XBM: A Crossbar Column-wise Binary Mask Learning Method for Efficient Multiple Task Adaption

Fan Zhang[1], Li Yang, Jian Meng, Yu (Kevin) Cao, Jae-sun Seo, Deliang Fan[2]

Email: 1. fzhang95@asu.edu 2. dfan@asu.edu

Lab : https://dfan.engineering.asu.edu

School of Electrical, Computer and Energy Engineering,
Arizona State University, Tempe, AZ

Ira A. Fulton Schools of Engineering
Arizona State University

CONTENTS

Background

What is Multi-task adaption and In-Memory computing

Algorithm

How to efficiently apply multi-task adaption on ReRAM crossbar accelerator

Hardware

ReRAM crossbar accelerator design

Experiment

Performance of XBM
Power
Area
Latency

01 Background

- ✓ Deep Learning Applications
- ✓ Multitask Adaption
- ✓ ReRAM-based DNN Accelerator

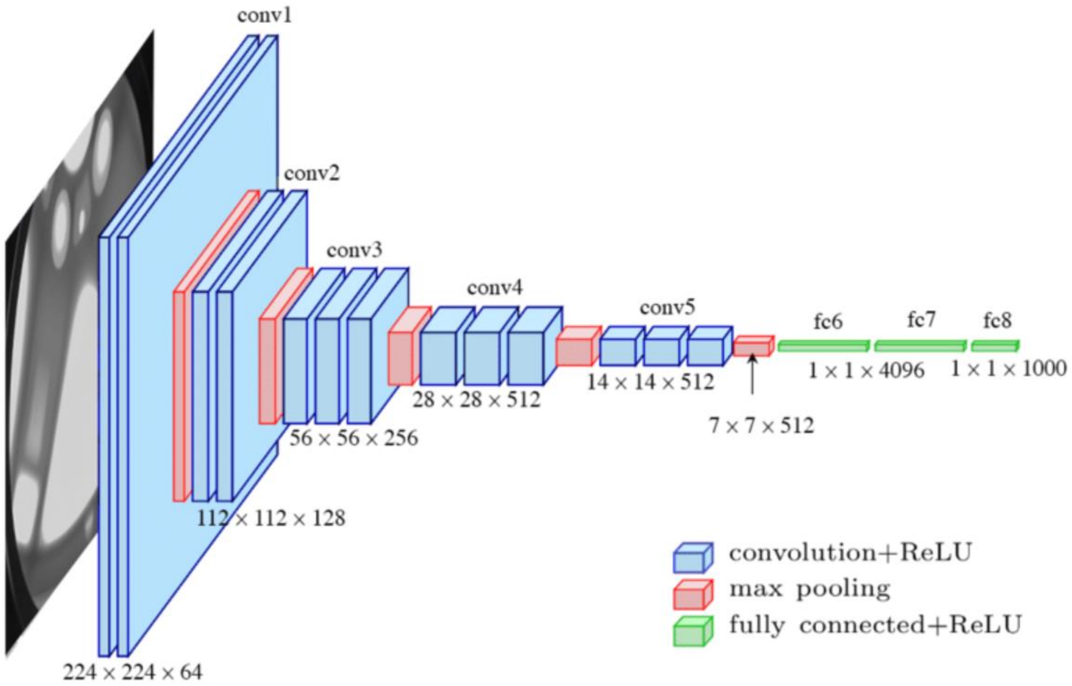
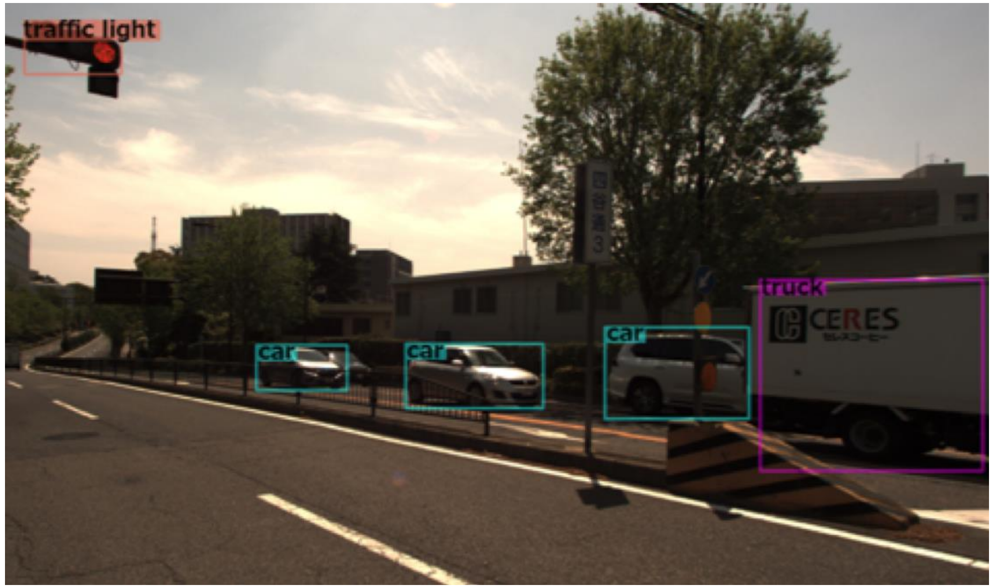
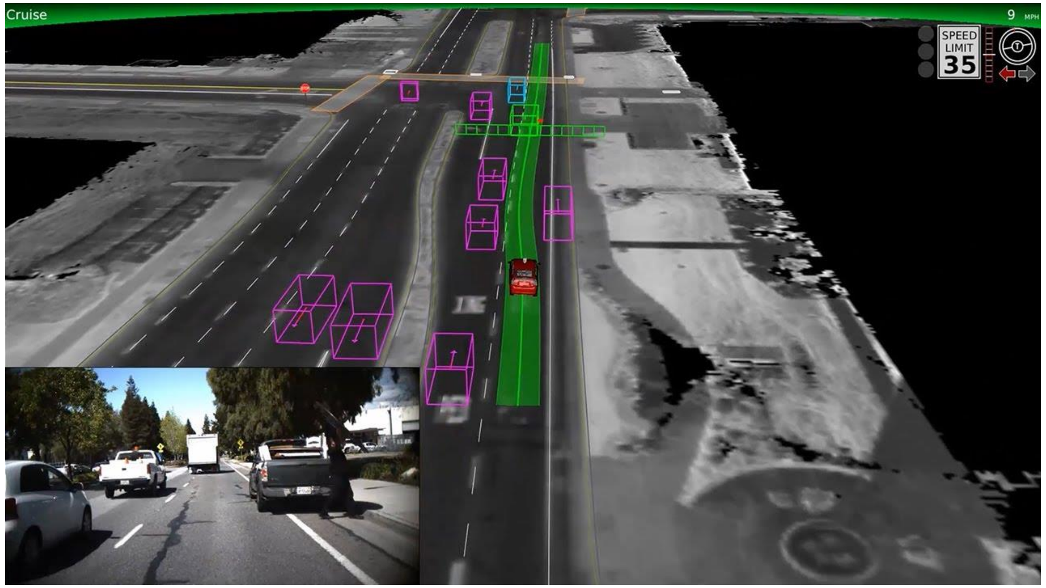


Image Classification

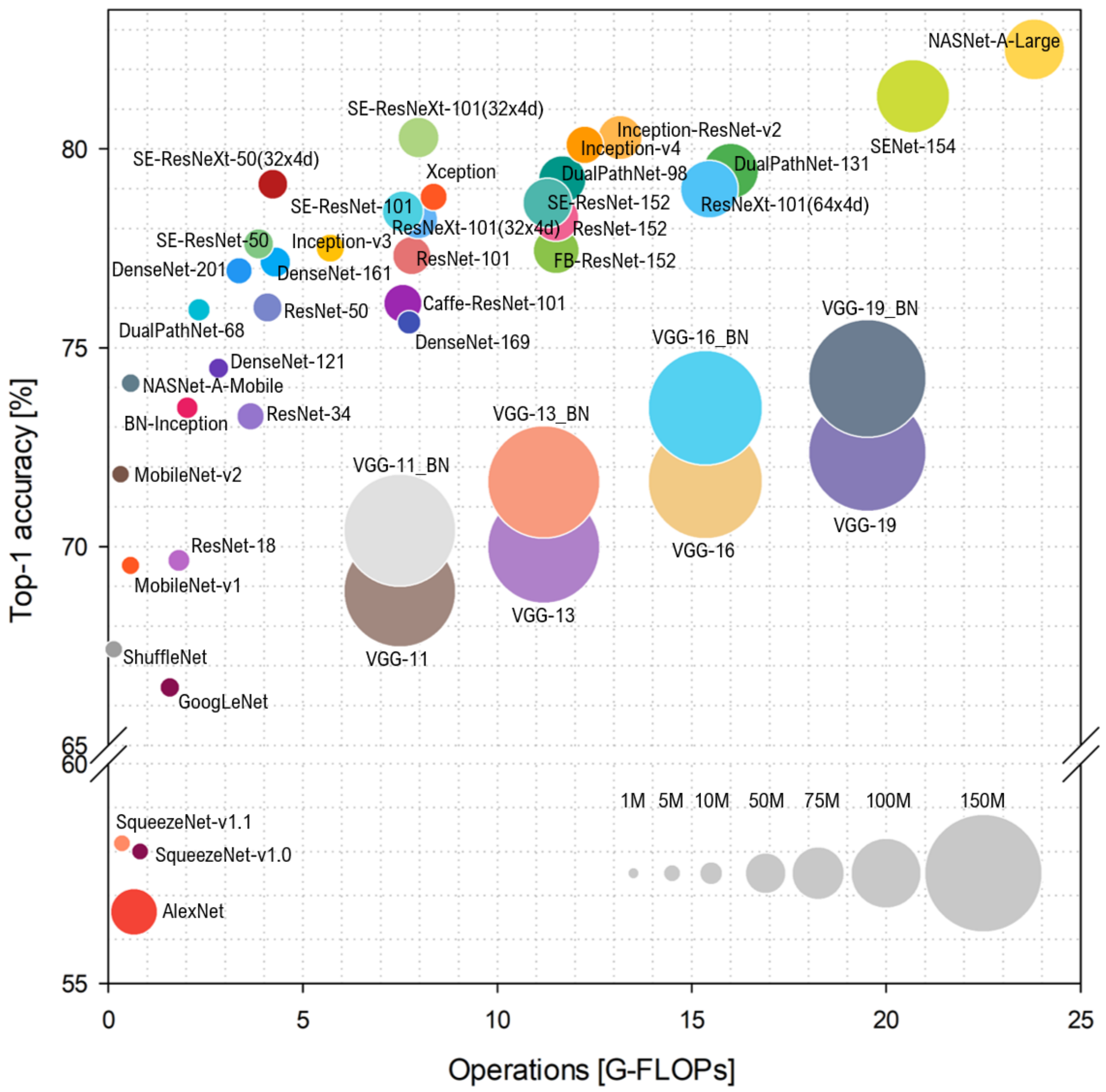


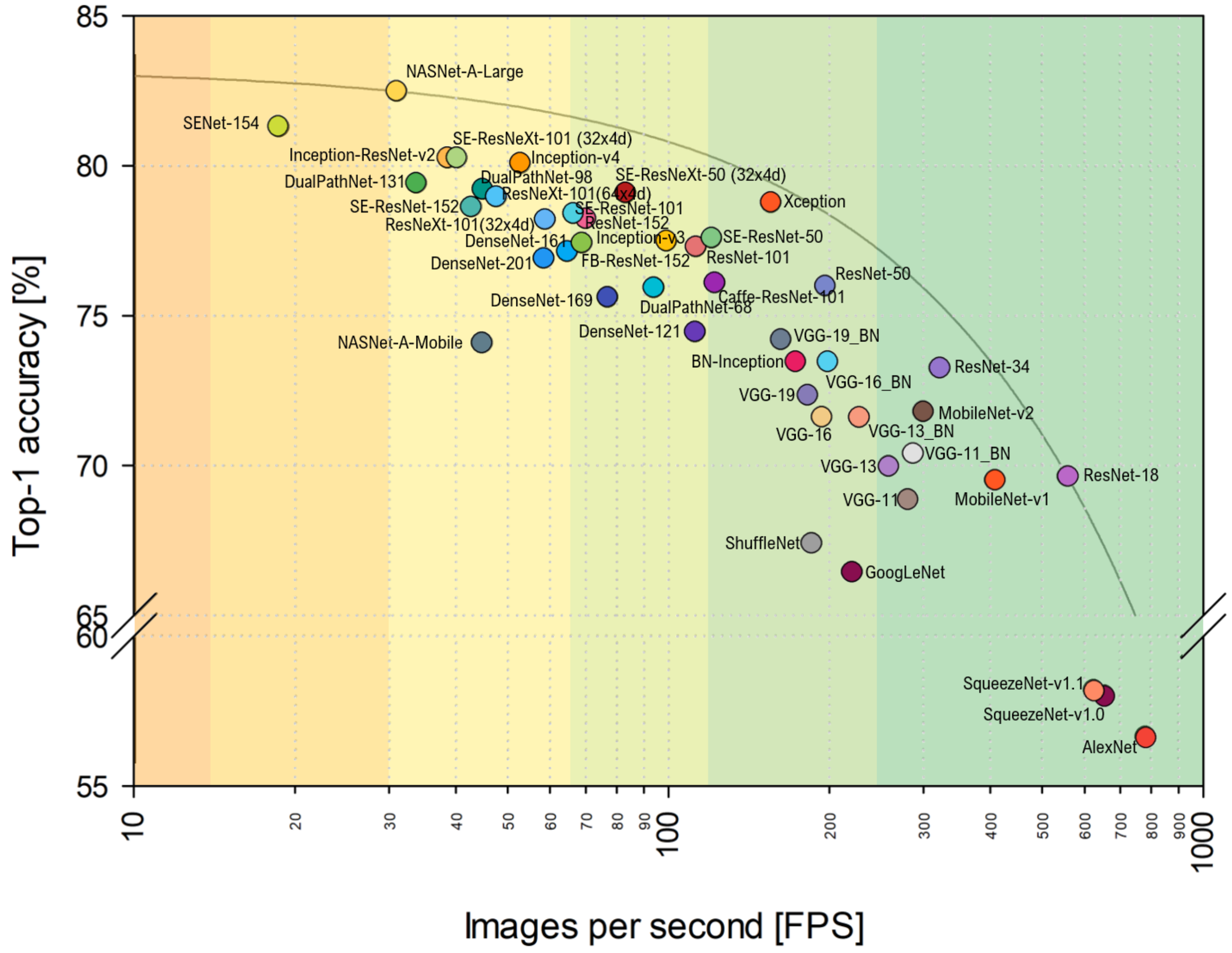
Object Detection



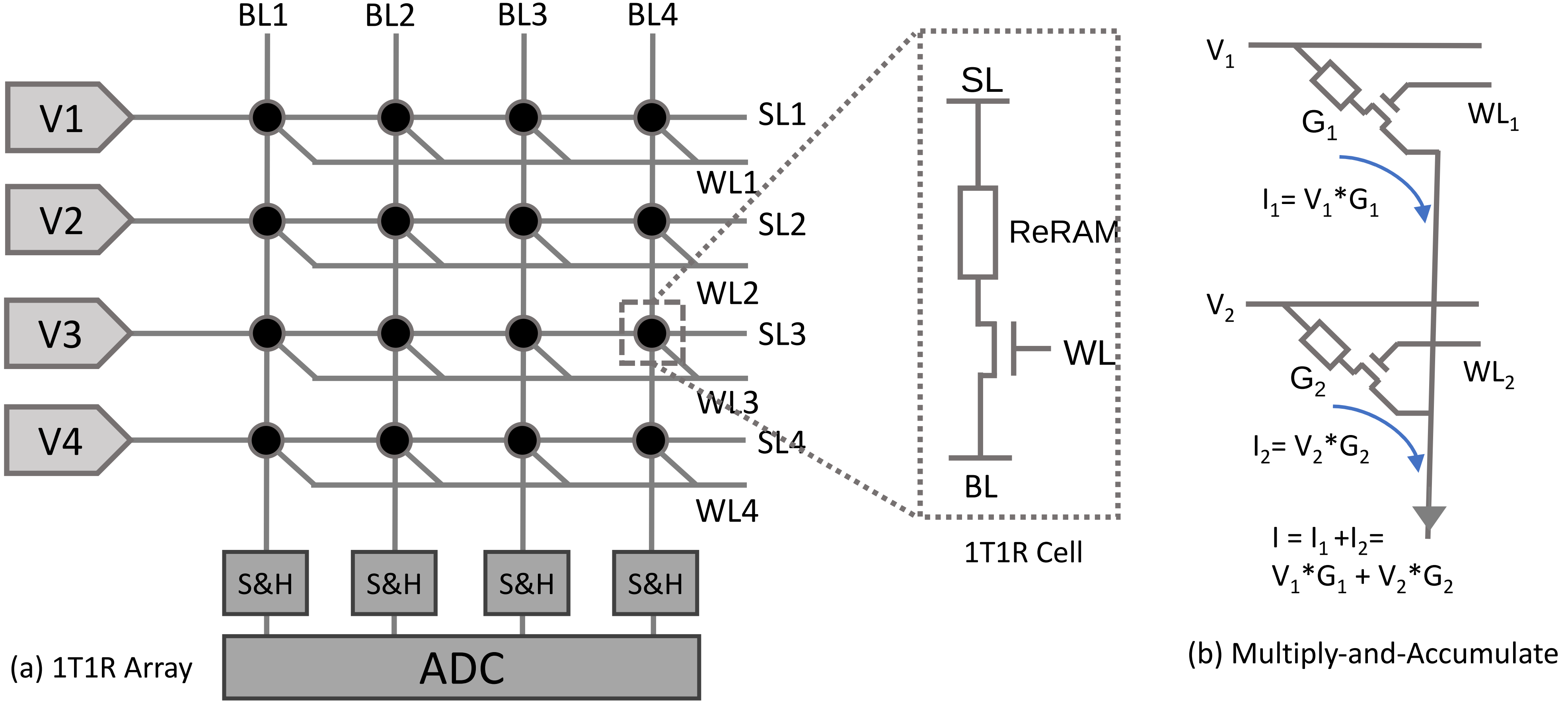
Autonomous Driving

Issues: “Catastrophic Forgetting” & "Memory-wall"

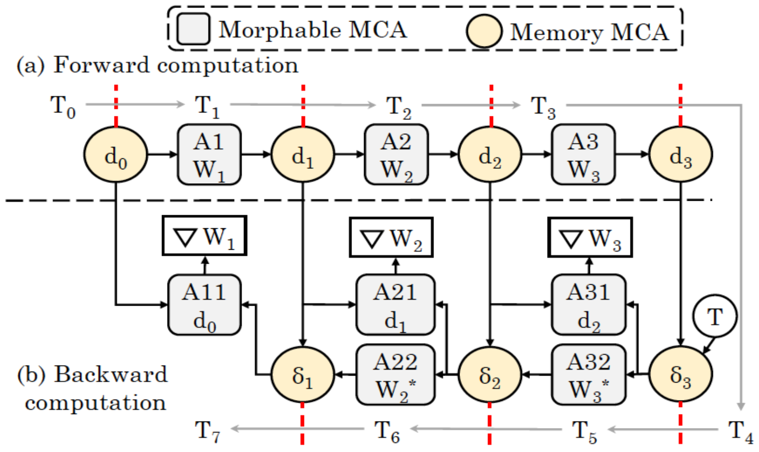




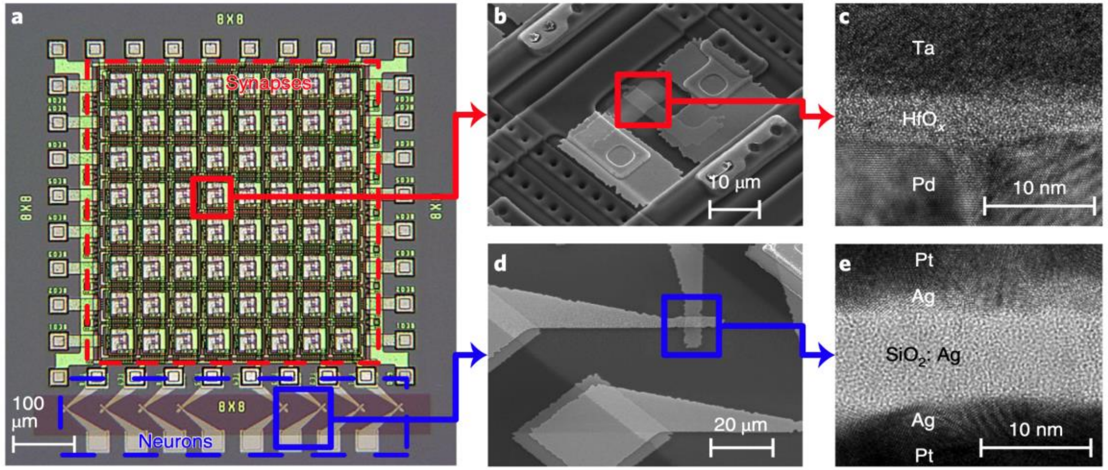
* Bianco, S.(2018). Benchmark Analysis of Representative Deep Neural Network Architectures. IEEE Access, 6, 64270–64277.



BL1
BL2
BL3
BL4
V1
V2
V3
V4
SL1
WL1
SL2
WL2
SL3
WL3
SL4
WL4
S&H
S&H
S&H
S&H
ADC
(a) 1T1R Array
SL
ReRAM
WL
BL
1T1R Cell
V_1
G_1
WL_1
$I_1= V_1*G_1$
V_2
G_2
WL_2
$I_2= V_2*G_2$
$I = I_1 +I_2=$
$V_1*G_1 + V_2*G_2$
(b) Multiply-and-Accumulate



PipeLayer(HPCA'17)



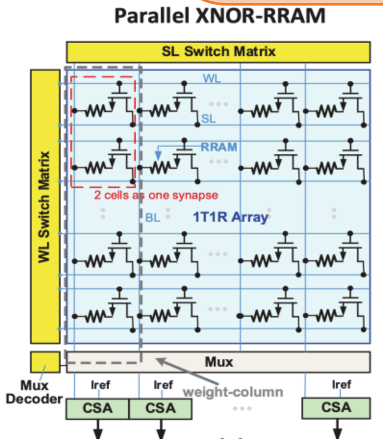
Nature Electronics'18

Only focus on single task DNN model.

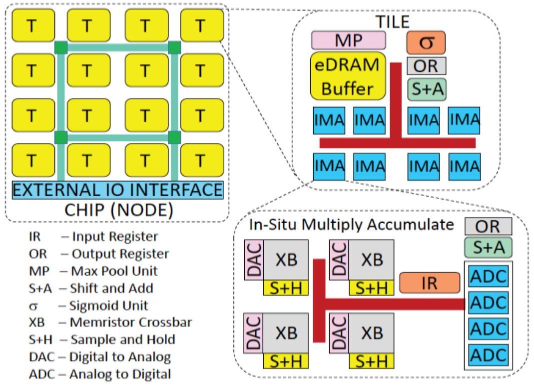
Not suitable for the multitask adaption.

PRIME(ISCA'16)

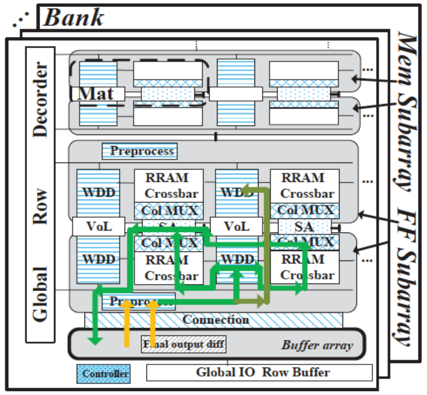
AtomLayer(DAC'18)



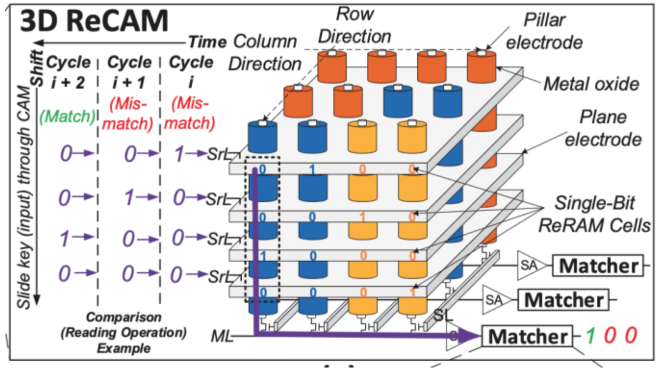
XNOR-RRAM(DAC'18)



ISAAC(ISCA'16)



Time(DAC'17)



RADAR(DAC'18)

Key Design Requirements

01

ReRAM Friendly

No need to update the weight stored in ReRAM.

02

Easy to Implement

Easy to implement on the existing ReRAM based accelerator with minimum overhead.

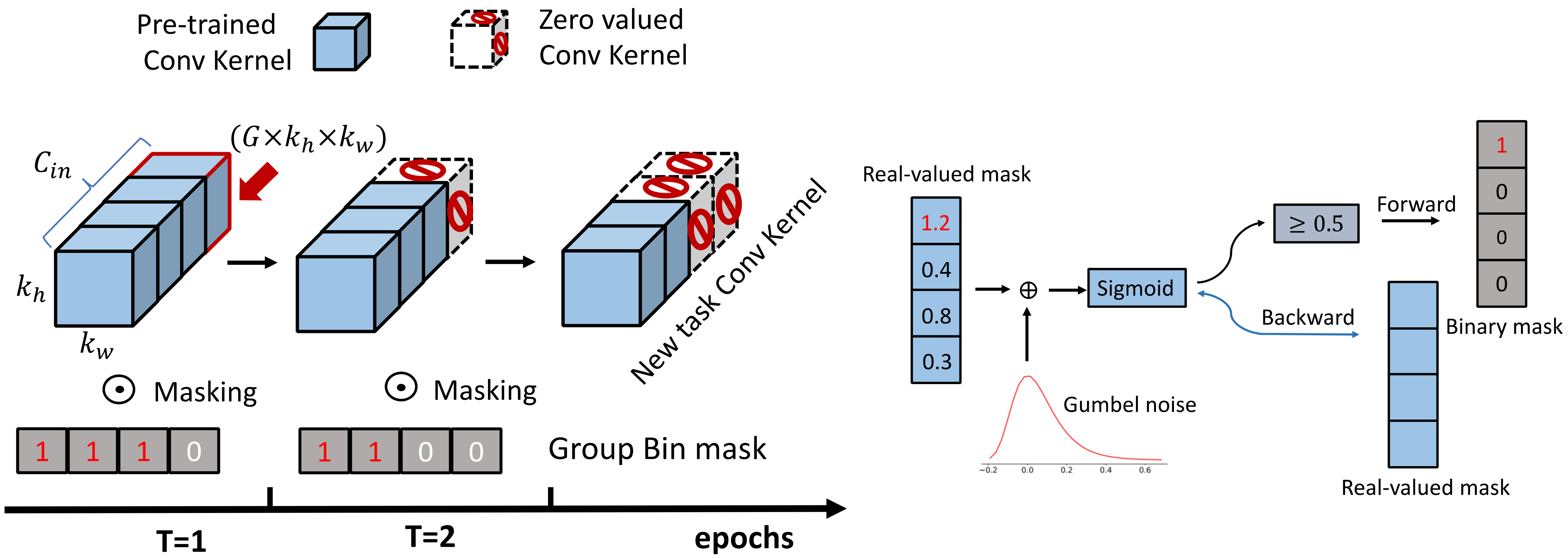
03

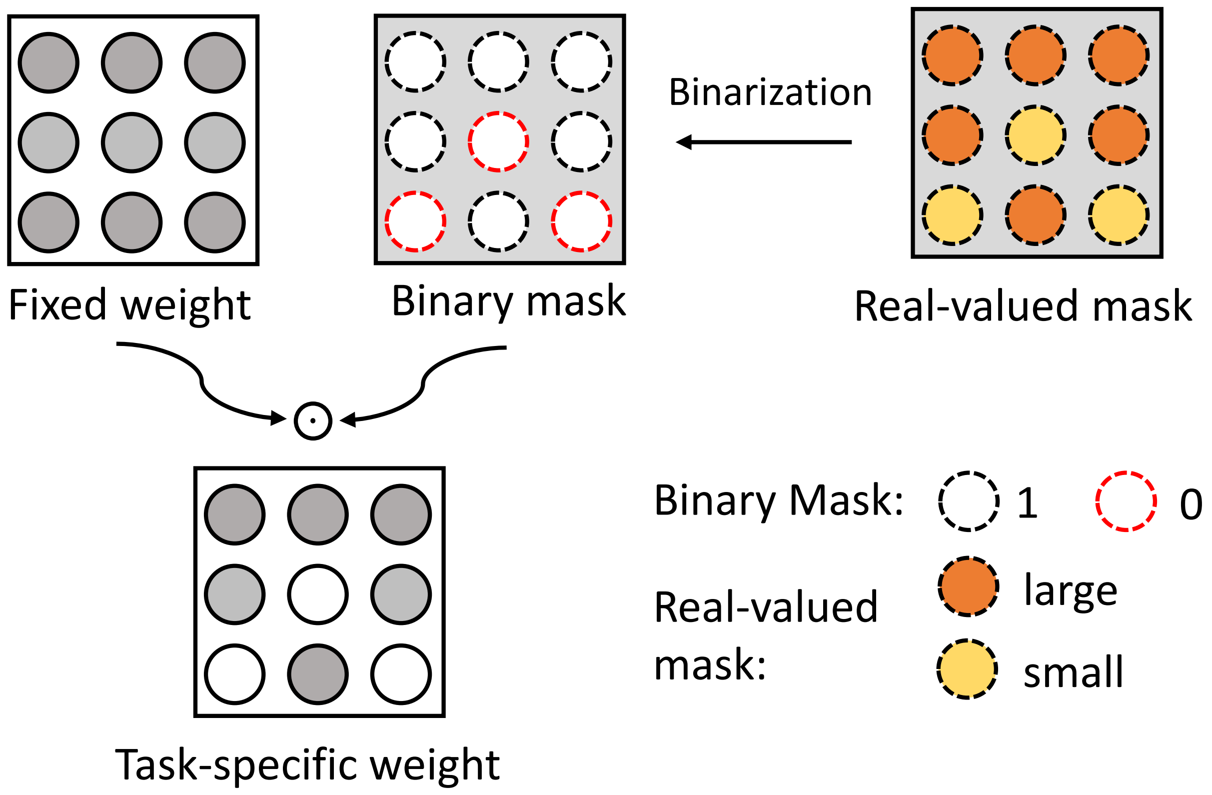
High Accuracy

Do not or minimize the harm on accuracy.

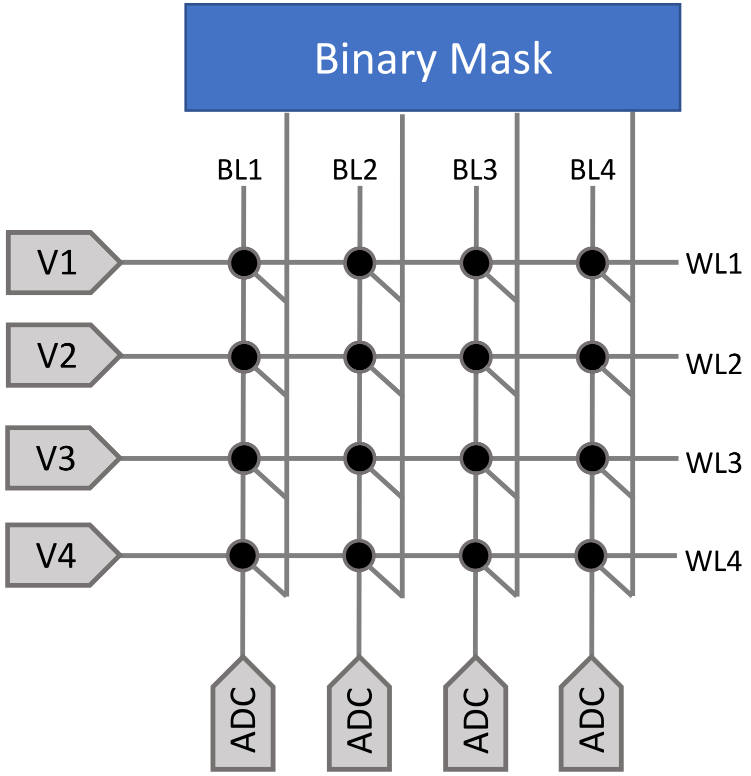
02 Algorithm

- ✓ Mask-based Multitask Adaption
- ✓ Column-wise Mask
- ✓ Gumbel-Sigmoid trick


Pre-trained
Conv Kernel
Zero valued
Conv Kernel
C_{in}
$(G\times k_h\times k_w)$
k_h
k_w
New task Conv Kernel
⊙ Masking
⊙ Masking
1 1 1 0
1 1 0 0
Group Bin mask
T=1
T=2
epochs
Real-valued mask
1.2
0.4
0.8
0.3
⊕
Sigmoid
Gumbel noise
≥ 0.5
Forward
Backward
1
0
0
0
Binary mask
Real-valued mask



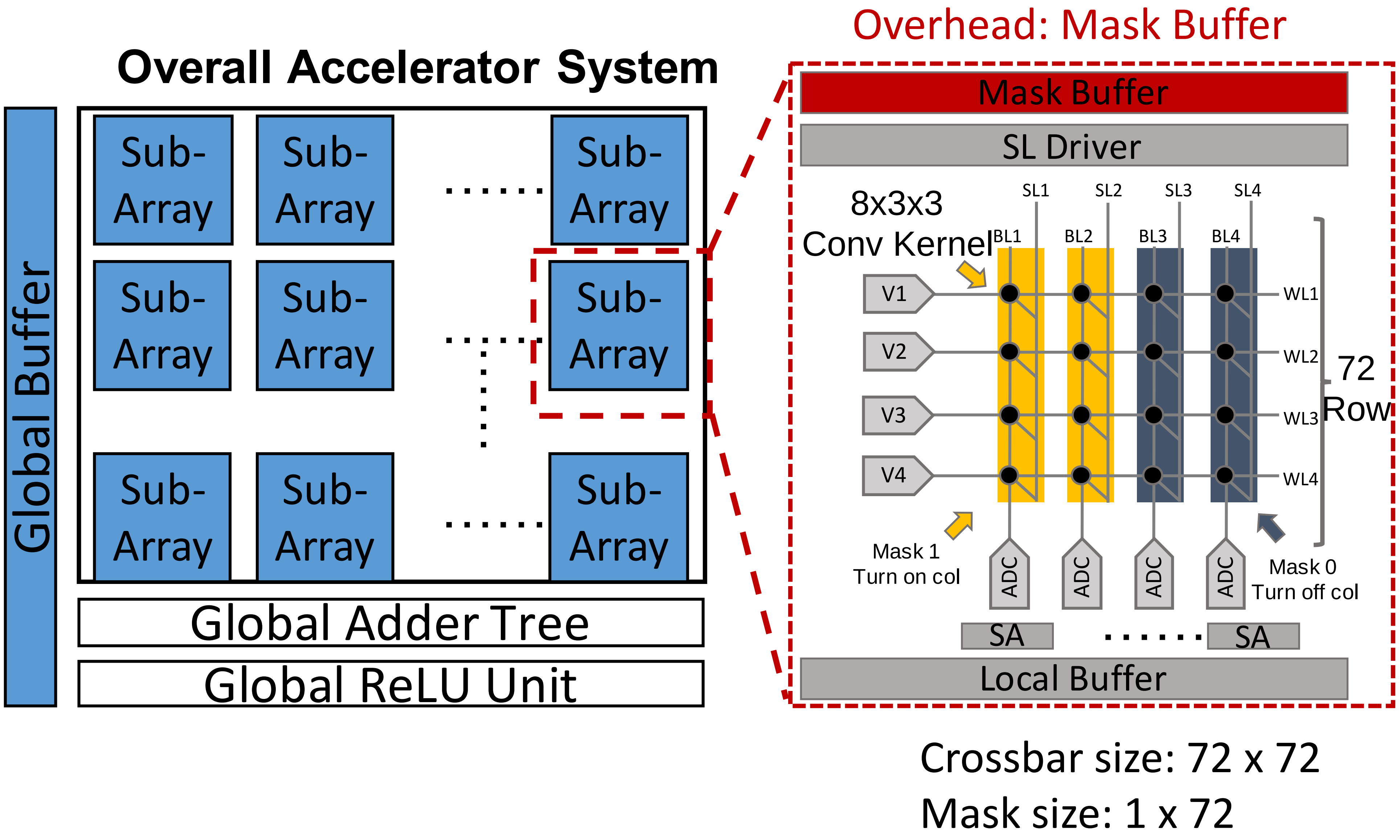
Binarization
Fixed weight
Binary mask
Real-valued mask
Task-specific weight
Binary Mask:
1
0
Real-valued mask:
large
small



Binary Mask
BL1
BL2
BL3
BL4
V1
V2
V3
V4
WL1
WL2
WL3
WL4
ADC
ADC
ADC
ADC

03 Hardware

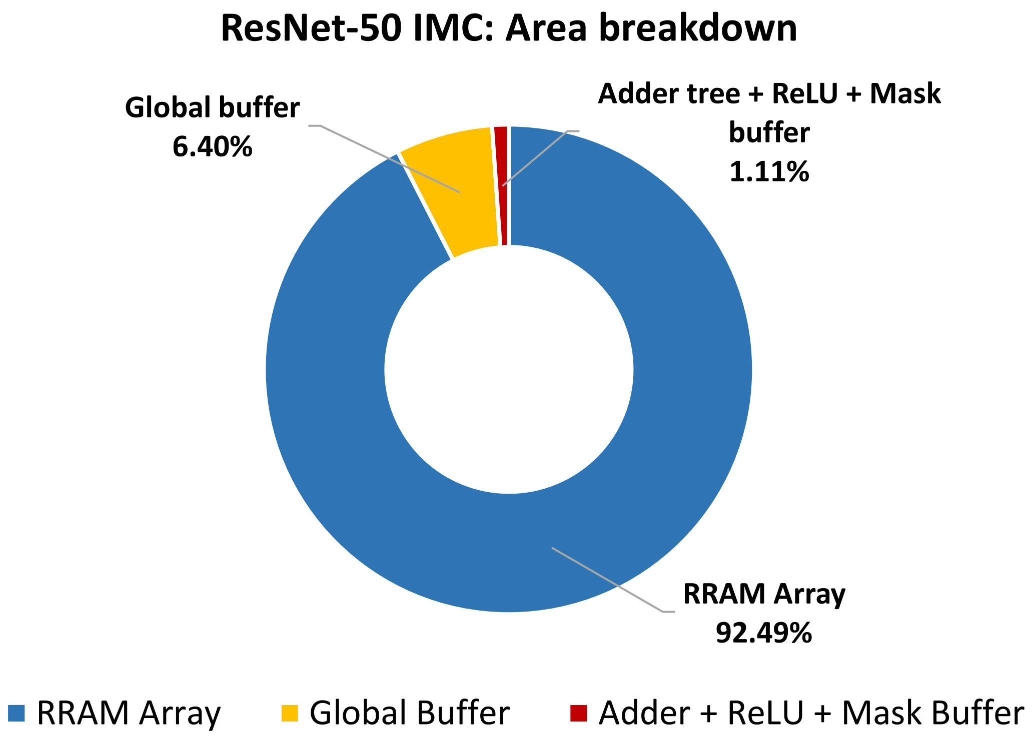
- ✓ ReRAM-based NN Accelerator
- ✓ Hardware Specification
- ✓ Area Breakdown



Overall Accelerator System
Global Buffer
Sub-Array
Sub-Array
Sub-Array
Sub-Array
Sub-Array
Sub-Array
Sub-Array
Sub-Array
Sub-Array
Global Adder Tree
Global ReLU Unit
Overhead: Mask Buffer
Mask Buffer
SL Driver
8x3x3
Conv Kernel
SL1
SL2
SL3
SL4
BL1
BL2
BL3
BL4
V1
V2
V3
V4
WL1
WL2
WL3
WL4
72
Row
Mask 1
Turn on col
Mask 0
Turn off col
ADC
ADC
ADC
ADC
SA
SA
Local Buffer
Crossbar size: 72 x 72
Mask size: 1 x 72

Hardware specification

RRAM Sub-Array		
Components	**Area** (μm^2)	**Energy** (pJ)
Memory Array (72×72)	84.93	
Switch Matrix (WL and SL)	457.3	1.1
SAR ADC (5-bit)	8,409.3	8.3
Shift-Add-Input	1,412.9	6.8
Shift-Add-Weight (2 col use 1)	825.8	1.0
Mask Buffer (72×1)	190.4	0.003/bit/access
Total	11,380.2	17.2
Peripheral Circuits		
1 stage AdderTree (128 units)	2,510.3	4.4
2 stage AdderTree (128 units)	7,740.1	13.7
3 stage AdderTree (128 units)	18,408.8	32.6
Global Buffer ($64 \times 112 \times 112 \times 4$)	8,490,034	0.003/bit/access
ReLU (128 units)	939.5	0.9



For the ResNet-50 backbone model, the totally model size is 23M parameters while the XBM only needs 40KB for binary mask.

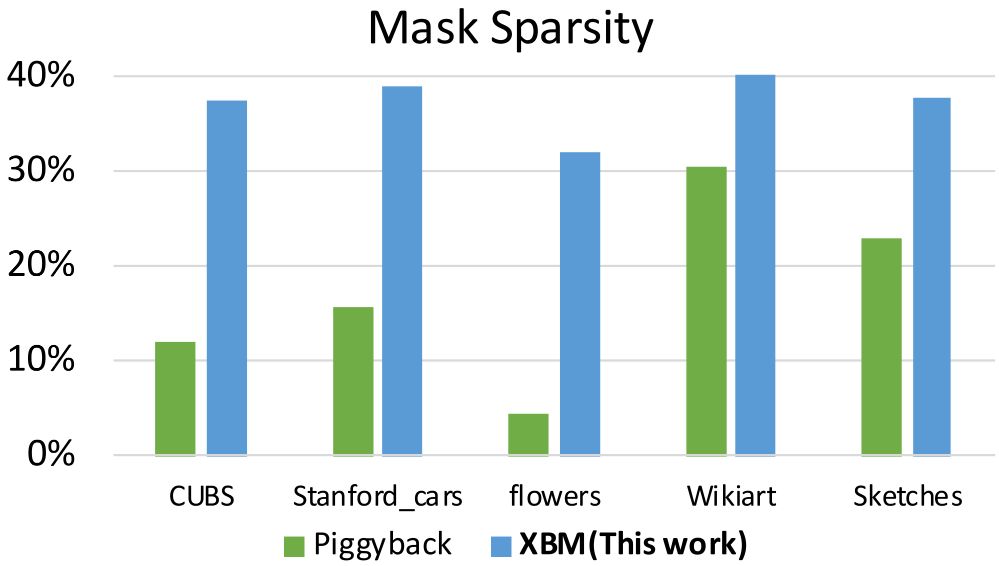
04 Experiment

- ✓ Algorithm Performance
- ✓ Hardware Evaluation
- ✓ Comparison

Multi-task Adaption Accuracy

		Continual Learning (4-bit Quantization)	
	Finetune	Piggyback	XBM (This work)
CUBS	73.02%	74.47%	**75.53%**
Stanford_cars	85.92%	**86.85%**	85.96%
flowers	**95.34%**	91.09%	90.81%
Wikiart	**74.96%**	68.97%	67.6%
Sketches	**80.92%**	78.88%	76.95%

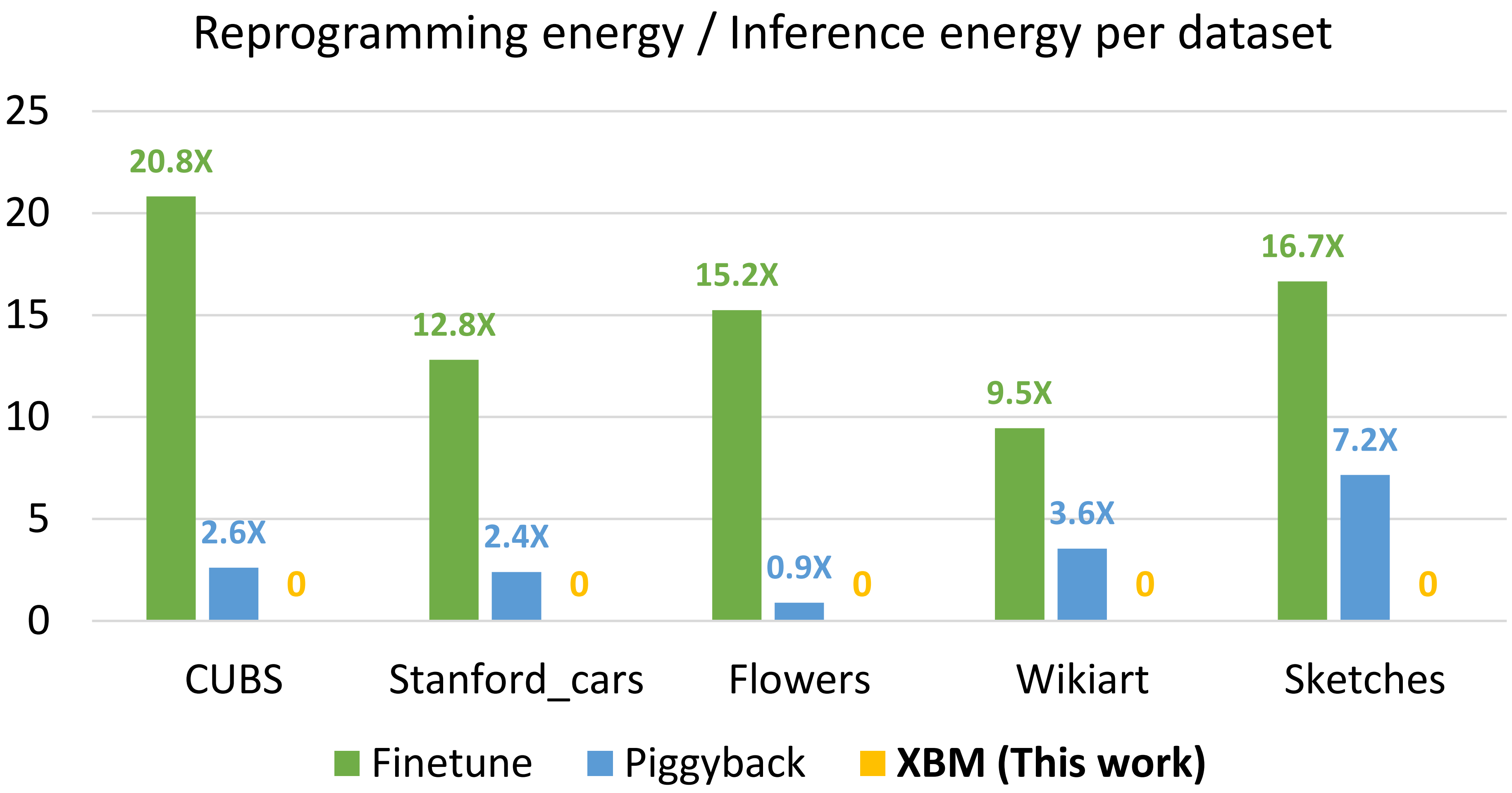
Backbone model: ResNet50 pre-trained on ImageNet dataset.
Piggyback uses an element-wise binary mask without gumbel-sigmoid trick
Both Weight and Activation have been quantized to 4-bit.



Inference Energy perimage

	4-bit ResNet-50		
Method / Dataset	Finetune	Piggyback	**XBM(Binary Group Mask)**
CUBS	30.25 μJ	30.25 μJ	21.20 μJ
Stanford_cars	30.25 μJ	30.25 μJ	20.82 μJ
flowers	30.25 μJ	30.25 μJ	22.53 μJ
Wikiart	30.25 μJ	30.25 μJ	20.63 μJ
Sketches	30.25 μJ	30.25 μJ	21.12 μJ

XBM: Turn-off unnecessary columns ➔ Saving Inference Energy



Finetune: Reprogramming almost the entire crossbar-array.

Piggyback: Reset partial weight according to binary mask.

XBM: No need of reprogramming.

Conclusions

- Memory Bottleneck
- Catastrophic Forgetting

1) Crossbar friendly multi-task learning method

✓Binary mask to overcome the catastrophic forgetting

✓Gumbel-Sigmoid trick

2) Hardware friendly crossbar column-wise mask

✓Mask size reduction

✓Minimum hardware overhead and modification

3) Comparison

✓Performance comparison with State-of-the-art mask-based method

Thanks for your listening

Question?

Fan Zhang : fzhang95@asu.edu

❖ This work is supported in part by the National Science Foundation under Grant No. 1931871, No.2003749