DEFA: Efficient Deformable Attention Acceleration via Pruning-Assisted Grid-Sampling and Multi-Scale Parallel Processing

Yansong Xu[1], Dongxu Lyu[1], Zhenyu Li[1], Yuzhou Chen[1], Zilong Wang[1], Gang Wang[1], Zhican Wang[1], Haomin Li[1], Guanghui He[1,2*]

[1]School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, China

[2]MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University, China

{xys-13,lvdongxu,ambitious-lzy,huygens,wangzilongsjtu,wangganginJSTU,Zhican_wang,haominli,guanghui.he}@sjtu.edu.cn

ABSTRACT

Multi-scale deformable attention (MSDeformAttn) has emerged as a key mechanism in various vision tasks, demonstrating explicit superiority attributed to multi-scale grid-sampling. However, this newly introduced operator incurs irregular data access and enormous memory requirement, leading to severe PE under-utilization. Meanwhile, existing approaches for attention acceleration cannot be directly applied to MSDeformAttn due to lack of support for this distinct procedure. Therefore, we propose a dedicated algorithm-architecture co-design dubbed DEFA, the first-of-its-kind method for MSDeformAttn acceleration. At the algorithm level, DEFA adopts frequency-weighted pruning and probability-aware pruning for feature maps and sampling points respectively, alleviating the memory footprint by over 80%. At the architecture level, it explores the multi-scale parallelism to boost the throughput significantly and further reduces the memory access via fine-grained layer fusion and feature map reusing. Extensively evaluated on representative benchmarks, DEFA achieves 10.1-31.9× speedup and 20.3-37.7× energy efficiency boost compared to powerful GPU platforms. It also rivals the related accelerators by 2.2-3.7× energy efficiency improvement while providing pioneering support of MSDeformAttn.

KEYWORDS

Transformer, Deformable Attention, Pruning, Domain-Specific Acceleration, Grid-Sampling

1 INTRODUCTION

DEtection TRansformer (DETR) has gained increasing popularity in object detection due to the promising performance from the end-to-end optimizable network architecture. Recently, multi-scale deformable attention (MSDeformAttn) [1], inspired by deformable convolution (DeformConv) [2], is proposed to further improve the DETR resolution on small objects with linear complexity, which only samples a small set of key points from multi-scale feature maps (fmaps) instead of traversing across all of them via $O(n^2)$-level $\mathbf{Q}\times\mathbf{K}^T$ in traditional attention [3]. Benefiting from multi-scale grid-sampling (MSGS), Deformable DETR achieves state-of-the-art detection accuracy so that MSDeformAttn has been widely adopted in recent 2D [4, 5, 1] and 3D [6, 7, 8] object detection networks.

However, MSDeformAttn struggles with great computation inefficiency on deployment compared to convolution neural networks (CNN), especially on general-purposed platforms, like CPUs and GPUs. For instance, Deformable DETR (173GFLOPs) [1] executes at only 9.7fps even on a powerful Nvidia RTX 3090Ti GPU, while Faster R-CNN[9] with a similar workload (180GFLOPs) can reach over 25fps for the same task. With our deep analysis, MSDeformAttn takes up to 54.7% of end-to-end inference latency while the MSGS procedure dominates (over 60%) within each attention layer, which becomes the major efficiency bottleneck. Suffering from the dynamically unordered and unbounded sampling candidates all over the feature maps, grid-sampling results in heavily irregular memory access and high cache miss rate, further leading to severe PE under-utilization on GPUs. To make it even worse, applying multi-scale feature maps increases both the number of sampling points and the size of sampled fmaps by 21.3× compared to using single-scale ones, exacerbating the intensity of memory footprint.

Domain-specific accelerator (DSA) is an effective solution to improve the processing efficiency on resource-limited terminals. Many existing DSAs [10, 11, 12] have been proposed to optimize attention [3] via reducing redundant computation between weak-related tokens. Nevertheless, MSDeformAttn reconstructs vastly different attention pipelines based on MSGS so that these works cannot maintain the superiority of MSDeformAttn due to a lack of support for grid-sample. Although some works [13, 14] also enhance similar grid-sample-based operators like deformable convolution[2], their methods, especially the aggressive sampling range restriction induce unacceptable accuracy loss and also cannot be applied MSDeformAttn dataflow directly.

As a result, we propose **DEFA**, a dedicated MSDeformAttn accelerator with algorithm-architecture co-optimization. The main contributions are as follows:

(1) We comprehensively characterize the performance bottlenecks of MSDeformAttn in deformable transformers and identify the root cause of deployment inefficiency.

*Corresponding author: Guanghui He.

DAC '24, June 23–27, 2024, San Francisco, CA, USA

ACM ISBN 979-8-4007-0601-1/24/06…$15.00

https://doi.org/10.1145/3649329.3657328

(2) At the algorithm level, we propose a pruning-assisted grid-sampling scheme by deeply exploiting the sampling redundancy. To reduce the memory requirement of sampled fmaps, frequency-weighted fmap pruning (FWP) is adopted to ignore 43% unimportant pixels based on inter-layer pixel-wise acquisition. Meanwhile, probability-aware point pruning (PAP) is also applied to attain 84% memory access reduction through softmax-based sparsity exploitation.
(3) At the hardware level, we design an efficient MSDeformAttn architecture with fully harnessing the performance gains from algorithm-level optimization. First of all, we decouple the intra-level sampling computation to explore the multi-scale parallelism for bank conflict elimination, boosting the MSGS throughput by 3.06×. Furthermore, fine-grained operator fusion is adopted within each MSDeformAttn layer to avoid heavy data movement between sample and aggregation, which benefits from a reconfigurable PE array for MSGS and matrix computation.
(4) Implemented on 40nm technology and extensively evaluated on representative benchmarks, MSDeformAttn achieves up to 10.1-31.9× speedup and 20.3-37.7× energy efficiency improvement over Nvidia RTX 2080Ti & 3090Ti GPUs. Compared with the related accelerators, it improves the energy efficiency by 2.2-3.7×, while supporting MSDeformAttn.

2 PRELIMINARY

2.1 Multi-Scale Deformable Attention

MSDeformAttn identifies relations between each query and a small set of sampling points in multi-scale fmaps $X \in \mathbb{R}^{N_{in} \times D_{in}}$. Let N_{in} and D_{in} denote the length of flattened feature maps from N_l levels ($N_{in} = \sum_{l=0}^{N_l-1} H_l \times W_l$) and hidden dimension of pixel vectors respectively. The computation of MSDeformAttn is shown as:

$$\begin{aligned} MSDeformAttn(\mathbf{Q}, \mathbf{P}, \mathbf{X}) &= Concat(\mathbf{H}_0, ..., \mathbf{H}_{N_h-1}) \\ \text{where} \quad \mathbf{H}_{ij} &= Softmax(\mathbf{Q}_i \mathbf{W}_j^A)\mathbf{V}_j(\mathbf{P}_i + \Delta\mathbf{P}_{ij}) \\ \mathbf{V} &= \mathbf{X}\mathbf{W}^V, \quad \Delta\mathbf{P} = \mathbf{Q}\mathbf{W}^S \end{aligned} \tag{1}$$

where $\mathbf{Q} \in \mathbb{R}^{N_{in} \times D_{in}}$ and $\mathbf{V} \in \mathbb{R}^{N_{in} \times D_{in}}$ denote the query matrix and the multi-scale fmaps respectively, i indexes the row vector in the matrix and j indexes the N_h heads. $\mathbf{W}^A \in \mathbb{R}^{D_{in} \times D_{in}}$, $\mathbf{W}^V \in \mathbb{R}^{D_{in} \times D_{in}}$ and $\mathbf{W}^S \in \mathbb{R}^{D_{in} \times (2N_h N_l N_p)}$ are all learnable weights, where N_p denotes the fixed number of sampling points in each level of fmaps. $\mathbf{H}_{ij} \in \mathbb{R}^{N_{in} \times D_h}$ ($D_h = D_{in}/N_h$) is the i^{th} row vectors of the output matrix in the j^{th} attention head. $\mathbf{P}_i \in \mathbb{R}^{N_{in} \times N_l \times 2}$ enumerates N_{in} coordinates of regular grids over each level fmap.

Figure 1 (a) illustrates the whole procedure of MSDeformAttn. In each head of MSDeformAttn, the softmax unit normalizes all points in different levels projected from a row vector of the input query and generates the attention probability vector. The MSGS procedure adopts the bilinear interpolation (BI) kernel to process the fractional sampling points and obtains sampling value from the multi-scale fmaps. In the aggregation stage, each pixel vector of the sampling value is multiplied by an element of the attention probability vector and then all the weighted vectors are summed to gain an output vector of a head.

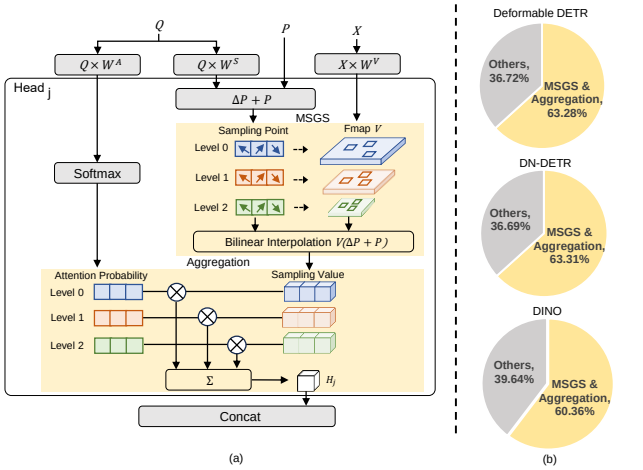

Figure 1: (a) Introduction of MSDeformAttn. (b) MSDeformAttn latency breakdown.

2.2 Computational Properties Analysis

As shown in Figure 1 (b), We profile the MSDeformAttn latency breakdown on Deformable DETR [1], DN-DETR [4] and DINO [5] on Nvidia RTX 3090Ti. MSGS and aggregation account for over 60% of the inference runtime of MSDeformAttn, while their computation cost only constitutes 3.25%, demonstrating severe inefficiency on GPU platforms. This is caused by irregular memory access in MSGS. In MSGS, unbounded sampling coordinates are dynamically generated and incur unpredictable memory access. It violates the locality principles typically used to reduce DRAM access and entails little fmap reuse. Moreover, since multi-scale fmaps in MSGS have more pixels than single-scale fmaps, the sampling points scatter in a wider range and this further increases memory access. In addition, The number of sampling points in MSGS multiplies by the same ratio as the pixel number of fmaps, which amplifies the inefficiency of the unordered sampling procedure.

Although DeformConv also applies a similar grid-sampling module, the workload of MSGS in MSDeformAttn is multiple times higher than DeformConv. Not only the multi-scale fmaps in MSDeformAttn are 21.3× larger than the single-scale fmaps in DeformConv, but the sampling points in MSDeformAttn are also $N_l N_p$× more than in DeformConv in each head. This incurs larger on-chip buffer requirement and more irregular memory access in MSGS, thus leading to higher inefficiency for MSDeformAttn processing.

Existing attention accelerators [11, 10, 12] reduce the computation of the tokens with weak relevance via random projection (ELSA [11]), attention score sort (SpAtten [10]) and approximate computation (BESAPU [12]). However, there is no explicit computation of token relevance in MSDeformAttn, so their methods are inapplicable to MSDeformAttn. Besides, they do not efficiently support the MSGS procedure and lack optimization on the dataflow of MSGS. In MSGS, the irregular memory access on multi-scale fmaps demands the attention accelerators for up to 9.8MB on-chip buffer size, which significantly enlarges the area and damages efficiency. DEFA is the first work that exploits sparsity in fmaps and sampling points to reduce computation and energy consumption. It also first

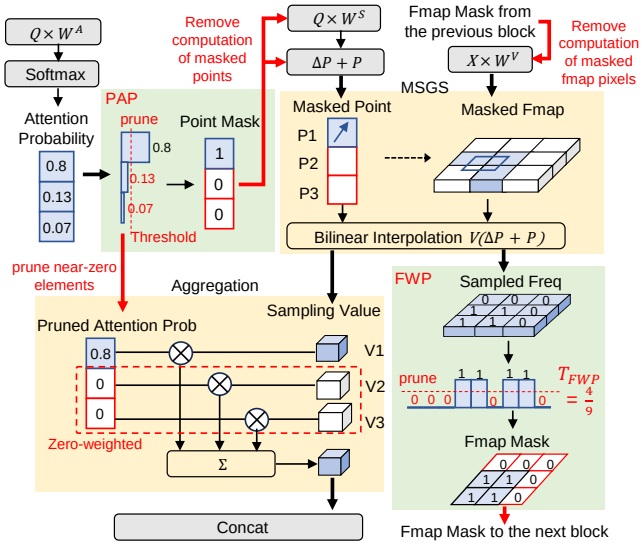

Figure 2: Overview of sparsity-aware grid-sampling.

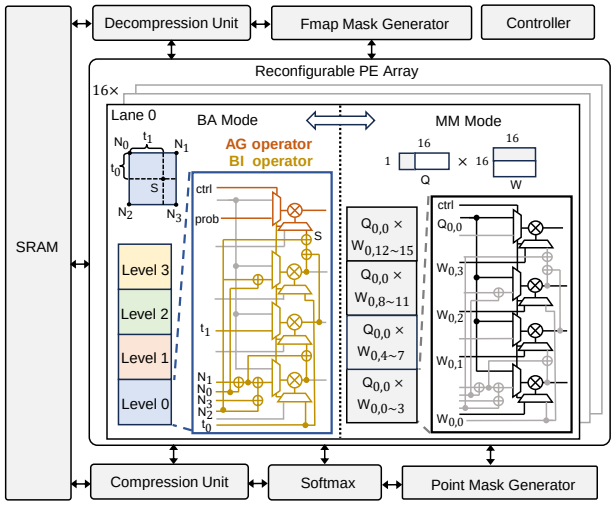

Figure 3: Overview of DEFA architecture.

explores the multi-scale parallelism in MSGS processing to enhance the throughput.

3 SPARSITY-AWARE GRID-SAMPLING FOR MSDEFORMATTN DATAFLOW

In MSGS, numerous fmap pixels and sampling points cause enormous memory access and have high redundancy. To remove unnecessary memory access and enhance the efficiency of MSGS, we exploit the sparsity in multi-scale fmaps and sampling points via FWP and PAP approaches respectively.

3.1 Frequency-Weighted Fmap Pruning

We introduce the sampled frequency of each pixel in multi-scale fmaps and propose FWP that restricts the size of large fmaps. The sampled frequency of each pixel varies significantly under non-uniform distribution in the multi-scale fmaps and the sampling points usually concentrate on a fixed portion of pixels along the continuous MSDeformAttn blocks based on our experiments. We find that the fixed portion of pixels has a much higher probability of being accessed and is essential to the detection accuracy, while the other pixels seldom accessed are redundant. Therefore, we measure the sampled freq of each pixel in fmaps and prune the pixels that have a lower sampled frequency than the defined threshold. In inference, FWP utilizes $k \cdot AF$ as the pruning threshold, where k is a hyperparameter to adjust sparsity and AF denotes *Average frequency = (sum of sampled frequency) / (sum of pixels)*. The locations of the pruned pixels are recorded in a bit mask as the fmap mask for the next MSDeformAttn block so that the linear projection of the masked fmap pixels can be eliminated. As shown in Figure 2 right, the neighboring points of the sampling point in BI are accessed once, so the sampled frequency of these pixels is counted to 1 and the sampled frequency of the others is 0. When the MSGS completes, *sum of sampled freq* is obtained as 4 and *the sum of pixels* is 9. Assuming that k equals 1, the threshold can be calculated as $\frac{4}{9}$. Then the pixels with the lower sampled freq than $\frac{4}{9}$ are recorded with 0 in the fmap mask.

3.2 Probability-Aware Point Pruning

We propose PAP to detect and remove unessential sampling points in MSGS using their related attention probabilities as illustrated in Figure 2 left. In the aggregation process, the sampling values in an attention head are multiplied by the normalized attention probabilities from softmax and then summed up. The summation of attention probabilities is confined to 1 and their differences are exponentially amplified. We set a threshold to filter out the near-zero attention probabilities and they constituted a dominant proportion (over 80% in Deformable DETR). The sampling values weighted by them make small contributions to the results of aggregation and detection accuracy. Hence, the near-zero attention probabilities are pruned and the point mask is established as a bit mask to eliminate the following processing of the sampling points generating the zero-weighted sampling values in the current MSDeformAttn block. In Figure 2 left, the zero-weighted sampling values $V2, V3$ are multiplied with 0 in the aggregation process after PAP, and thereby the sampling points $P2, P3$ producing $V2, V3$ are unnecessary and removed to save computation cost and memory access.

4 HIGH-THROUGHPUT DEFORMABLE ATTENTION ARCHITECTURE

In this section, we present DEFA architecture co-designed with the pruning algorithms to efficiently process MSDeformAttn in Figure 3. The fmap mask generator and sampling point mask generator implement FWP and PAP, respectively. The compression unit decompression unit eliminates the redundant bandwidth and computation of the masked data. The reconfigurable PE array can switch between the matrix multiplication (MM) mode and the BI mode to accelerate MM or fine-grained operator fusion.

4.1 Dataflow Overview

DEFA rearranges the operators in MSDeformAttn to save computation and memory access through the proposed pruning approaches.

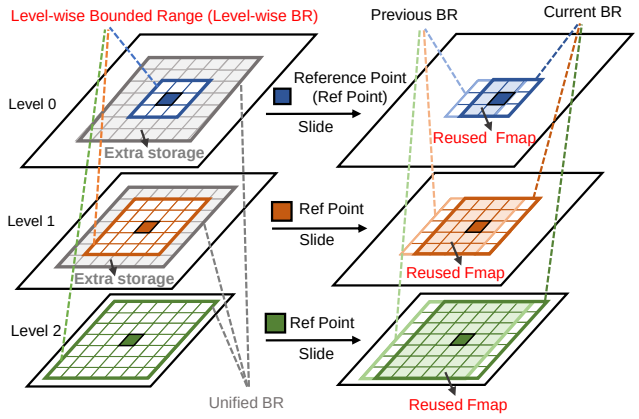

Figure 4: Level-wise range-narrowing scheme.

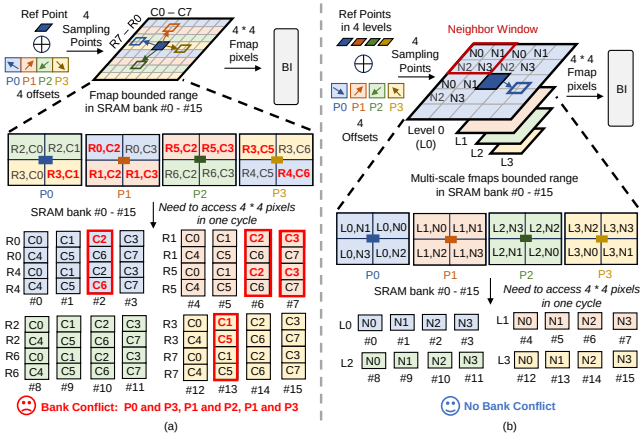

Figure 5: Illustration of (a) intra-level parallel processing and (b) inter-level parallel processing.

The attention probability is first calculated and the point mask updates. Then the reconfigurable PE array at the MM mode generates the sampling points and the fmaps pruned by the point mask and the fmap mask respectively. Finally, DEFA processes the fused MSGS and aggregation operators with the reconfigurable PE array at the BA mode. Meanwhile, the fmap mask generator receives the sampling address in the BI and runs FWP for the next block.

Inspired by the prior work on DeformConv [13], we propose the level-wise range-narrowing scheme to reduce on-chip storage in MSGS, as illustrated in Figure 4 left. DEFA adopts different sizes of bounded ranges to restrict the sampling offsets around a reference point according to the level of multi-scale fmaps, which is based on that the bounded range in some levels can be further shrunk without any accuracy loss. Directly applying unified restriction on all levels of the multi-scale fmaps causes 25% extra storage. Figure 4 right shows that when the reference point slides to the next pixel in fmaps, the overlapping pixels between the previous and the current bounded range are reused to avoid repetitive memory access.

4.2 Multi-Scale Parallel Processing

Figure 5 illustrates the multi-scale parallel processing of MSGS. As the MSGS on DEFA calculates the BI of 4 sampling points in parallel, the 16 neighboring fmap pixels of the sampling points need to be accessed from 16 SRAM banks in one cycle. If DEFA processes 4 sampling points in a level of multi-scale fmaps, which is intra-level parallel processing presented in Figure 5 (a), bank conflicts happen when multiple sampled pixels are stored in the same bank. Instead, we propose inter-level parallel processing to completely avoid bank conflicts in MSGS while attaining the same parallelism as the intra-level parallel processing. As shown in Figure 5 (b), DEFA processes 4 sampling points extended from 4 reference points in each level of the multi-scale fmaps respectively. The neighboring pixels of a sampling point named *Neighbor Window* are stored in 4 SRAM banks and the multi-scale fmaps are arranged in every 4 banks of the overall 16 SRAM banks. Because the sampling points only scatter in the specific level of multi-scale fmaps where their reference points are located, the neighboring pixels of the multi-level sampling points are stored in different banks. Therefore, the inter-level parallel processing eliminates bank conflicts in the intra-level parallel processing and achieves higher throughput.

4.3 Fine-Grained Operator Fusion with the Reconfigurable PE Array

We propose fine-grained operator fusion of MSGS and aggregation, removing the off-chip transfer of sampling value. To support the process with the limited computing resources in the PE array, we transform BI and design a reconfigurable PE array alternating between the BA mode and the MM mode, as shown in Figure 3. Assume there is a sampling point, denoted as S at (x, y), with four neighboring points denoted as N_0 at (x_0, y_0) (top left), N_1 at (x_1, y_0) (top right), N_2 at (x_0, y_1) (bottom left) and N_3 at (x_1, y_1) (bottom right) respectively. Then BI is implemented as:

$$\begin{aligned} S = N_0(x_1 - x)(y_1 - y) + N_1(x - x_0)(y_1 - y) \\ + N_2(x_1 - x)(y - y_0) + N_3(x - x_0)(y - y_0) \end{aligned} \tag{2}$$

As the coordinates of the four neighboring points are all integers and located at the four corners of the grid, x_1 and y_1 are equal to $x_0 + 1$ and $y_0 + 1$ respectively. After replacing them and a series of transformations, Eq.2 becomes:

$$\begin{aligned} S = N_0 + (N_2 - N_0)t_0 + [(N_1 - N_0) \\ + (N_3 - N_2 - N_1 + N_0)t_0]t_1 \end{aligned} \tag{3}$$

Where t_0 = y - y_0 and t_1 = x - x_0. The calculation of t_0 and t_1 is performed in other units. As a result, the BI operator part in Figure 3 only employs three multipliers and seven adders. The AG operator part performs the multiplication of attention probability and S. Additionally, the PE array in the MM mode conducts MM between a 16-element vector (Q) and a 16 × 16 tile (W) in the output-stationary dataflow.

5 EVALUATION

5.1 Experimental Methodology

5.1.1 Benchmarks. We evaluate DEFA on MSDeformAttn layers in the encoders of Deformable DETR (De DETR) [1], DN-DETR [4] and DINO [5]. The evaluation task is object detection on the COCO 2017 dataset [15]. We utilize PyTorch to conduct experiments on the benchmarks with reference to the official implementation. Fine-tuning is applied to the modified models by the software methods

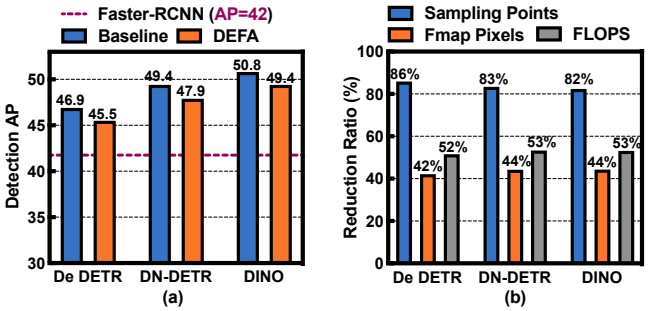

Figure 6: (a) Detection Average Precision of our methods and comparisons with other works. (b) Reduction in sampling points, fmap pixels, and computation cost.

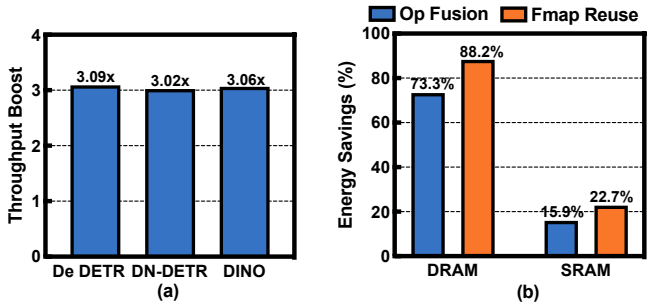

Figure 7: (a) MSGS throughput boost of inter-level parallel processing over intra-level parallel processing. (b) Energy savings of fine-grained operator fusion and fmap reuse.

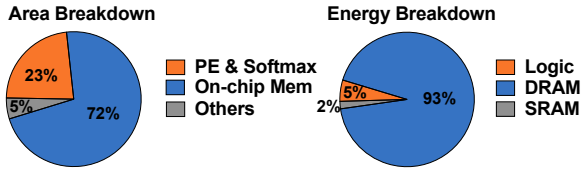

Figure 8: Area breakdown and energy breakdown of DEFA.

in Section 3 to recover the accuracy. The MSDeformAttn modules in the encoder layers of the models are quantized to 12bits during the inference. We compare DEFA with NVIDIA RTX 2080Ti and 3090Ti GPUs and SOTA attention accelerators [11, 10, 12].

5.1.2 Hardware Implementation. DEFA is described in SystemVerilog and synthesized with Synopsys Design Compiler for 400MHz clock frequency to estimate the area and power under a 40nm technology. We implement a cycle-accurate simulator to model the computation and memory access and evaluate the performance of DEFA. We obtain the area and energy consumption of SRAM with CACTI [16], and a moderate 256GB/s HBM2 is used as the external memory system, consuming 1.2pJ/b [17] for data access.

5.2 Algorithm Evaluation

Figure 6 (a) presents the standard average precision (AP) of our methods and comparisons with the baseline of the benchmarks as well as the Faster R-CNN [9]. The processing of FWP, PAP, level-wise range-narrowing, and INT12 quantization causes 0.8, 0.3, 0.26, and 0.07 AP drop on average on the benchmarks, respectively. Compared to INT12 quantization, INT8 quantization is not adopted because it results in an average drop of 9.7 AP on the benchmarks, which is an unacceptable accuracy degradation. DEFA preserves a relatively high detection accuracy with negligible AP loss, which is 3.5-7.4 AP higher than Faster R-CNN. Figure 6 (b) shows the reduction ratio in sampling points, fmap pixels, and computation cost achieved by our pruning algorithms. FWP and PAP reduce 43% fmap pixels and 84% sampling points on average, and the computation cost on the unimportant fmap pixels and sampling points is also eliminated, accounting for more than 50% of the overall computation. For level-wise range-narrowing, we adjust bounded ranges of sampling offsets in each level to achieve a trade-off between accuracy and SRAM size.

5.3 Performance Gain from Our Hardware Optimization Tactics

5.3.1 Multi-scale Parallel Processing. Figure 7 (a) indicates the MSGS throughput improvement of inter-level parallel processing over intra-level parallel processing on the selected benchmarks. The bank conflicts are detected in intra-level parallel processing when plural sampling points need to access different addresses in the same SRAM bank in one clock cycle. In this case, extra clock cycles are spent on detecting bank conflicts, stopping the pipeline, and sequentially processing the requests. The proposed inter-level parallel processing completely avoids bank conflicts and achieves 3.06× higher MSGS throughput than the intra-level parallel processing on average under the same degree of parallelism.

5.3.2 Fine-grained Operator Fusion. As shown in Figure 7 (b), fine-grained operator fusion (op fusion) of MSGS and aggregation results in 73.3% and 15.9% energy saving on DRAM access and SRAM access of the overall MSGS energy consumption in memory access. Fine-grained operator fusion reduces the off-chip transfer of the BI result and utilizes the BI result directly to compute aggregation in the reconfigurable PE array in the BA mode, avoiding SRAM access. DEFA only adds 0.5% extra SRAM storage to support fine-grained operator fusion.

5.3.3 Fmap Reuse. Figure 7 (b) presents the energy saving of fmap reuse. Fmap reuse significantly reduces DRAM access of the fmap pixels in the overlapping bounded range, saving 88.2% of the total MSGS energy consumption in memory access. The writing operations to the SRAM of the repetitive fmap pixels fetched from the DRAM are also eliminated, which reduces 22.7% energy consumption of the overall MSGS energy consumption in memory access.

5.4 Comparisons with Other Platforms

We compare the speedup and the energy efficiency (EE) improvement of DEFA with the Nvidia GPUs in Figure 9. The performances of CPUs are not evaluated because MSDeformAttn only has a CUDA implementation. During the comparison with GPUs, we scale up DEFA to attain 13.3 TOPS and 40 TOPS peak throughput respectively, which matches Nvidia 2080Ti (13.5 TFLOPS@FP32) GPU and Nvidia 3090Ti (40 TFLOPS@FP32) GPU. DEFA achieves 10.1-31.9× speedup as well as 20.3-37.7× energy efficiency improvement

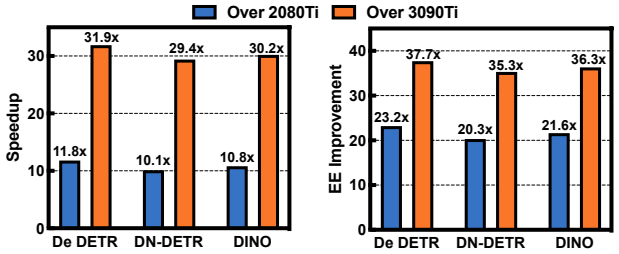

Figure 9: Speedup and energy Efficiency improvement of DEFA over GPUs.

over Nvidia 2080Ti GPU (250W) & Nvidia 3090Ti GPU (450W) on average. The high speedup results from the inter-level parallel processing without any bank conflicts in MSGS. DEFA further accelerates MSDeformAttn via FWP and PAP reducing the computation of redundant fmap pixels and sampling points. The energy saving is attributed to DRAM and SRAM access reduction through fine-grained operator fusion and fmap reuse. Figure 8 shows the area and energy breakdown of DEFA. The SRAM occupies the largest area since MSGS requires a large amount of on-chip memory to store multi-level fmaps. The DRAM access dominates the energy consumption due to the large data transfer in MM.

Table 1 compares DEFA and other attention ASIC platforms [11, 10, 12]. DEFA achieves 3.7×, 3.4× and 2.2× higher energy efficiency than ELSA [11], SpAtten [10] and BESAPU [12] for the following reasons. ELSA prefetches the key matrix and speculates candidates for every query token through orthogonal projection before attention computation. Its preprocessing needs to access the query and key matrix in SRAM numerous times and consumes large power, while the SRAM access of the pruning processing in DEFA takes less than 0.1% of the overall SRAM access and the pruning process in DEFA consumes extremely less power. SpAtten structurally removes the attention tokens and heads with low cumulative scores. Compared to the coarse-grained pruning methods in SpAtten, DEFA performs fine-grained FWP and PAP on fmap pixels and sampling points. Therefore, DEFA attains a higher pruning ratio with acceptable AP loss, which saves more computation and energy than SpAtten. BESAPU bidirectionally speculates and approximately computes weakly related tokens with an out-of-order scheduler to save energy consumption. However, the improvement of the energy efficiency resulting from the approximate computation highly depends on the ratio of weakly related tokens, which denotes the sparsity of attention, because the complex control logic consumes large energy. In contrast, fine-grained operator fusion and fmap reuse can save a large amount of memory access to enhance energy efficiency in MSGS without the restriction of sparsity. Besides, the speculation in BESAPU only reduces the negative operations of weakly related tokens, while FWP and PAP in DEFA can remove all the computation of the redundant fmap pixels and sampling points, thus saving more energy. Hence, DEFA achieves higher energy efficiency than BESAPU.

6 CONCLUSION

We propose DEFA, the first algorithm-architecture co-design for efficient MSDeformAttn acceleration. On the algorithm level, we present FWP and PAP to effectively prune the fmap pixels and sampling points in MSGS so that the memory access and computation are significantly reduced. On the architecture level, DEFA adopts inter-level parallel processing for throughput enhancement. DEFA further utilizes fine-grained operator fusion and fmap reuse to alleviate memory footprint. DEFA achieves 21.5×, 36.5× more energy saving than Nvidia RTX 2080Ti, 3090Ti and 3.7×, 3.4× and 2.2× higher energy efficiency than the baseline attention accelerators.

Table 1: Comparison with Other ASIC Platforms

	[11] ISCA'21	[10] HPCA'21	[12] JSSC'22	**DEFA**
Function	Attention			Deform Attn
Technology(nm)	40	40	28	40
Area(mm^2)	1.26	1.55	6.82	2.63
Frequency(MHz)	1000	1000	500	400
Precision	INT9	INT12	INT12	INT12
Power(mW)	969.4	294.0	272.8	99.8
Throughput(GOPS)	1088	360	522	418
Energy Effi.(GOPS/W)	1120	1224	1910	4187

ACKNOWLEDGMENTS

This work was supported by the National Natural Science Foundation of China under Grant 62074097.

REFERENCES

[1] Xizhou Zhu et al. 2021. Deformable DETR: Deformable Transformers for End-to-End Object Detection. In *ICLR*.
[2] Jifeng Dai et al. 2017. Deformable Convolutional Networks. In *ICCV*, 764–773.
[3] Nicolas Carion et al. 2020. End-to-End Object Detection with Transformers. In *ECCV*, 213–229.
[4] Feng Li et al. 2022. DN-DETR: Accelerate DETR Training by Introducing Query Denoising. In *CVPR*, 13619–13627.
[5] Hao Zhang et al. 2022. DINO: DETR with Improved DeNoising Anchor Boxes for End-to-End Object Detection. In *ICLR*.
[6] Zhiqi Li et al. 2022. BEVFormer: Learning Bird's-Eye-View Representation from Multi-Camera Images via Spatiotemporal Transformers. In *ECCV*, 1–18.
[7] Yiming Li et al. 2023. VoxFormer: Sparse Voxel Transformer for Camera-based 3D Semantic Scene Completion. In *CVPR*, 9087–9098.
[8] Yuanhui Huang et al. 2023. Tri-Perspective View for Vision-Based 3D Semantic Occupancy Prediction. In *CVPR*, 9223–9232.
[9] Shaoqing Ren et al. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *NeurIPS*, 28.
[10] Hanrui Wang et al. 2021. SpAtten: Efficient Sparse Attention Architecture with Cascade Token and Head Pruning. In *HPCA*, 97–110.
[11] Tae Jun Ham et al. 2021. ELSA: Hardware-software co-design for efficient, lightweight self-attention mechanism in neural networks. In *ISCA*, 692–705.
[12] Yang Wang et al. 2022. An Energy-Efficient Transformer Processor Exploiting Dynamic Weak Relevances in Global Attention. *IEEE JSSC*, 58, 1, 227–242.
[13] Qijing Huang et al. 2021. CoDeNet: Efficient Deployment of Input-Adaptive Object Detection on Embedded FPGAs. In *FPGA*, 206–216.
[14] Shan Li et al. 2022. A Computational-Efficient Deformable Convolution Network Accelerator via Hardware and Algorithm Co-Optimization. In *SiPS*, 1–6.
[15] Tsung-Yi Lin et al. 2014. Microsoft COCO: Common Objects in Context. In *ECCV*, 740–755.
[16] Naveen Muralimanohar, Rajeev Balasubramonian, and Norman P Jouppi. 2009. CACTI 6.0: A tool to model large caches. *HP laboratories*, 27, 28.
[17] Soroush Ghodrati et al. 2020. Bit-parallel vector composability for neural acceleration. In *DAC*, 1–6.