Bit-Trimmer: Ineffectual Bit-operation Removal for CIM Architecture

Yintao He[1,3], Shixin Zhao[2,3], Songyun Qu[2,3], Huawei Li[1,3,4], Xiaowei Li[1,3], Ying Wang[2,3]
SKLP, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China[1]
CICS, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China[2]
University of Chinese Academy of Sciences, Beijing, China[3], *Peng Cheng Laboratory, Shenzhen, China*[4]
{heyintao19z, zhaoshixin22s, qusongyun18z, lihuawei, lxw, wangying2009}@ict.ac.cn

***Abstract*—ReRAM-based accelerator of bit-slicing architecture is a promising solution to neural network inference, which allows ineffectual bit-operation removal for greater potential gains. However, existing techniques mostly exploit the removal of weight-associated ineffectual operations, which cannot eliminate the activation-induced ineffectual operations. Alternatively, some approaches adopt an isolated two-stage approach to remove at the weight and activation-level, which leaves a big proportion of ineffectual bit-level operations. Therefore, in contrast to all these coarse-grained operation removal techniques, it is challenging to jointly eliminate ineffectual bit-operation induced by either activation or weight bit-slices for ReRAM-based accelerators. This work presents a novel ineffectual bit-operation removal approach and the accompanied ReRAM-based bit-operation clipping architecture that skips all those bit-level operations that make negligible impacts on neural network outputs. In experiments, the proposed bit-operation clipping ReRAM accelerator, Bit-Trimmer, achieves 5.28× energy efficiency and 2.04× speedup on average. Besides, compared with two SOTA ReRAM accelerator designs with bit-operation removal, it outperforms by 1.56× and 1.88× energy efficiency.**

I. Introduction

Computing-in-memory (CiM) accelerator using emerging resistive random access memory (ReRAM), shows huge potential in the energy-efficient acceleration of deep neural networks (DNNs). With the low-power ReRAM crossbar (XB) structure, ReRAM-based accelerators have been considered as a promising solution in the scenarios of IoT and mobile computing systems. Previously, many ReRAM-based designs bring up two orders of magnitude energy efficiency compared with CMOS-based designs [1]–[3].

Typical ReRAM-based XB structure conducts bit-slicing technique to represent high-precision input feature and weights [1], as shown in Fig. 1. It brings a new opportunity to prune bit-level ineffectual operations for great potential gains. For instance, the performance improvement potential of removing ineffectual bit-operation is 42.8× on average than the word-level operation elimination [4]. Recently, many ReRAM-based designs pursue a performance boost by eliminating the ineffectual bit-operations in weight bit-slices terms [5], [6]. PIM-Prune [5] compresses the model parameters into the specified number of XB. ASBP [6] adopts reinforcement learning (RL) based search algorithm saving 79% energy consumption.

However, there is still a lack of ReRAM-based design approaches to exploit ineffectual bit-operation removal in NNs and unleash the performance potential at the co-exploration of

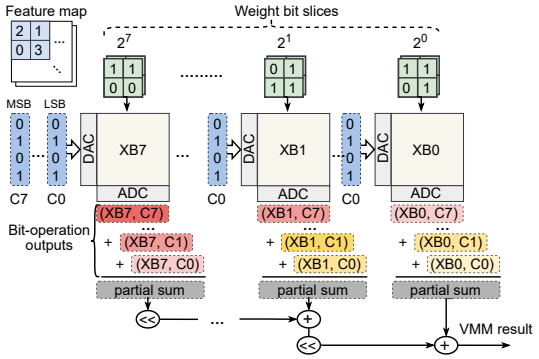


Fig. 1. The vector-matrix multiplication (VMM) process of CIM architecture with bit-slicing technique.

weight and activation bit-slices. According to the research on traditional architectures, optimizing both activation and weight can result in ∼10x speedup potential compared to weight skip alone [4]. In the previous methods of pruning weight bit-slice, it is unrealistic to search both A and W at the same time due to the expanding search space. For example, the search space size for a ResNet-18 network structure with 8-bit weights and activations (W8A8) will expand from 8^{18} to 8^{36}, thus the search algorithm is hard to converge.

Since the approach considering simultaneous weight and activation is hard to conduct, Multi-Precision [7] proposes a two-stage approach that eliminates bit-operations in the terms of weight and activation bit-slices in sequence to achieve a higher efficacy of ineffectual bit-operation removal, as shown in Fig. 2(b). However, the two-stage separated search approach will lead to a sub-optimal removal result caused by the limited number of exploitable ineffectual bit-operations.

Prior studies are mostly focused on the bit-slice of input, i.e. activation or weight, which cannot fully cover the ineffectual bit-operations. These approaches either remove weight-associated ineffectual operations, which cannot eliminate the activation-induced ineffectual operations, or rely on an isolated two-stage approach to remove at the weight or the activation level, which misses a big proportion of ineffectual bit-level operations. Therefore, it is challenging to eliminate ineffectual bit-operation in activation and weight terms simultaneously for ReRAM-based accelerators.

We observe that bit-operation output results naturally reveal the importance of each bit-operation. Different bit-operation make vastly different contributions to the final computational

979-8-3503-4859-0/DATE24/© 2024 EDAA

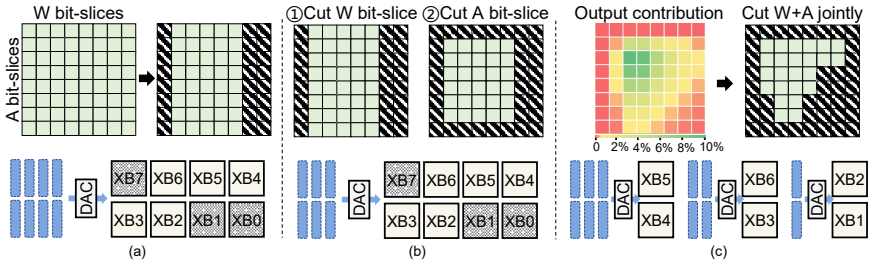


Fig. 2. Three ineffectual bit-operation removal approaches. (a) Only in weight bit-slices term without activation-induced ineffectual operations elimination. (b) In the terms of weight and activation bit-slices in sequence. (c) The proposed removal approach in the terms of weight and activation bit-slices jointly.

results, but have the same computing cost. As shown in the left Fig. 2(c), the output absolute value of the bit-operations in the red regions only accounts for 3.4% of the total output results, but brings in 46.88% computation overhead. It is because different bit-operations have different output significance [8]. Take a W8A8 network as an example, the bit-operation output value with the most significant bit (MSB) of weight and activation is 16128× the LSB's one. Hence, the bit-operations contributing to less significant output values can be seen as ineffectual bit-operations.

Based on these important observations, we for the first time propose a ReRAM-based bit-operation clipping accelerator architecture, Bit-Trimmer. The proposed approach allows to remove the ineffectual bit-operations associated with the weight and activation bit-slices jointly, which provide opportunities to eliminate more bit-operations as shown in Fig. 2(c). With the bit-operation removal technique and the architecture design, we achieve both energy saving and performance speedup by eliminating inefficient computation to the highest degree. Specifically, our contributions are as follows:

- We propose a ReRAM-based bit-operation clipping accelerator architecture, Bit-Trimmer, to eliminate ineffective bit-operations at a negligible implementation overhead.
- We propose to remove different proportions and positions of ineffectual bit-operations for each layer and introduce a novel approach to find the most rewarding bit-level computing redundancy.
- We evaluate the network accuracy, energy consumption, and performance with Bit-Trimmer. The results show that compared with the baseline, our system achieves 5.28× energy efficiency and 2.04× speedup on average.

II. Background

A. Ineffectual Bit-Operations in CIM Architecture

ReRAM crossbar has shown great potential as an efficient in-situ vector-matrix multiplier. The matrix is programmed as the resistances of ReRAM cells. Due to the low precision of the ReRAM cell, many works conduct the bit-slicing technique to execute high-precision VMM operation [6], [9], [10]. Fig. 1 illustrates an 8-bit VMM calculation. Each XB on the ReRAM-based accelerator holds a unique bit-slice with different significance of a weight block (WB). For example, the least significant bit-slice of the WB resides on XB0. And the input bit-slices are fed into the XBs as bit-streams from C0 to C7. Assuming the DAC resolution and cell bit is 1-bit, one 8-bit VMM calculation requires 64 bit-operations.

Here we group bit-operations of each layer in the neural network, and a group of bit-operations is called a BOG. Each BOG is labeled (XBi, Cj), where i is the weight significant bit of the bit-operation, and j is the bit of activation significant bit. For example, The BOG (XB0, C0) includes the bit-operation with the least significant bit-slices of weight and activation. The number of BOGs for a layer can be calculated as below:

$$\#BOG = \frac{weight\_bit}{cell\_bit} \times \frac{activation\_bit}{DAC\_resolution} \quad (1)$$

As Fig. 2(c) shows, since the output significance of bit-operations from different BOGs varies greatly, it enables a new opportunity to exploit bit-level ineffectual computing.

B. ReRAM-based Ineffectual Bit-Operation Removal Design

To boost performance and energy efficiency, existing studies have proposed several bit-operation removal methods for ReRAM-based DNN accelerator designs in different terms, as shown in Fig. 2. The conservative bit-operation elimination methods have achieved obvious energy efficiency improvement. PIM-Prune [5] crossbar-wise prunes the ineffectual bit-operations and achieves a significant compression of weight matrices. ASBP [6] conducts an automatic model compression method at the granularity of crossbar size. However, those designs only focus solely on the ineffectual bit-operations induced by weight bit-slice, which leaves the activation-induced one. It is challenging to remove ineffectual bit-operations in terms of both weight and activation jointly. On the one hand, the values of activation bit-slices are dynamically decided by the input feature but weight parameters are static. On the other hand, it is difficult to converge using RL-based search algorithm or evolutionary algorithm in the joint space, since the search space size will expand for L^n times, where L is layer number and n is activation bit-width.

An intuitive approach to exploit ineffectual operations in terms of weight and activation bit-slices is in sequence, as Fig. 2(b) shows. Zhu et al. [7] proposed a precision configurable computing framework with a ReRAM-aware quantization method, and compressed ReRAM network in the order of weight and then activation, which achieves 75% energy saving on average. However, the disjointed search stages may lead to a sub-optimal removal result. The reason is that weight slices and the corresponding activation bit-slices are not considered together and it misses lots of potential ineffectual bit-operations. Therefore, we propose a joint bit-operation clipping method that integrates activation and weight bit-slice terms to quarry more ineffectual bit-operation in Fig. 2(c).

III. Approach

The Bit-Trimmer workflow consists of two stages: offline profiling and online inference, as depicted in Fig. 3. The offline profiling is further divided into two parts: 1. output contribution based removal approach (OCRA). This approach is used to clip ineffectual bit-operations. It analyzes the contribution of

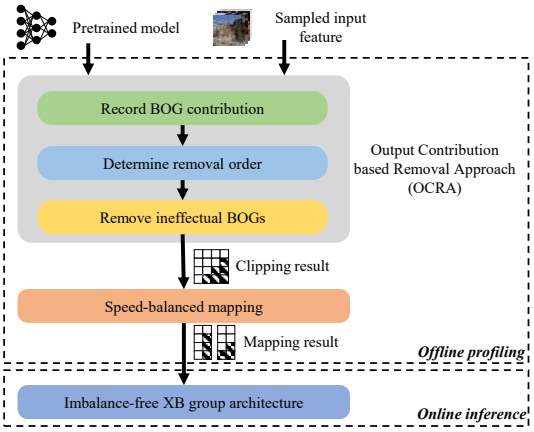


Fig. 3. The workflow of Bit-Trimmer.

each bit-operation to the final output and removes those that have negligible impact. 2. Speed-balanced mapping scheme. This scheme is employed to profile the clipped model and map it onto the proposed imbalanced-free XB group architecture for online inference. The details of this mapping scheme and the architecture design will be discussed in Section IV.

The goal of OCRA is to determine the effectual bit-operations of each layer and eliminate the ineffectual ones for efficiency boost. Assuming the size of the output feature map for Layer i is o_i and the mapped crossbar number C_i, the total number of bit-operations is:

$$\#BOPs = \frac{C_i}{(weight\_bit/cell\_bit)} \times o_i^2 \times \#BOG \qquad (2)$$

OCRA digs and removes the insignificant BOGs to reduce $\#BOG$, so that the whole computation cost will be decreased. The whole process consists of three steps below:

1) BOG Contribution Recording: First, we measure the output contribution of each BOG layer-by-layer. Take Layer 1 in Fig. 4(a) as an example, we cut the BOG from (XB3, C3) to (XB0, C0) and inference the clipped model to record the output absolute value sum of each BOG, i.e. output contribution, which displays the significance of each BOG. According to the smaller-norm-less-important assumption, the BOG with a larger output absolute value sum is more important.

2) BOG Sorting: Then, we sort the BOGs by the recorded output contribution. As Fig. 4(b) shows, the BOG (XB0, C0) in Layer 1 has the smallest output contribution and the output of the BOG (XB3, C2) contributes the most value of the full-precision result. Since we prioritize removing bit-operations with less significance, we sort the BOGs for each layer in ascending order, which is the removal order in Step 3.

3) Iterative Clipping: Once the sorting for each layer finishes, the iterative clipping starts. The iterative clipping approach defines which BOG will be cut out, which follows the two-tier cyclic order of BOG, layer. In the BOG-level loop, we iterate over each BOG in order to determine whether it is eliminated. And in each iteration, We introduce a parameter named target accuracy drop and layer by layer decide if the corresponding BOG can be clipped or not. Fig. 4(c) illustrates an example of ineffectual bit-operation removal process for a CNN with three layers. The target accuracy drop is set to 0.5% in this case. In Iteration 1, we start to determine whether clip (XB0, C0), i.e. the first BOG in Layer 1. We examine the validation accuracy drop when clipping (XB0, CO) is 0.01%, which is smaller than 0.5%. Thus, we clip the first BOG in Layer 1. After the first BOG in the order of each layer is judged, we move forward to Iteration 2. Until the whole BOGs for each layer have been iterated, OCRA stops and returns the clipping results of the neural network. In this case, it takes 16 iterations to traverse all the BOGs. Note that we did not choose layer-wise preference for removal, since it may causes too much clipping of the preceding layers and less removal of the middle layers with more computational operations.

After the OCRA process, the clipping result for each layer is determined and stored in a dictionary strcuture. Specifically, the clipping result for Layer N is represented as $Clip\_N = \{XBi\text{: }Abitflagi\}$, which is a dictionary with key-value pairs. The key XBi represents the weight significant bit of the bit-operation, and the corresponding value $Abitflagi$ represents each bit of the activation bit-slice, indicating whether it is preserved or not. Take XB5 in Fig. 5(a) as an example, if a BOG in XB5 is clipped, the corresponding value in the $Abitflag$ list is set to 0. Otherwise, if the BOG remains, the bit flag is 1. Thus, $Abitflag5$ is recorded as [0, 1, 0, 0, 0, 0].

IV. Architecture

OCRA eliminates the ineffectual bit-level operations through lightweight offline profiling. However, it is challenging for the conventional architecture design to leverage the bit-operation removal to boost performance speedup. In the conventional architecture, the analog-to-digital converters (ADCs) are sharing with the bitlines in the same XB [1]. Therefore, the computational latency of multiple parallel XBs in the conventional architecture is limited by the slowest XB with the most activation times. Take XB0-5 with different activation times as an example, in Fig. 5(b), XB0-3 and 5 have to wait for the calculation on XB4 to complete. And one VMM operation requires 6 cycles. The different activation times of XBs result in the temporal under-utilization of the ADCs.

To address this issue, the weight replication technique has been commonly employed to alleviate the XB-level imbalance [1], [3]. Weight replication involves duplicating weights in proportion to the activation time, thereby balancing the activation times required for each XB and minimizing the ADC underuti-

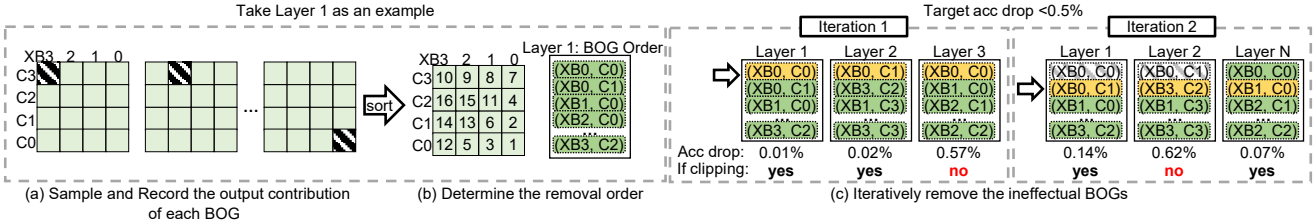
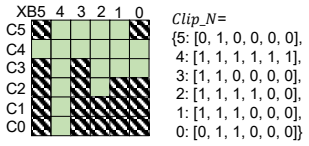


Fig. 4. The process of the output contribution based ineffectual bit-operation removal approach in Bit-Trimmer.



(a) Clipping Results for Layer N

XB ID	Activation Time	Computing Latency
XB0	2	$2t_{ADC}$
XB1	3	$3t_{ADC}$
XB2	4	$4t_{ADC}$
XB3	2	$2t_{ADC}$
XB4	6	$6t_{ADC}$
XB5	1	$1t_{ADC}$

(b) Conventional Mapping Scheme: $6t_{ADC}$

XB ID	Activation Time	Physical XBG ID	Computing Latency
XB5	1	0	$3.5t_{ADC}$
XB4	6	0	
XB0	2	1	$3t_{ADC}$
XB2	4	1	
XB3	2	2	$2.5t_{ADC}$
XB1	3	2	

(c) Speed-Balanced Mapping Scheme: $3.5t_{ADC}$

Fig. 5. (a) A case of clipping results for Layer N. (b) The conventional mapping result. (c) The speed-balanced mapping result of Bit-Trimmer.

lization. However, weight replication employs additional XBs to store weight parameters, resulting in increased ReRAM XB requirements and area overhead. For example, for Layer N depicted in Fig. 5(a), the weight replication scheme requires three times the original number of XBs. It motivates us to design a novel imbalance-free crossbar-based architecture to eliminate the issue without extra XB requirements.

A. Insight from the Real Distribution of Clipped Bit-Operation

To gain insights into imbalanced bit-operation removal, we analyze the ReRAM workload by examining the distribution of ineffectual bit-operation. Fig. 5(a) illustrates the elimination of redundant bit-operations, where green represents effectual bit-operations and grey indicates eliminated redundancies. We find that the XBs hold different weight bit-slices are activated unevenly. For instance, while XB5 is activated for only one cycle, XB4 requires 6 cycles for activation.

In the conventional architecture and mapping scheme, the imbalanced number of crossbar activation times within the XBs leads to inefficiencies. For example, XB5 and XB3-0, which have shorter activation bit-streams, have to wait for the slowest XB4 due to the different activation times of different XBs. Thus, there is idle time for the CIM components within XB5 and 3-0, particularly the inactivity of ADC, which serves as the computing bottleneck in the CIM architecture [11].

Considering the expensive area overhead of ADC, multiple bitlines in existing architectures share a single ADC [1]. The results of VMM calculations performed by O(1) stored in sample-and-hold (S+H) unit wait for the ADC bitline-by-bitline conversion. Thus, the sampling frequency and the number of ADC per XB decide the computing latency of one XB [12]. Assuming the ADC number per XB is N_{ADC} and input activation for XBi has m_i bit-slices, the computing latency of XBi to complete a VMM calculation is below:

$$T_{XBi} = \frac{m_i}{N_{ADC}} \times t_{ADC} \tag{3}$$

where t_{ADC} represents the latency that one ADC samples all bitlines on an XB. We assume that one ADC one XB, then the total latency of the XB5-0 in the conventional ReRAM architecture can be calculated as in Eq. 4, which is equal to the computing latency of the slowest XB4:

$$T = max(T_{XB0}, T_{XB2}, ..., T_{XB5}) = 6t_{ADC} \tag{4}$$

The computing latency of the example with conventional mapping scheme is $6t_{ADC}$ and the ADC utilization is only 50%. Therefore, OCRA can only reduce the computational energy consumption without speedup.

B. Speed-Balanced Mapping Scheme

To further leverage the removal of bit-operations for performance speedup, we propose a design that combines a computation imbalance-free XB group (XBG) with a speed-balanced mapping scheme. The XBG allows to map the

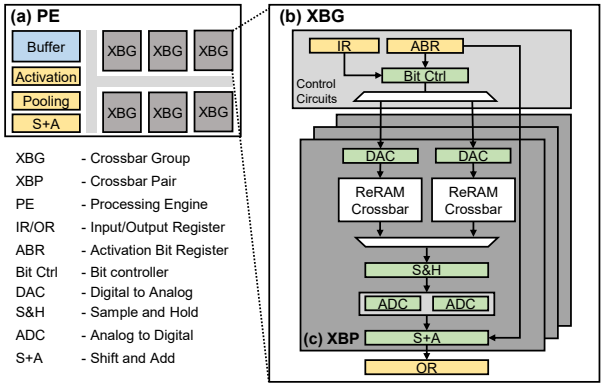


Fig. 6. The proposed Bit-Trimmer architecture design.

weight parameters from two different XBs with different bit significances, denoted as XBi and XBj. This design aims to balance the computing latency between the mapped XBi and XBj, and mitigate the issue of idle cycles resulting from imbalanced activation times. We first introduce the speed-balance mapping scheme, which maps the clipped neural network onto the XBGs. Subsequently, we provide the details of XBG design in the upcoming subsection. By implementing the accompanied mapping scheme for the imbalance-free XBG architecture, the computing speed can be increased due to higher ADC utilization.

The proposed speed-balanced mapping scheme aims to address the significant differences in activation time by pairing them and map them onto the same physical XBG. This ensures a more balanced distribution of activation times among multiple XBGs. The mapping schme executes layer by layer and Fig. 5(c) provides an example to illustrate the process for a layer. The six XBs are mapped to three XBGs: (XB5, XB4), (XB0, XB2), and (XB3, XB1) respectively. Since the two XBs in each XBG share ADCs, we can calculate the computing latency of XBG0 (consisting of XB5 and XB4) as follows:

$$T'_{XBG0} = \frac{m_5 + m_4}{2N_{ADC}} \times t_{ADC} \tag{5}$$

And the total latency of the six XBs can be calculated by:

$$T' = max(T'_{XBG0}, T'_{XBG1}, T'_{XBG2}) = 3.5t_{ADC} \tag{6}$$

Consequently, the computing latency through the speed-balanced mapping scheme is $3.5t_{ADC}$ in the presented case. Besides, the ADC utilization improves from 50% to 85.7%.

C. Architecture Overview

We propose Bit-Trimmer ReRAM accelerator architecture for CNN to support the implementation of the speed-balanced mapping scheme. The Bit-Trimmer architecture requires both bit-operation removal inference and ADC sharing between crossbars within the same XBG. This architecture requires minimal extra components, consisting only of a controller and a small flag register to support the scheme. Fig. 6 illustrates the Bit-Trimmer architecture. It comprises multiple processing

engines (PEs), where each PE includes a local buffer, XBGs, and the function circuits such as the activation unit.

The imbalance-free XBG, depicted in Fig. 6(b), comprises components such as control circuits, XB pairs (XBPs), and an output register. The control circuits play a crucial role and are divided into three parts: the activation bit flag register (ABR), the bit controller, and the input register. ABR is responsible for storing the clipped results for XBi and XBj in the XBG (i.e. the significance of the weight bit-slices, the activation clipping results $Abitflagi$ and $Abitflagj$). The required size of the ABR register is determined by the number of activation bit-slices and weight bit-slices, calculated as ((#activation bit-slice)×2 + #weight bit-slice) bits. For instance, when there are 6 activation bit-slices and 6 weight bit-slices, the ABR register size would be 18 bits, resulting in a small index overhead. Besides, as shown in Fig. 6(c), the XBs in one XBG are paired together to form XBPs. The two XBs in one XBP share two ADCs, allowing for efficient resource utilization.

Fig. 6(b) shows the dataflow within an imbalanced-free XBG in the proposed ReRAM-based DNN accelerator. In this configuration, the two XBs are activated sequentially due to the differing activation bits for each XB. Firstly, we retrieve the clipping result of the activation bit-slice $Abitflag$ from the ABR and transmit it to the bit controller. Simultaneously, IR receives the activation bit-slice data. The bit controller evaluates whether to send the bit-slice to the DAC according to $Abitflag$ of the corresponding XB. Once the XB is activated, the computing currents are accumulated to perform the analog bit-operation, and the bit controller issues a signal to open the multiplexer, allowing the analog output to be directed towards the ADCs for conversion. Then, the analog outputs are sampled and held to wait for the ADCs to generate digital outputs. To ensure computational correctness, the bit controller generates the shift time and sends the signal to the S+A unit. Finally, the digital output is stored in the output register.

V. Evaluation

A. Experiment Setup

1) Workloads: We applied the proposed bit-operation removal technique to six classic image classification network models, including Svhn on SVHN, Convnet, VGG-7, ResNet-18 on CIFAR10, and ResNet-34, VGG-16 on ImageNet dataset. All the models are trained with 8-bit weight parameters and 8-bit activations in Pytorch. After the removal process, we retrain the network with the same epochs for accuracy recovery on both baseline and Bit-Trimmer designs.

2) The Configuration of Bit-Trimmer: To evaluate the inference accuracy and performance of our accelerator, we run several pretrained models on this framework with the proposed ineffectual bit-operation removal technique in Bit-Trimmer. We adopt the ReRAM device model from [13] and the crossbar size is 128×128. The ReRAM cell is single-level bit to protect the computing precision, and the HRS/LRS are 100KΩ/10KΩ. We simulate the SRAM buffer using CACTI simulator [14] with 32nm technology. We use other crossbar peripheral circuit models in 32nm technology node from ISAAC [1], including digital-to-analog converter (DAC), ADC, Shift and Add Unit (S+A), input register, output register, etc.

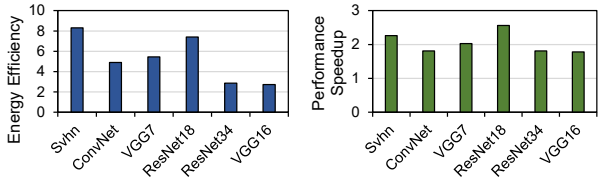


Fig. 7. The energy efficiency and performance speedup of Bit-Trimmer over the baseline.

We adopt ISAAC-like architecture as the baseline, in which each XB has a unique ADC without inter-XB sharing. Besides, we compare the performance improvement of Bit-Trimmer with two state-of-the-art ReRAM-based accelerators [6], [7] with different ineffectual bit-operation removal approaches.

B. Experimental Result

1) Overall Performance: We evaluate the Bit-Trimmer approach on the aforementioned datasets. Table I shows the bit-operation reduction and the variation of network inference accuracy. Specifically, we set the target accuracy drop to 0.5% for Svhn and ConvNet, 5% for VGG-7, ResNet-34 and VGG-16, and 30% for ResNet-18 respectively, according to the model size. Compared with ISAAC-like that executes all bit-operations for neural network inference, our design achieves 77.8% bit-operation reduction on average for the six benchmarks. Furthermore, we compared the results of different models on the same dataset. And we observed that larger-scale model architectures with the same dataset, such as VGG-7 and ResNet-18 with CIFAR-10, exhibit higher BOP reduction and suffer less accuracy loss after retraining, despite setting a higher target accuracy drop compared to smaller models (Convnet). This suggests that the larger models have a higher capacity to recover from the pruning process and retain their accuracy even with a significant BOP reduction.

We evaluate the total energy efficiency and the performance speedup under different workloads. As Fig. 7 shows, Bit-Trimmer achieves 5.28× energy efficiency and 2.04× performance speedup over the baseline on average.

Then, we present a decomposition study to demonstrate the effect of the proposed imbalanced-free XBG architecture. Fig. 8 illustrates the performance speedup of OCRA with the traditional ISAAC-like architecture and OCRA the with XBG architecture compared to the baseline. Compared to OCRA with ISAAC-like, the introduction of XBG provides an additional speedup of 0.34× on average over the baseline. This indicates that the imbalance-free XBG architecture enhances the acceleration effect of OCRA by mitigating the imbalance in inter-XB activation times.

TABLE I
The bit-operation reduction and inference accuracy of Bit-Trimmer.

Dataset	Network	BOP Reduction	Accuracy Loss
SVHN	Svhn	88.79%	0.29%
CIFAR10	Convnet	80.54%	0.28%
	VGG-7	82.08%	0.27%
	ResNet-18	86.72%	0.08%
ImageNet	ResNet-34	65.25%	1.25%
	VGG-16	63.53%	1.56%

BOP: the abbreviation for bit-operation.

TABLE II
COMPARISON STUDY ON RESNET-18 WITH RELATED BIT-OPERATION REMOVAL APPROACHES.

	BOP Reduction	Accuracy Loss	Energy Efficiency	Performance Speedup
ASBP [6]	82.89%	0.09%	4.76	1.00
Multi-Precision [7]	74.88%	0.12%	3.95	2.11
Bit-Trimmer	**86.72%**	**0.08%**	**7.41**	**2.56**

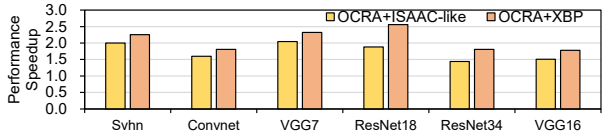


Fig. 8. Comparison study of speedup with OCRA+ISAAC-like/OCRA+XBG.

2) Comparisons of Two SOTA Designs: We compare Bit-Trimmer with the related bit-operation removal methods for ReRAM accelerators, ASBP [6] and Multi-Precision [7], which are representative state-of-the-art ReRAM designs. For ineffectual bit-operation removal, ASBP adopts the RL-based pruning method in terms of weight bit-slice. And Multi-Precision eliminates the redundant bit-operations by quantifying weight and activation sequentially according to the set accuracy drop. However, both of them do not consider bit-operation removal in terms of weight and activation jointly. Since the configuration of DAC resolution in Multi-Precision is different from ASBP and Bit-Trimmer, we re-implemented the removal approach of Multi-Precision with the same target accuracy drop and hardware parameter configuration for a fair comparison. The comparison results for ResNet-18 on CIFAR-10 are shown in Table II. Compared with ASBP, Bit-Trimmer achieves 1.56× energy efficiency and 2.56× speedup. Compared with Multi-Precision, Bit-Trimmer enhances 1.88× energy efficiency and 1.21× speedup.

3) Sensitivity on the Crossbar Size: We performed experiments on ResNet-18 to investigate the impact of XB size on our design. Table III presents the results of BOP reduction, accuracy loss, and index size for four different XB sizes: 128, 64, 32, and 16. The index size refers to the storage requirement of the clipping result stored in ABR. The experimental results indicate that as the XB size decreases, there is a slight improvement in BOP reduction without accuracy loss. Besides, the index size increases with the smaller XB size. The results also demonstrate that our method can be conducted in the ReRAM-based design with small operating units (OU) [15].

4) Sensitivity on the Target Accuracy Drop: Finally, We explore the effect of target accuracy drop setting on our Bit-Trimmer. Fig. 9 shows the bit-operation reduction and inference accuracy with different configurations of the target accuracy drop for VGG-7 and ResNet-18 on CIFAR-10. We change the value of the target accuracy drop from 0.5% to 30%. It can be seen that the bit-operation reduction is growing along with the increase of the setting of target accuracy drop. However, the inference accuracy loss increases as well, since the effect of network fine-tuning to recover accuracy is limited.

VI. CONCLUSION

In this paper, we investigated the design towards ineffectual bit-operation removal in terms of activation and weight bit-slice jointly for CIM architecture, which reduces the operating time in crossbars for low-power IoT and edge devices. With the bit-level computing redundancy removal approach and the bit-operation clipping ReRAM accelerator architecture, the overall computational effort can be reduced. Compared with the baseline, Bit-Trimmer achieves 5.28× energy efficiency and 2.04× speedup with negligible accuracy loss.

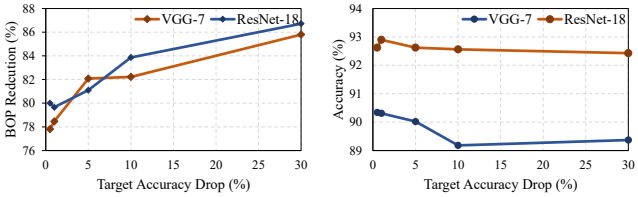


Fig. 9. The effect of target accuracy drop setting on bit-operation reduction and inference accuracy.

TABLE III
THE SENSITIVITY ON THE CROSSBAR SIZE.

XB Size	BOP Reduction	Accuracy Loss	Index Size
128	86.72%	0.08%	2.79KB
64	86.81%	0.11%	10.90KB
32	87.29%	0.04%	43.59KB
16	87.30%	0.09%	174.38KB

ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China (NSFC) under grant No.(62222411, 92373206, 62090024). The corresponding authors are Ying Wang and Huawei Li.

REFERENCES

[1]A. Shafiee *et al.*, "Isaac: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars," in *ISCA*, 2016.
[2]P. Chi *et al.*, "Prime: A novel processing-in-memory architecture for neural network computation in reram-based main memory," in *ISCA*, 2016.
[3]L. Song *et al.*, "Pipelayer: A pipelined reram-based accelerator for deep learning," in *HPCA*, 2017.
[4]S. Sharify *et al.*, "Pruning filters for efficient convnets," *arXiv preprint arXiv:1805.04513*, 2018.
[5]C. Chu *et al.*, "Pim-prune: Fine-grain dcnn pruning for crossbar-based process-in-memory architecture," in *DAC*, 2020.
[6]S. Qu *et al.*, "Asbp: Automatic structured bit-pruning for rram-based nn accelerator," in *DAC*, 2021.
[7]Z. Zhu *et al.*, "A configurable multi-precision cnn computing framework based on single bit rram," in *DAC*, 2019.
[8]B. Feinberg *et al.*, "Enabling scientific computing on memristive accelerators," in *ISCA*, 2018.
[9]F. Liu, W. Zhao *et al.*, "Sme: Reram-based sparse-multiplication-engine to squeeze-out bit sparsity of neural network," in *ICCD*, 2021.
[10]Y. He *et al.*, "Infox: An energy-efficient reram accelerator design with information-lossless low-bit adcs," in *DAC*, 2022.
[11]W. Li *et al.*, "Timely: Pushing data movements and interfaces in pim accelerators towards local and in time domain," in *ISCA*, 2020.
[12]Q. Zheng *et al.*, "Mobilattice: A depth-wise dcnn accelerator with hybrid digital/analog nonvolatile processing-in-memory block," in *ICCAD*.
[13]M.-F. Chang *et al.*, "19.4 embedded 1mb reram in 28nm cmos with 0.27-to-1v read using swing-sample-and-couple sense amplifier and self-boost-write-termination scheme," in *ISSCC*, 2014.
[14]N. Muralimanohar *et al.*, "Cacti 6.0: A tool to model large caches," *HP laboratories*, 2009.
[15]Y. Zhang *et al.*, "A practical highly paralleled reram-based dnn accelerator by reusing weight pattern repetitions," *IEEE TCAD*, 2022.