [3]. CNNs with lower model dimensions and fewer features, such as SqueezeNet, ShuffleNet, and MobileNet, operate comparably to bigger networks like ResNet-50 and AlexNet [4]. In the available research, a number of FPGA-based accelerator solutions have been created expressly to execute the you only look once (YOLO) set of computations [5]. The majority of the CNN's processing time is often spent on the convolution phase. The convolution's computational demands can be decreased without sacrificing accuracy using a number of quick techniques [6].

Convolutional procedures can be used to increase computing speed and accomplish data reuse because of their significant data duplication between neighbouring processes. Several data reuse strategies, such as no local reuse (NLR), input stationary (IS), output stationary (OS), and weighted stationary (WS), dataflow, were suggested by researchers for this specific reason [7]. Memory access effectiveness is still a recurring constraint, even though recent research has significantly improved compute speed through hardware acceleration and computational enhancements. Because non-contiguous memory accesses and frequent address transformations result in significant latency and energy expenses, this problem is particularly important in peripheral devices and massive implementations. The incidence of memory row conflicts (MRCs) is a major cause of these difficulties. When several parallel memory access attempts are routed to the identical dynamic random-access memory (DRAM) row, these disputes occur, leading to numerous row initiations and prior charges and ultimately higher latency [8], [9].

Problem statement*:* even though FPGA-based CNN accelerators have made great strides, recurrent MRCs' unproductive use of external double data rate (DDR) memory continues to be a serious and unresolved obstacle. Recurrent access to the same memory row causes these conflicts, which in turn limit accelerator capacity by causing frequent row activations, higher latency, and decreased effective bandwidth. Memory access behavior, in particular DDR row buffer locality and conflict prevention, has not received as much attention as functional optimization, parallelism, and data layout modifications. Additionally, additional access contention is introduced when neighboring feature maps of successive CNN layers are stored within the same DDR, which interferes with continuous dataflow during inference. Static address mapping solutions make this problem worse by failing to adjust to different feature map widths between layers, which leads to less-than-ideal access patterns. Conflict-free, high-throughput execution is further limited by the absence of DDR-aware designs with coordinated dual-memory access techniques. These drawbacks emphasize the necessity of a memory-focused, dynamically adaptive strategy that specifically addresses DDR access inefficiencies at the architecture level.

Contribution*:* an understudied bottleneck in FPGA-based designs, DDR memory inefficiencies, are directly addressed by the memory-centric CNN accelerator presented in this work. To facilitate concurrent data access and remove memory contention, the suggested architecture combines a dual-DDR framework with coordinated read/write scheduling. The dynamic cursor-based address mapping approach, which maximizes row buffer locality and dramatically reduces MRCs by adjusting to different feature map sizes across CNN layers, is a significant contribution. Furthermore, a layer-aware data scheduling system improves processor element (PE) utilization and maintains continuous dataflow during inference by aligning memory access patterns with DDR organization. In order to ensure interoperability across DDR3/DDR4/DDR5 standards and preserve scalability for a variety of CNN workloads, the design is further made hardware-independent.

This study differs from previous methods that mainly concentrate on computational optimization or static data layout alterations in that it employs a hardware-level, DDR-aware conflict prevention solution. Instead, than considering memory inefficiencies as side effects, the suggested technique specifically targets row hit-rate maximization and dynamic memory behavior. Adaptive address mapping and dual-memory design together offer a cohesive framework for reducing row conflicts and enhancing bandwidth usage, allowing for significant throughput and latency gains. This shows the work as a transition from traditional compute-centric optimization to systematic architectural memory access optimization. The organization of the paper is as follows: the proposed work, including the algorithm and architecture, is discussed in section 2. The results and discussion of the data layout approach accelerator, including synthesis results and comparative analysis, are highlighted in section 3. Lastly, the overall work is concluded in section 4.

Recent research on CNN accelerators across diverse hardware platforms highlights significant advances in computation, parallelism, and data handling. Kang *et al*. [10] proposed the on-the-fly lowering engine (OLE) to reduce convolution overhead by mapping it to general matrix multiplication (GEMM), achieving a 57.7% reduction in computation time and 2.3× average speedup. Wang *et al*. [11] introduced a parallelism strategy with group computation and channel shuffling, reducing memory usage by 34% while

maintaining accuracy and improving performance by 1.42×. He *et al*. [12] leveraged data reordering for efficient multiplier reuse, achieving 102.3 giga operations per second (GOPS) for visual geometry group 16 (VGG16) at 100 MHz using 512 multipliers. Similarly, Lee *et al*. [13] developed flexible diagonal cyclic arrays (FDCA), reaching 1.075 tera operations per second (TOPS) at 400 MHz with reduced inference latency. To enhance flexibility and resource utilization, Tong *et al*. [14] proposed a customizable CNN accelerator supporting diverse architectures, reducing performance and resource gaps by 63.7% and 59.6%, respectively. Fu *et al*. [15] introduced optimized data layouts (number-height-width-channel (NHWC), channel-height-width-number (CHWN), and channel-height-width-number-8 (CHWN8)) for im2win convolution, achieving up to 355% speedup and ~95% efficiency. Xu *et al*. [16] designed a low-power programmable accelerator using vector dot products, improving power efficiency by up to 23.989×. Neelam and Prince [17] presented the vision convolution framework (VCONV) framework, enabling scalable multi-FPGA deployment with 56 GOPS and minimal block random-access memory (BRAM) usage (1.64 Mb). Further, Sun *et al*. [18] utilized a 4×4×3 sliding window to reduce bandwidth, achieving 121.36 GOPS and 39.1× speedup over CPUs. Duarte *et al*. [19] demonstrated a co-optimized hardware–software design for real-time recognition with 62× acceleration and low power. Li *et al*. [20] proposed an efficient direct convolution method preserving channel-first data format, outperforming existing optimizations. Bosio *et al*. [21] explored FPGA-based DL with skip connections, achieving 2115 frames per second (FPS) on MobileNetV2 and demonstrating the efficiency of high-level synthesis (HLS) over traditional register-transfer level (RTL) designs.

2. METHOD

The memory module has a major impact on the efficiency of FPGA-based CNN accelerators because communication between peripheral DDR memories and on-chip processing elements (PEs) is a major performance constraint. MRCs, particularly lowering the efficient use of memory bandwidth and raising access latency, are one of the main problems. An adaptive MRC mitigation flow with a row hit-rate maximization technique and a dual-DDR-based convolution processor memory structure is introduced in the suggested approach to handle this. When combined, these tactics guarantee lower row disputes, better row hit-rate, and moderate usage of the PE.

2.1. Memory row conflict reduction using search algorithm

Figure 1 depicts the MRC minimization strategy. The search method that establishes the cursor values for the feature maps in each layer comes first. The logical address mapping system, which specifies the arrangement of rows, columns, and bank groups, incorporates these cursor values.

- Successive feature map items are often placed in the identical row of the identical bank in the conflict mapping strategy. Row conflicts and decreased memory throughput occur from the need for repeated row activations while seeking elements consecutively.
- Addresses the reorganization in the suggested cursor-based mapping method such that successive data are dispersed among several banks and rows. To ensure that every access may be served without the need for redundant row activation, for example, the mapping distributes consecutive addresses across rows rather than maintaining the ordered list (A0, A1, A0, and A1) in a single row.

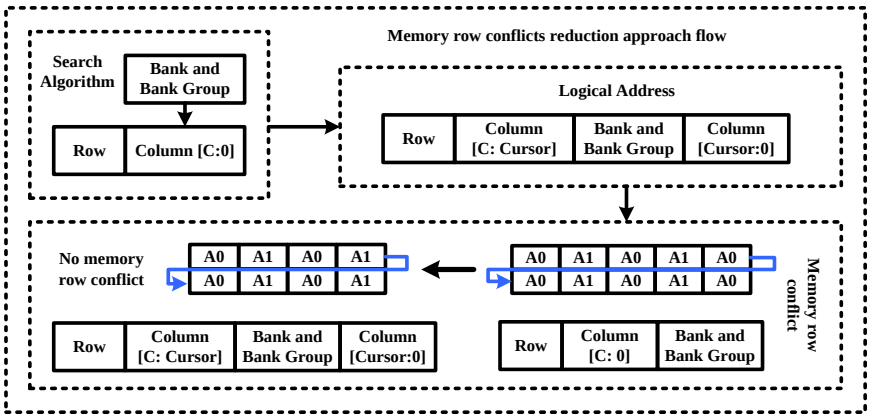

Figure 1. Flow of the MRC reduction approach

The system chooses the best cursor values based on the size of each layer's feature map by using Algorithm 1 before inference. This ensures that the successful row hit-rate is maximized and MRCs are kept to a minimum throughout CNN execution. Consequently, latency is decreased, PE usage stays moderately high, and DDR bandwidth usage is enhanced.

Let $H_f[i]$ and $W_f[i]$ be the dimensions of the input feature map of the ith CNN layer, mapped onto DDR tiles of size $H_t \times W_t$ times, where C is the number of candidate cursor offsets (i.e., column shifts regulating address dispersion over rows/banks). The suggested search algorithm computes a row hit-rate metric (Hit) based on alignment between feature map traversal and DDR row organization after evaluating each of the C candidate cursor positions per layer. In order to minimize row activations, the optimal cursor ($Cursor[i]$) is chosen as the one that maximizes row locality (Maxhit). The procedure is run offline before inference, resulting in little runtime overhead, and has linear complexity $O(n.C)$, where n is the number of CNN layers. In order to prevent repeated row activation, the chosen cursor values are integrated into a logical address mapping unit at the architectural level. This unit distributes successive feature map elements among DDR rows and banks. This is closely integrated with a dual-DDR memory architecture, which removes inter-layer storage conflicts and permits parallel read/write operations. Because the memory controller abstracts physical timing and protocol constraints, the mapping logic is hardware-neutral and compatible with DDR3/DDR4/DDR5. Overall, the technique maintains high PE usage, lowers memory access latency, and greatly increases row hit-rate with little control overhead, supporting the stated gains in system-level efficiency.

Algorithm 1. Search algorithm for row hit rate maximization

Input: $C, W_t, H_t, W_f[n], H_f[n]$
Output: $Cursor[n]$;
for $i = 0$; $i < n$; $i++$ **do**
$Cursor[i] = 0$;
$Maxhit = 0$;
for $j = 0$; $j < C$; $j++$ **do**
    Hit is obtained from $H_f[i], W_f[i], H_t, W_t$;
    if $Maxhit < Hit$ **then**
      $Maxhit = Hit$;
      $Cursor[i] = j$;
    end
end
end
Return $Cursor[n]$;

Algorithm 1 presents the first step of the suggested method, which is to optimize row hit-rate during memory access. For every feature map, the method iteratively looks through every conceivable column configuration. It then uses the feature height (H_f) and width (W_f) parameters, along with the memory tile dimensions (H_t, W_t), to evaluate the row hit-rate. Each candidate arrangement's hit value (Hit) is calculated by the algorithm and compared to the highest observed hit-rate ($Maxhit$). If a higher hit rate is obtained with the present setup, the algorithm adjusts $Maxhit$ as well as the corresponding column index pointer ($Cursor[i]$). The improved cursor array $Cursor[n]$, which represents the memory layout arrangement that maximizes data locality and avoids row conflicts, is finally constructed. The suggested approach includes adaptive logical address modification, in contrast to static mappings. The distribution of successive feature map data among several banks is guaranteed by the mapping approach. Furthermore, to effectively avoid row conflicts, the improved mapping approach divides each $2^{\text{Cursor}+1}$ entry among rows in the same bank. The best cursor values for each convolutional layer may be identified before execution starts, and the row hit-rate for each mapping method can be assessed offline due to the consistent and repetitive memory usage patterns involved in CNN inference. Thus, by striking a compromise between data location and row conflict prevention, Algorithm 1 makes sure the accelerator runs with layer-specific optimal memory layouts. By ensuring that subsequent accesses are in line with the memory row structure, this cursor-based mapping technique considerably lowers the number of duplicated row activations.

2.2. Convolution processor-based memory architecture

The FPGA-based convolution processor incorporates the improved cursor values into its hardware memory structure, as shown in Figure 2. Using two DDR memories with separate address locations, the suggested design avoids conflicts that arise from storing neighbouring feature maps of various levels in an identical DDR unit. While CNN is running, one DDR reads the feature maps of that particular layer while the other saves the feature maps of the subsequent layer. This alternate accessing pattern prevents storage conflicts and data overwrites while guaranteeing smooth operation. More precisely, both DDR0 and DDR1

are used for feature maps, while DDR0 is mostly used to hold weights. Weights and feature maps from the input are sent to the PEs by DDR0 in one phase, and the resulting feature maps are stored by DDR1.

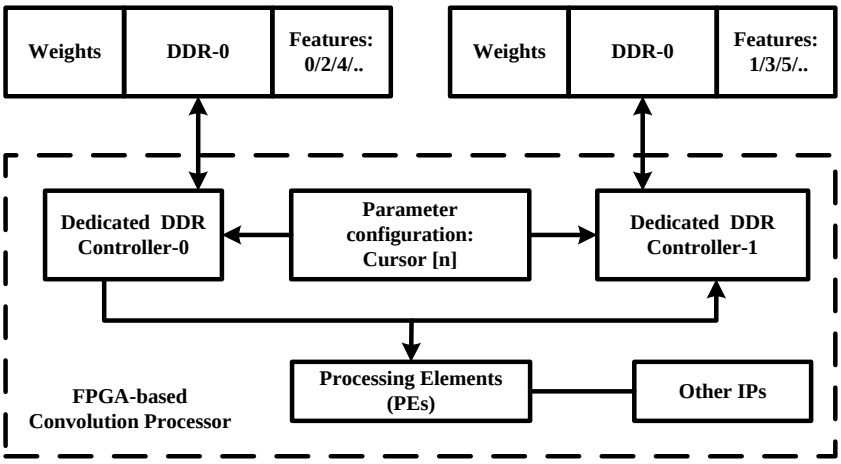

Figure 2. Convolution processor-based on the memory architecture

In order to enable concurrent reading and writing activities between the two modules, DDR0 is given another slot to store result feature maps in the following phase, while DDR1 supplies input feature maps. PEs will always receive data because of this design, which also makes sure the accelerator pipeline doesn't stall. Additionally, DDR3, DDR4, and DDR5 protocols are compatible with the memory module's design. Designed at the logical level, the address mapping logic abstracts away hardware-specific characteristics including bus width, data rate, error-correcting code (ECC), and pin count. Based on the selected DDR standard, the controller dynamically modifies scheduling policies and timing limitations at system initiation. Because of this, the suggested architecture can be used to a variety of hardware architectures without necessitating changes to the memory controller logic or optimization method. Several significant advantages result from the combination of the dual-DDR memory architecture, dynamic conflict reduction technique, and row hit-rate maximization algorithm: fewer MRCs because of the cursor-based mapping that is layer-specific. increased row hit-rate by the alignment of DDR architecture with memory access patterns. Alternate DDR read and write processes allow for smooth CNN implementation. increased PE utilization because data stalls are reduced by better memory scheduling. Cross-compatibility with DDR3/DDR4/DDR5 memories is accomplished by separating physical memory characteristics from logical mapping. scalability to a variety of CNN workloads because each convolutional layer's cursor values are dynamically adjusted.

3. RESULTS AND DISCUSSION

The suggested accelerator was created using the Xilinx Vitis HLS 2020.1 tool on the Xilinx Kintex-7 XC7K480T FFG1156 FPGA, which was made in a 28 nm process. Through the integration of two separate DDR4 memory modules controlled by adaptive address mapping controllers, the design minimizes MRCs and optimizes data access. Synthesis findings are included in the results section, which is followed by a discussion of MRC minimization and a comparison of the suggested work's performance with that of cutting-edge accelerators.

3.1. Synthesis results

Table 1 describes the suggested accelerator's post-synthesis resource usage on the Kintex-7 FPGA. The design uses 83,187 flip-flops (FFs) (13.93%) and 56,986 look-up tables (LUTs) (19.08%), demonstrating a balanced logic structure with plenty of growth potential. The number of dedicated resources used is quite minimal, since just 118 BRAM blocks (12.36%) and 162 digital signal processing (DSP) elements (8.44%) are used. This minimal DSP utilization demonstrates that the architecture has a large capacity for parallel computing scaling and is not compute-bound. Additionally, the small amount of BRAM used shows that the suggested hardware-level data layout technique is; the architecture reduces the need for huge on-chip

memory buffers to provide high throughput by effectively resolving DDR MRCs. The total sub-20% usage of all resource types attests to the viability of the concept and provides sufficient resources for further improvements.

Table 1. Resource usage of the proposed model on Kintex-7 FPGA

Resources	Usage	Available	Usage (%)
LUTs	56,986	298,600	19.08
FFs	83,187	597,200	13.93
BRAMs	118	955	12.36
DSP elements	162	1,920	8.44

Table 2 show the latency and throughput evaluation of the suggested accelerator with the most advanced compilers for three CNN models. Three common CNN models—VGG16, YOLOv2, and AlexNet—are used to compare the performance of the suggested accelerator to two cutting-edge platforms, OneDNN [22], and Vitis artificial intelligence (AI) [23]. Weights, operation frequency (200 MHz), the data input, pre-processing pipeline, and analytical precision (32-bit floating-point) are all the same under which the comparison is carried out. The outcomes show that the suggested hardware-level data layout technique consistently and significantly improves performance.

Table 2. Latency and throughput analysis using different compilers and CNNs

Compiler	VGG16	YOLOv2	AlexNet
Latency (ms)			
One DNN [22]	83.8	66.9	48.6
AI-based Vitis [23]	77.4	63.2	42.8
This work	65.4	52.1	36.2
Throughput (FPS)			
One DNN [22]	11.73	12.93	20.04
AI-based Vitis [23]	12.98	14.72	22.86
This work	16.88	21.23	27.56

The accelerator achieves the maximum throughput (in FPS) and the lowest latency among the three networks. When compared to OneDNN, the suggested work lowers inference latency by up to 18.4 ms (21.9%) for VGG16, 14.8 ms (22.1%) for YOLOv2, and 12.4 ms (25.5%) for AlexNet. Additionally, improvements are evident when compared to the more sophisticated Vitis AI baseline. Throughput increases significantly as a result. The suggested architecture outperforms OneDNN by 44.2% on YOLOv2 (21.23 vs. 12.93 FPS) and 43.9% on VGG16 (16.88 vs. 11.73 FPS). AlexNet has improved by 37.5%. All model architectures exhibit the same decrease in latency and increase in throughput. This performance improvement can be directly attributed to the effective resolution of MRCs, which minimizes memory access stalls and enhances compute unit efficiency, ultimately optimizing the external memory bandwidth.

Variations in feature map sizes, layer depth, and memory access patterns are responsible for the disparity in latency and throughput improvements across VGG16, YOLOv2, and AlexNet. Larger intermediate feature maps in deeper networks (like VGG16) result in greater baseline MRC rates, making the suggested mapping more beneficial. On the other hand, because of less intrinsic memory load, lighter models (like AlexNet) exhibit very modest increases. This data supports the efficacy of the suggested strategy by confirming that memory access intensity and row conflict probability are closely connected to performance improvement.

3.2. Memory row conflict reduction analysis

This work employs a customized DDR4-3200 (8-bank) memory model in conjunction with the SystemC modelling environment to assess MRCs. Classes (CLs) used from three popular CNNs—YOLOv2, AlexNet, and VGG16—form the basis of the simulations. The network architecture matches those of versions from the Darknet and Caffe Model Zoo that are made accessible to everyone. The ImageNet ILSVRC 2012 validating set and the Pascal visual object classes (VOC) 2007 collection, two well-known standards in the computer vision domain, are utilized as input datasets for creating feature maps for computation. Figure 3 shows the simulation outcomes of MRC (%) over CLs using the baseline and suggested models. The effectiveness of the suggested hardware-level data layout optimization is crucially shown by the quantitative measurement of the MRC rate, expressed as a percentage of total memory accesses. When contrasted to the baseline memory controller, the suggested approach consistently and significantly lowers MRC rates across

all CLs, as seen in the findings for VGG16, YOLOv2, and AlexNet. Regardless of the layer's depth or complexity, this reduction is consistently seen, highlighting the method's resilience and broad application.

The baseline MRC rate in VGG16 varies from 18.4% in early levels to a maximum of 32.5% in deeper layers like Conv-9. These rates are significantly decreased by the suggested approach; the largest reduction is shown at Conv-3, where the MRC decreases from 21.5% to 13.2% (Figure 3(a)). The baseline MRC rates in YOLOv2 also rise gradually from 12.2% at Conv-1 to 28.6% at Conv-12, but the suggested approach reduces these peaks, most notably the MRC at Conv-12 from 28.6% to 21.7% (Figure 3(b)). For AlexNet, the suggested modification reduces the baseline's maximum MRC rate at Conv-3 (27.9%) to 18.4% (Figure 3(c)). The effective data layout technique, which reduces row misses and bank conflicts in the DDR4 memory system, is directly responsible for these decreases. The suggested approach guarantees more consecutive and contiguous memory requests by rearranging data access patterns, which lowers latency and enhances bandwidth consumption. The improved performance metrics—lower inference latency and higher throughput—that were previously demonstrated are clearly correlated with the lower MRC rates, confirming that the memory bottleneck has been successfully removed. This illustrates that a basic issue with FPGA-based CNN acceleration is effectively resolved by the suggested modification, allowing for increased efficiency without consuming more resources.

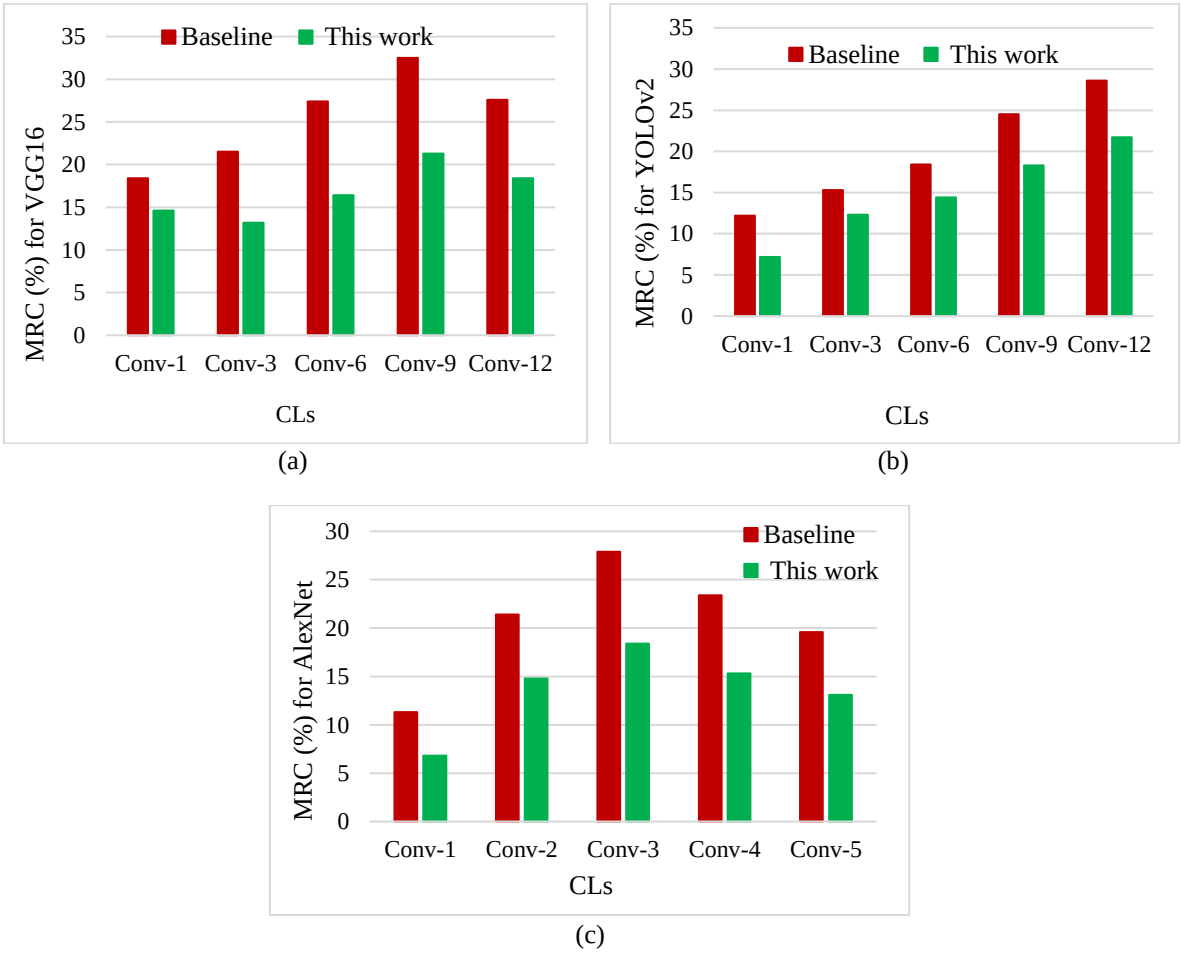

Figure 3. Simulation results of MRC (%) over CLs using baseline and proposed models for; (a) VGG16, (b) YOLOv2, and (c) AlexNet

3.3. Performance comparison

A thorough comparison of the suggested CNN accelerator with other cutting-edge implementations across important design and resource usage aspects is shown in Table 3. Interestingly, even though some of the compared designs were used in more sophisticated complementary metal-oxide-semiconductor (CMOS) processes (16 nm or 20 nm), the design manages a total power usage of 1.52 W, which is less than all the referred studies except [24] (1.599 W).

Table 3. Performance comparison of the proposed work with existing CNN accelerators

Parameters	Ref. [25]	Ref. [16]	Ref. [24]	Ref. [26]	Ref. [27]	This work
Design entry	HLS	RTL	RTL	RTL	HLS	HLS
FPGA	XCZU9EG	7A100T	VUC118	VUC 108	KCU1500	XC7K480
CMOS process (nm)	16	28	16	16	20	28
Clock (MHz)	100	200	200	100	200	200
LUTs (K)	61.72	101.4	120.2	33.64	131.28	56.98
FFs (K)	27.87	NA	33.02	27.76	166.22	83.19
DSP	123	240	955	176	315	162
BRAM	102	NA	43	1728	237	118
Power (W)	1.67	1.82	1.59	3.62	1.83	1.52

This demonstrates the suggested memory-centric optimization's energy efficiency. The use of resources is balanced and moderate. Compared to [16], [24], [27], the DSP count (162) is significantly lower, suggesting that effective memory access optimization—rather than excessive computational parallelism—is the primary means of achieving the performance increases. In a similar vein, BRAM consumption (118) is considerably lower than in [26] but greater than in [24], indicating a customized buffering strategy that helps the suggested data layout strategy without relying too heavily on on-chip memory. The proposed architecture offers a compelling blend of efficiency, performance, and power than existing CNN accelerators [16], [24]–[26]. When compared to the existing similar data layout and memory optimization [27] approach, the proposed work, offers minimal area, memory and power usage. The results show that the suggested solution provides better performance by explicit DDR-aware row conflict reduction, supporting its architectural benefit, in contrast to previous work [27] that mainly improve computation or static layouts.

The chosen CNN models (VGG16, YOLOv2, and AlexNet) ensure meaningful evaluation across a range of workloads by representing a variety of computational and memory access characteristics. The suggested approach is scalable and layer-agnostic since cursor optimization is carried out each layer according to feature dimensions, strengthening generalization. Furthermore, a theoretical connection is established: memory stalls have an inverse effect on throughput, and delay is proportional to row misses. The suggested approach lowers row activations by raising row hit-rate, which minimizes access latency and enhances efficient bandwidth utilization—both of which immediately result in increased throughput.

The term "DDR compatibility" has been defined as "design-level portability," which allows for controller-level adaption to support DDR3/DDR4/DDR5 by separating the logical address mapping from the physical DDR parameters. Although DDR4 is now the subject of experimental validation, this abstraction guarantees extensibility between standards. Furthermore, the following limitations have been made clear: i) the overhead of offline cursor calculation for very deep networks; ii) CNN inference's reliance on predictable memory access patterns; and iii) the possibility of diminished gains for compute-bound or highly irregular workloads.

4. CONCLUSION

The performance of FPGA-based CNN accelerators is severely hampered by MRCs in external DDR memory, which is addressed in this study. A dual-DDR architecture and an adaptive, layer-specific cursor-based address mapping mechanism are combined in a memory-centric optimization framework. While maintaining balanced and moderate resource use on the Kintex-7 FPGA, the results show a consistent decrease in MRC rates across several CNN models, resulting in ~40% improvement in throughput, and ~23% reduction in latency. This work focuses exclusively on hardware-level, DDR-aware memory access optimization, in contrast to current state-of-the-art methods that mostly stress computational optimization, parallelism, or static data layout modifications. The suggested approach successfully reduces row conflicts and improves memory bandwidth usage by specifically focusing on row buffer locality and dynamic access behavior across CNN layers, thereby resolving a largely unexplored problem in earlier approaches. Overall, this study provides a scalable and hardware-independent solution that closes the gap between FPGA accelerators' computational power and memory efficiency. Future studies will investigate real-time adaptive mapping for multi-network workloads, incorporate sparsity-aware scheduling for optimal memory access in compressed CNN models, and expand the suggested framework to handle cutting-edge memory technologies like DDR5 and high bandwidth memory (HBM).

FUNDING INFORMATION

Authors state no funding involved.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Shalini Prasad	✓	✓		✓	✓	✓			✓	✓			✓	
Suman Jayakumar		✓		✓		✓		✓	✓	✓	✓			
Bellary Kursheed	✓		✓	✓			✓			✓	✓		✓	
Aruna Mogarala Guruvaya	✓	✓		✓	✓		✓	✓	✓	✓	✓	✓		
Rashmi Shivaswamy		✓	✓	✓	✓	✓		✓	✓			✓		

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

Data availability does not apply to this paper as no new data were created or analysed in this study.

REFERENCES

[1] J. Jiang, Y. Zhou, Y. Gong, H. Yuan, and S. Liu, "FPGA-based acceleration for convolutional Neural Networks: A comprehensive review," *arXiv preprint*, 2025, doi: 10.48550/arXiv.2505.13461.

[2] C. Li, Y. Yang, M. Feng, S. Chakradhar, and H. Zhou, "Optimizing Memory Efficiency for Deep Convolutional Neural Networks on GPUs," in *SC16: International Conference for High Performance Computing, Networking, Storage and Analysis*, Nov. 2016, pp. 633–644, doi: 10.1109/SC.2016.53.

[3] J. S. Fata, W. M. Elmannai, and K. Elleithy, "Balancing Performance and Cost—FPGA-Based CNN Accelerators for Edge Computing: Status Quo, Key Challenges, and Prospective Innovations," *IEEE Access*, vol. 13, pp. 142852–142877, 2025, doi: 10.1109/ACCESS.2025.3594348.

[4] D. Ghimire, D. Kil, and S. Kim, "A Survey on Efficient Convolutional Neural Networks and Hardware Acceleration," *Electronics*, vol. 11, no. 6, p. 945, Mar. 2022, doi: 10.3390/electronics11060945.

[5] Z. Wang, K. Xu, S. Wu, L. Liu, L. Liu, and D. Wang, "Sparse-YOLO: Hardware/Software Co-Design of an FPGA Accelerator for YOLOv2," *IEEE Access*, vol. 8, pp. 116569–116585, 2020, doi: 10.1109/ACCESS.2020.3004198.

[6] Y. Zhao, D. Wang, and L. Wang, "Convolution Accelerator Designs Using Fast Algorithms," *Algorithms*, vol. 12, no. 5, pp. 1-14, May 2019, doi: 10.3390/a12050112.

[7] Y. Liu *et al.*, "Design of a Convolutional Neural Network Accelerator Based on On-Chip Data Reordering," *Electronics*, vol. 13, no. 5, pp. 1-17, Mar. 2024, doi: 10.3390/electronics13050975.

[8] W. Jiang, H. Yu, X. Liu, and Y. Ha, "Energy Efficiency Optimization of FPGA-based CNN Accelerators with Full Data Reuse and VFS," in *2019 26th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, Nov. 2019, pp. 446–449, doi: 10.1109/ICECS46596.2019.8964717.

[9] D. T. Nguyen, H. Je, T. N. Nguyen, S. Ryu, K. Lee, and H.-J. Lee, "ShortcutFusion: From Tensorflow to FPGA-Based Accelerator With a Reuse-Aware Memory Allocation for Shortcut Data," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 69, no. 6, pp. 2477–2489, Jun. 2022, doi: 10.1109/TCSI.2022.3153288.

[10] M. Kang, S. Hyun, T. H. Han, J. Kim, and S. Hong, "On-the-Fly Lowering Engine: Offloading Data Layout Conversion for Convolutional Neural Networks," *IEEE Access*, vol. 10, pp. 79730–79746, 2022, doi: 10.1109/ACCESS.2022.3192618.

[11] J. Wang, W. Tong, and X. Zhi, "Model Parallelism Optimization for CNN FPGA Accelerator," *Algorithms*, vol. 16, no. 2, pp. 1-13, Feb. 2023, doi: 10.3390/a16020110.

[12] J. He, M. Zhang, J. Xu, L. Yu, and W. Li, "Optimizing CNN Hardware Acceleration with Configurable Vector Units and Feature Layout Strategies," *Electronics*, vol. 13, no. 6, pp. 1-25, Mar. 2024, doi: 10.3390/electronics13061050.

[13] D.-Y. Lee *et al.*, "High-Speed CNN Accelerator SoC Design Based on a Flexible Diagonal Cyclic Array," *Electronics*, vol. 13, no. 8, pp. 1-18, Apr. 2024, doi: 10.3390/electronics13081564.

[14] H. Tong, K. Han, S. Han, and Y. Luo, "Design of a Generic Dynamically Reconfigurable Convolutional Neural Network Accelerator with Optimal Balance," *Electronics*, vol. 13, no. 4, pp. 1-23, Feb. 2024, doi: 10.3390/electronics13040761.

[15] X. Fu, X. Zhang, J. Ma, P. Zhao, S. Lu, and X. T. Liu, "High Performance Im2win and Direct Convolutions Using Three Tensor Layouts on SIMD Architectures," in *2024 IEEE High Performance Extreme Computing Conference (HPEC)*, Sep. 2024, pp. 1–7, doi: 10.1109/HPEC62836.2024.10938425.

[16] Y. Xu, J. Luo, and W. Sun, "Flare: An FPGA-Based Full Precision Low Power CNN Accelerator with Reconfigurable Structure," *Sensors*, vol. 24, no. 7, pp. 1-21, Mar. 2024, doi: 10.3390/s24072239.

[17] S. Neelam and A. A. Prince, "VCONV: A Convolutional Neural Network Accelerator for FPGAs," *Electronics*, vol. 14, no. 4, pp.

1-20, Feb. 2025, doi: 10.3390/electronics14040657.
[18] Y. Sun, Y. Ma, Z. Chen, Z. Liu, B. Chen, and R. Song, "A Sliding Window for Data Reuse in Deep Convolution Operations to Reduce Bandwidth Requirements and Resource Utilization," *Electronics*, vol. 14, no. 3, pp. 1-13, Feb. 2025, doi: 10.3390/electronics14030582.
[19] R. P. Duarte, T. Gonçalves, G. Jacinto, P. Flores, and M. Véstias, "Design of Real-Time Gesture Recognition with Convolutional Neural Networks on a Low-End FPGA," *Electronics*, vol. 14, no. 17, pp. 1-22, Aug. 2025, doi: 10.3390/electronics14173457.
[20] S. Li, F. Yu, S. Zhang, H. Yin, and H. Lin, "Optimization of Direct Convolution Algorithms on ARM Processors for Deep Learning Inference," *Mathematics*, vol. 13, no. 5, pp. 1-19, Feb. 2025, doi: 10.3390/math13050787.
[21] R. Bosio *et al.*, "NN2FPGA: Optimizing CNN Inference on FPGAs With Binary Integer Programming," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 44, no. 5, pp. 1807–1818, May 2025, doi: 10.1109/TCAD.2024.3507570.
[22] J. Li *et al.*, "oneDNN Graph Compiler: A Hybrid Approach for High-Performance Deep Learning Compilation," in *2024 IEEE/ACM International Symposium on Code Generation and Optimization (CGO)*, Mar. 2024, pp. 460–470, doi: 10.1109/CGO57630.2024.10444871.
[23] U. Xilinx/AMD, San Jose/Santa Clara, California, "Vitis AI Model Zoo," 2021. https://github.com/Xilinx/Vitis-AI/tree/master/model_zoo. (Accessed Sep. 09, 2025).
[24] R. T. Syed, Y. Zhao, J. Chen, M. Andjelkovic, M. Ulbricht, and M. Krstic, "FPGA Implementation of a Fault-Tolerant Fused and Branched CNN Accelerator With Reconfigurable Capabilities," *IEEE Access*, vol. 12, pp. 57847–57862, 2024, doi: 10.1109/ACCESS.2024.3392240.
[25] M. Cho and Y. Kim, "FPGA-Based Convolutional Neural Network Accelerator with Resource-Optimized Approximate Multiply-Accumulate Unit," *Electronics*, vol. 10, no. 22, pp. 1-16, Nov. 2021, doi: 10.3390/electronics10222859.
[26] D. Filippas *et al.*, "A High-Level Synthesis Library for Synthesizing Efficient and Functional-Safe CNN Dataflow Accelerators," *IEEE Access*, vol. 12, pp. 57194–57208, 2024, doi: 10.1109/ACCESS.2024.3390422.
[27] Y. Wang and H. Zhao, "An Improved Strategy for Data Layout in Convolution Operations on FPGA-Based Multi-Memory Accelerators," *Electronics*, vol. 14, no. 11, pp. 1-12, May 2025, doi: 10.3390/electronics14112127.

BIOGRAPHIES OF AUTHORS

Shalini Prasad is a Professor at City Engineering College, Doddakallasandra, Bangalore, with over 20 years of teaching experience. Her research expertise lies in mobile engineering with a focus on energy-efficient computational models for mobile applications. She is a member of ISTE, IERP, and InSc, and has published around 30 research papers in reputed national and international journals. She has been honored with prestigious awards including the Best Senior Faculty Award (2020–21), InSc Research Excellence Award (2020–21), and Intel India Conclave Quiz Award on 5G and 6G (2023). Under her guidance, student projects have won accolades such as the 2nd prize at Dr. APJ Abdul Kalam Innovation Ecosystem. She has also contributed as an editor for the book series Futuristic Trends in Network and Communication Technologies (2024). With multiple patents to her credit and experience as a Ph.D. thesis examiner, she continues to actively contribute as a researcher, mentor, and resource person in faculty and student development programs. She can be contacted at email: shaliniprasad5@gmail.com.

Suman Jayakumar is serving as Associate Professor, Department of Computer Science and Engineering at Seshadripuram Institute of Technology, Mysuru. She earned her Ph.D. in Computer Science from VTU and has served in various capacities, including as a Software Engineer, Specialist Learning, and Assistant Professor. She has over 16 years of experience in the IT industry and Academia. She has authored over 10 peer-reviewed publications in international journals and conferences. Her research interests include cloud computing, virtualization, software-defined networking, and artificial intelligence. She can be contacted at email: jayakumarsuman@gmail.com.

Bellary Kursheed is currently serving as Associate Professor in Department of Computer Science and Engineering (Artificial Intelligence and Machine Learning) at Don Bosco Institute of Technology, Bangalore. She earned her Ph.D. in Electronics and Communication Engineering from Visvesvaraya Technological University, Belagavi. She received her M.Tech. degree in the field of Computer Networks from Visveswaraya Technological University, Belagavi. Experienced as Academician from 21 years. She has contributed to the academic community by publishing many national and international journals/conferences. Had received 3 Lakhs funds from Government of Karnataka Vision Group on Science and Technology for the project "RFID based vehicle antitheft and automatic toll collection system" and received fund from KSTA and KSCST three times. Area of interest includes wireless communication, multimedia communication, cognitive radio, and micro controllers. She can be contacted at email: bkursheedofficial@gmail.com.

Aruna Mogarala Guruvaya is as Professor in Artificial Intelligence and Machine Learning at Dayananda Sagar College of Engineering, Bengaluru. She holds a Doctor of Engineering degree from Vishwesvaraya Technological University, Belgum in 2022 and has expertise in data security cloud computing, and AI/ML. Her research focuses on enhancing system security and applying intelligent algorithms to solve complex problems. She has published multiple papers and contributed to advancements in predictive analytics and computing technologies. She is active member of academic and research communities like Vidwan and Research Gate. She can be contacted at email: mgaruna@gmail.com.

Rashmi Shivaswamy is an academician and researcher with over two decades of experience in Computer Science and Engineering. Currently a Professor at RV University, Bengaluru, she has held several academic and leadership roles, including Head of the Department and Chairman of Boards of Studies and Examiners at reputed institutions. Her expertise spans cloud computing, AI, data science, and emerging technologies. She guides research scholars, authors impactful journal publications, and files patents in assistive technology and healthcare innovation. She is a recognized IEEE Senior Member. She can be contacted at email: rashmineha.s@gmail.com.