Mortar: Morphing the Bit Level Sparsity for General Purpose Deep Learning Acceleration

Yunhung Gao[3], Hongyan Li[12], Kevin Zhang[3], Xueru Yu[4], Hang Lu[12]

[1]State Key Laboratory of Computer Architecture, Institute of Computing Technology, CAS, Beijing, China

[2]University of Chinese Academy of Sciences, Beijing, China

[3]School of Electronics Engineering and Computer Science Peking University Beijing, China

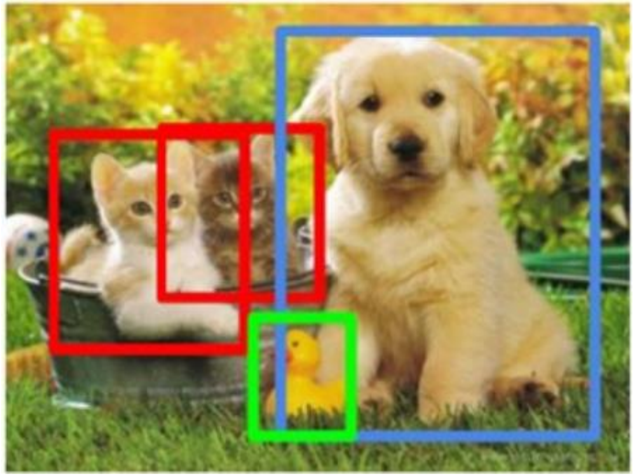
[4]Shanghai Integrated Circuits R&D Center Co. Ltd Shanghai, China

计算机体系结构国家重点实验室
State Key Laboratory of Computer Architecture, ICT, CAS

中国科学院计算技术研究所
INSTITUTE OF COMPUTING TECHNOLOGY, CHINESE ACADEMY OF SCIENCES

Introduction

DNN networks

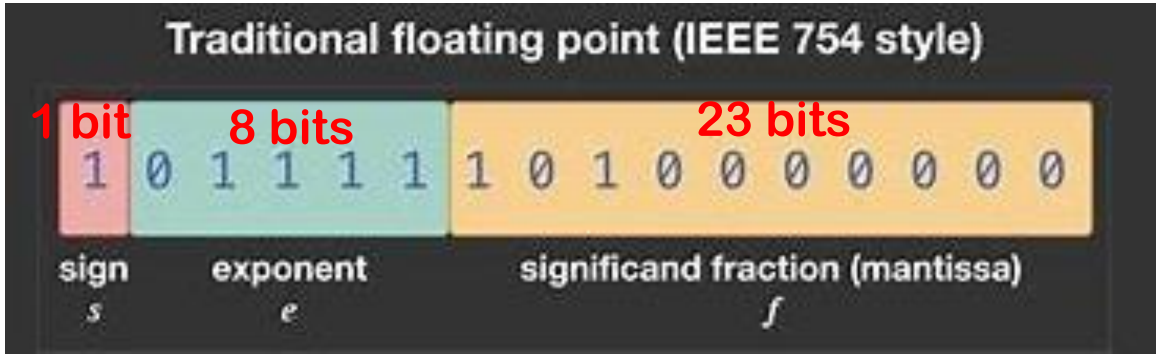


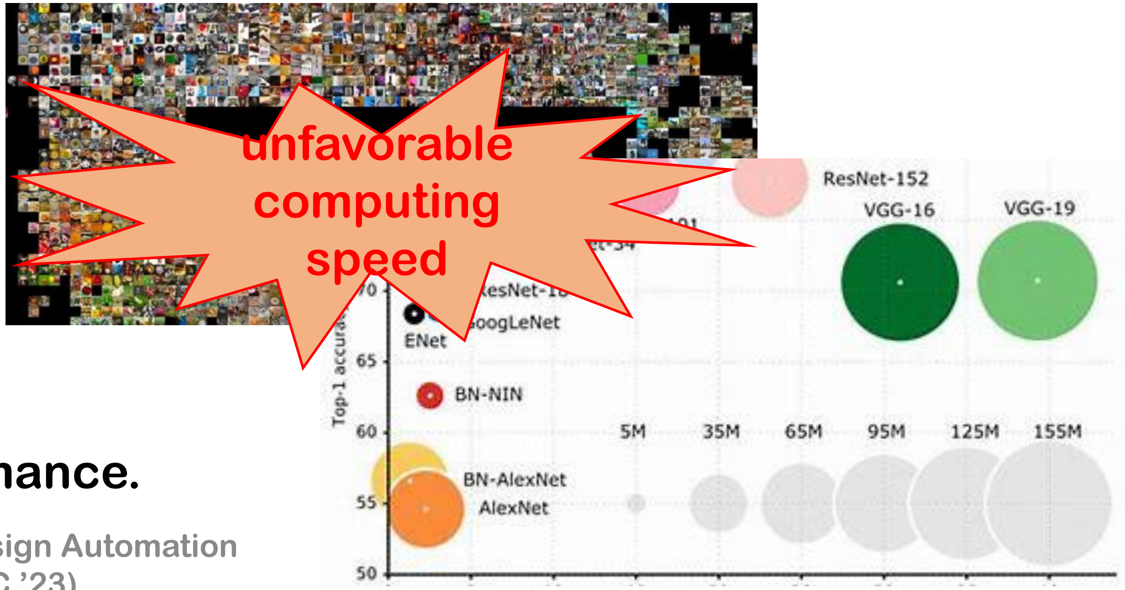
`torch.float32`

PyTorch

`tf.float32`

TensorFlow





Floating point 32 DNN usually exhibits satisfiable performance.

Introduction

The contributions of this work

Propose Mortar, a novel on/off-line collaborative approach for general purpose deep learning acceleration

① An off-line mantissa morphing algorithm to get higher model accuracy & higher bit-level sparsity

Mortar maintains on average 3.55% higher model accuracy while increasing more bit-level sparsity than the baseline.

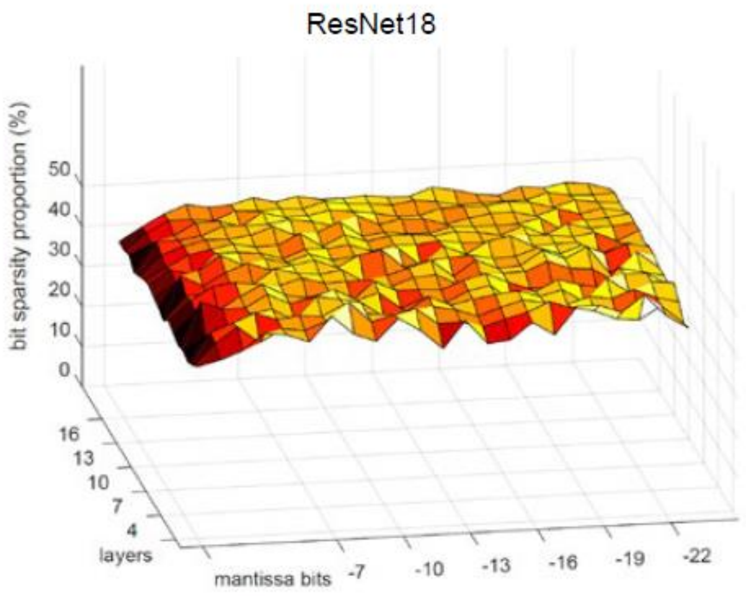
② The on-line associating hardware accelerator to speed up the on-line fp32 inference

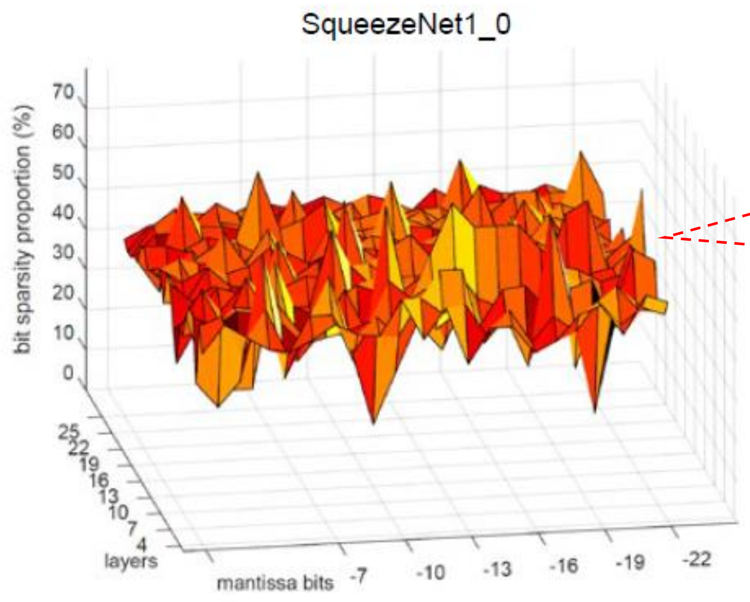
The hardware accelerator outperforms up to 4.8x over the baseline, with an area of 0.031 mm^2 and power of 68.58 mw.

Motivation

Sparsity Parallelism

X-axis : the bit slice of the binary represented weight (in fp32)
Y-axis : the sequential layers of the model
Z-axis : the fraction of bit ‘1’



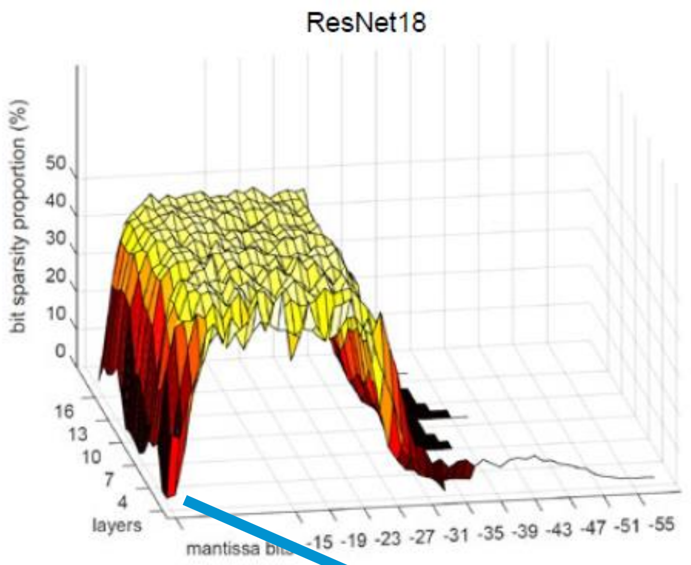


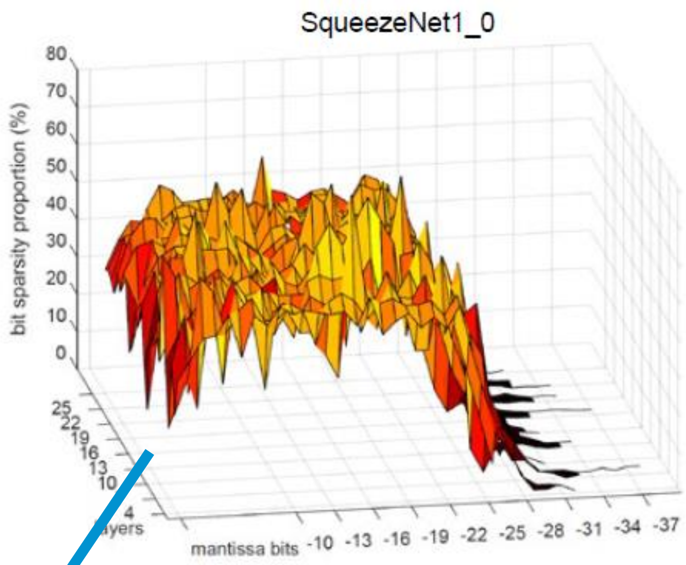
The sparsity distribution is even (~50%) throughout the bit positions.

Motivation

Sparsity Irregularity

Bit-level sparsity after exponent matching



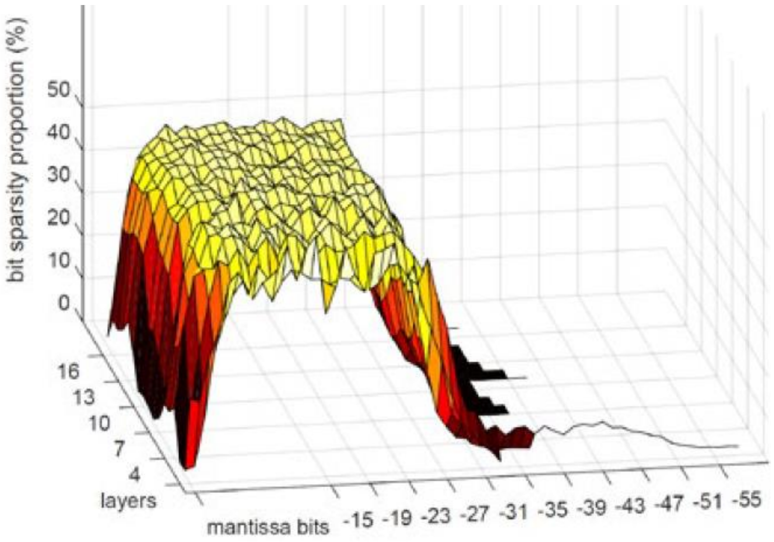


Less than 10%

- All the evaluated DNNs exhibit an "arched" shape.
- Exponent matching will generate more sparsity due to the shifting operation making the sparsity distribution highly irregular.

Problem tackled in this work

Goal：computing fewer 1 bits while still maintaining target accuracy



front 0 bits

migrate

rear 1 bits

trivial in value

accelerate networks

front 1 bits

compensate

rear 0 bits

We intend to solve the problem in Mortar!

Methodology – Mantissa Morphing

Mantissa Morphing

$$Precision = \left(\left| \frac{W_i' - W_i}{W_i} \right| = \left| \frac{\varepsilon_i}{W_i} \right| < P \right) \,?\, 1 : 0$$

a hyperparameter balancing the tradeoff between sparsity and accuracy.

Algorithm 1: Mantissa Morphing

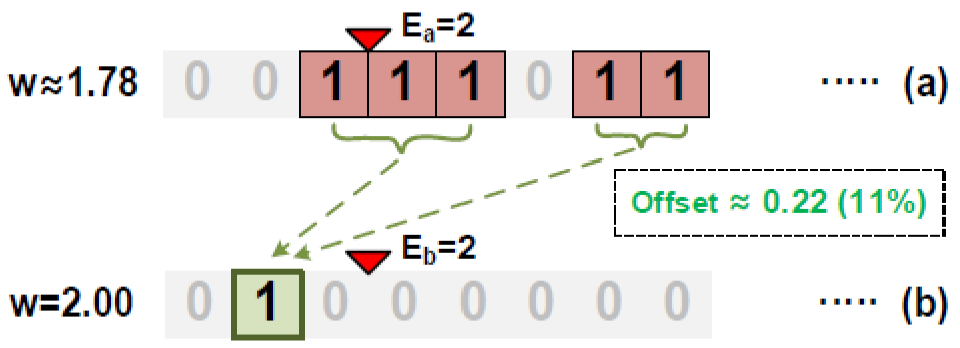
Input: Original fp32 weight, W_i

Output: New weight after mantissa morphing, $W_i{}'$,

```
1:  Interpret the n-bit exponent E = [e_1, ..., e_n] and mantissa
2:  M = [m_1, ... m_n], the actual position of E is determined.
3:  Set the value for parameter 'P' in Precision function
4:  foreach column j in W
5:     if W_j = 1 and W_{j-1} = 0:
6:        W'_{j-1} = 1;
7:        foreach column k in W [j : c]
8:           W'_k = 0;
9:        if (Precision(W, W', P)) # precision judge
10:          Return W', j + 1;
11:       else W' = W;
12:          continue
```

*Loop 7 can be parallelized for speedups

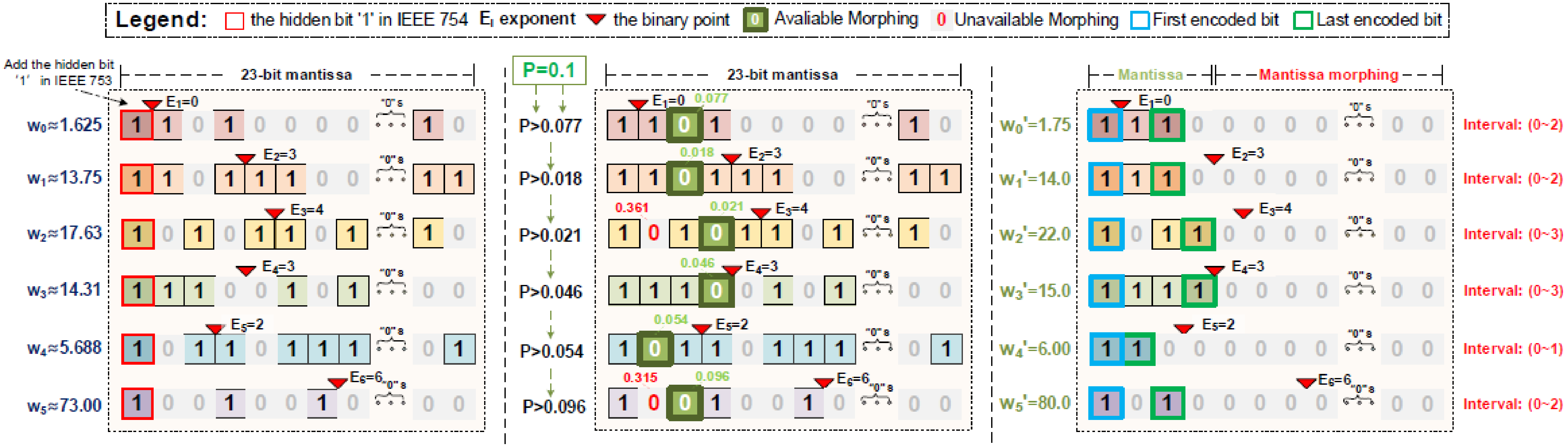
An example of Mantissa Morphing



$$\frac{number\ of\ bit\ 1s\ in\ (a)}{number\ of\ bit\ 1s\ in\ (b)} = 5$$

Methodology – core concept

off-line bit morphing



(a) **Step 1**: preprocessing the floating-point weights

(b) **Step 2**: selecting the bit for 'mantissa morphing'

(c) **Step 3**: applying 'mortar encoding' to infer computing interval for each weight

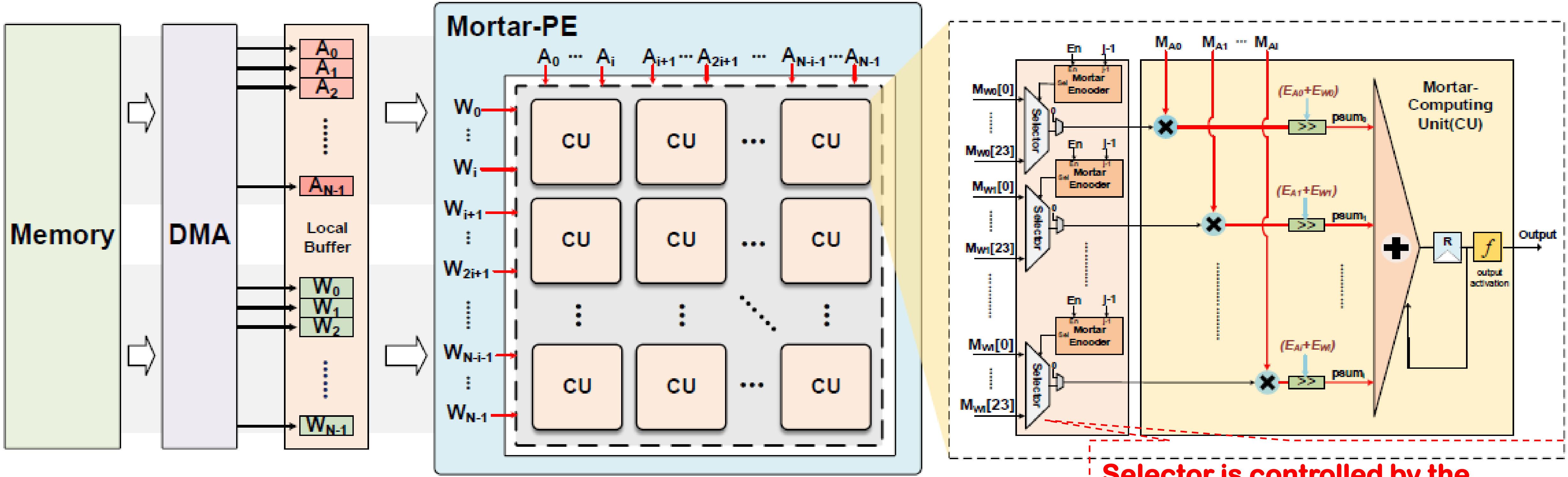
The sparsity and regularity of weights' bits have been significantly improved.

less computation

less hardware design overheads

Methodology – Mortar accelerator

on-line hardware accelerator



Selector is controlled by the 'mortar encoding' signal when receiving the weight mantissa.

Evaluation

- **Benchmark models**

Models	Domain	Type	Dataset	Metric	GFLOPS	Weights (M)
DenseNet161	Image Classification	2D Convolution	ILSVRC2012	Top-1 %	15.64	28.68
ResNext101	Image Classification	2D Convolution	ILSVRC2012	Top-1 %	33.02	88.79
ResNet18	Image Classification	2D Convolution	ILSVRC2012	Top-1	3.64	11.69
YoloV3	Object Detection	2D Convolution	COCO	mAP	25.42	61.95
FCOS	Object Detection	Feature Pyramid	COCO	mAP	80.14	32.02
ViT	Video Understanding	Transformers	ILSVRC2012	Top_1(%)	29.42	86.61
D3DNet	Video Super Resolution	3D Deformable	Viemo-90k	PSNR SSIM	408.82	2.58
LapSRN	Image Super Resolution	2D De-Convolution	SET14	/	736.73	0.87
CartoonGAN	Style Transfer	GAN	Flickr	/	108.98	11.69

Various benchmark models in different domains demonstrate the generalization capability of Mortar.

Evaluation

- **Design space exploration**

P balances the tradeoff between sparsity and accuracy.

Model	Baseline	P=0.0001	P=0.0005	P=0.001	P= 0.005	P=0.01	P=0.05	P=0.1	P=0.3	P=0.5
DenseNet161	75.28	75.28	75.28	75.28	75.28	75.27	75.29	75.31	75.37	75.06
ResNext101	78.24	78.24	78.24	78.25	78.25	78.27	78.29	78.26	77.93	77.41
ResNet18	67.28	67.29	67.29	67.29	67.29	67.28	67.28	67.22	67.13	66.92
YoloV3	52.73	52.75	52.75	52.75	52.73	52.73	52.69	52.50	51.72	51.38
FCOS	0.382	0.382	0.382	0.382	0.382	0.382	0.382	0.378	0.318	0.258
ViT	83.89	83.89	83.89	83.88	83.88	83.88	83.76	83.66	83.33	82.91
D3DNet	36.05	36.05	36.05	36.05	36.05	36.05	36.05	36.05	36.02	35.97
	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94	0.94

For most models, setting 0.0001< P <0.05 maintains equal accuracy with the baseline, and even improvements in certain cases. However, most model accuracy starts to decrease when P≥0.1.

Evaluation

- **Accuracy & sparsity**

Model accuracy and sparsity change after applying mantissa morphing at P=0.1.

Models	Baseline		Mortar	
	Accuracy / Sparsity			
DenseNet161	75.28/1x		75.37/1.58x	
ResNext101	78.24/1x		78.26/1.81x	
ResNet18	67.28/1x		67.22/2.09x	
YoloV3	52.73/1x		52.50/1.28x	
FCOS	0.382/1x		0.378/1.38x	
ViT	83.89/1x		83.66/2.51x	
D3DNet	36.05	1x	36.05	2.28x
	0.94		0.94	
Avg. loss/sparsity	0.00/1x		**-0.06/1.85x**	

In general cases, the sparsity improvement of a model can reach ~2x with a neglectable accuracy degradation.

Accuracy/Sparsity comparison with BitX [1] .

Models	Original	BitX	Mortar
DenseNet161	75.28/1x	74.79/1.61x	**75.37/1.58x**
ResNext101	78.24/1x	73.00/1.74x	**78.26/1.81x**
ResNet18	67.28/1x	62.52/1.90x	**67.22/2.09x**
Avg. loss / sparsity	0.00/1x	-3.50/1.75x	**+0.05/1.83x**

Mortar greatly outperforms SOTA approach in accuracy while maintaining a slightly improved sparsity.

[1] H. Li et al., “BitX: Empower Versatile Inference with Hardware Runtime Pruning,” in ICPP, 2021.

Evaluation

- **Visual Comparison**

Visual demonstrations of 4x super resolution inference via Mortar and cartoon style transfer via Mortar

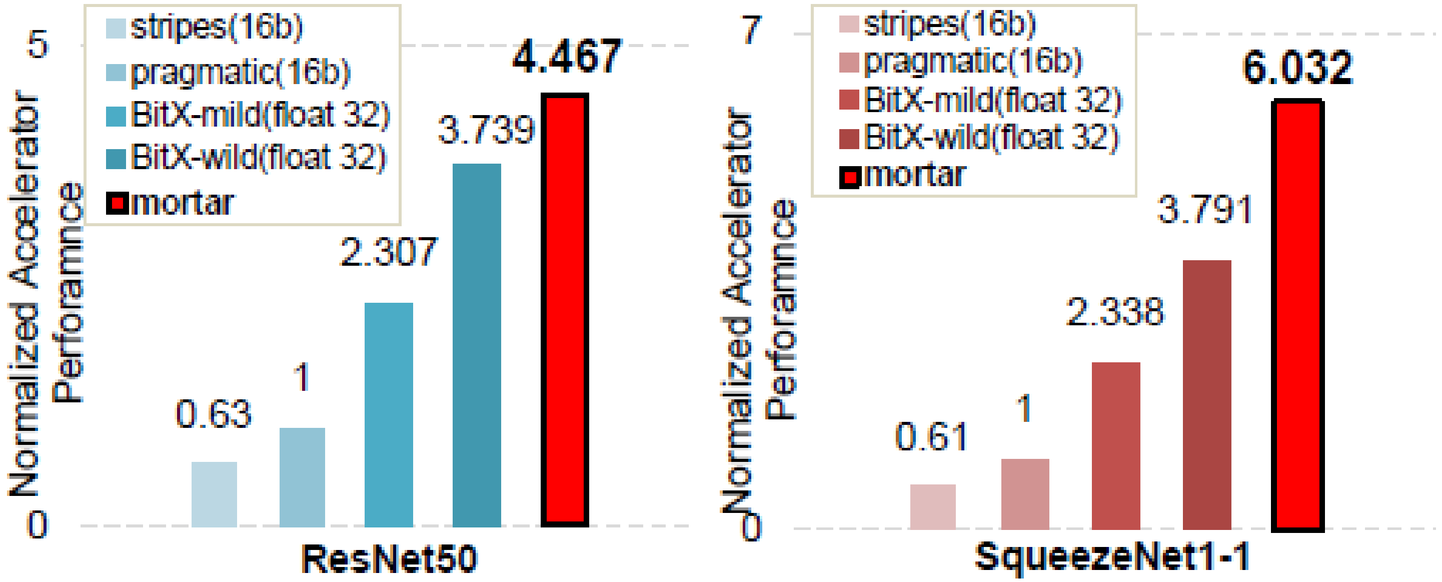
(a) visual comparison for LapSRN

(b) visual comparison for CartoonGAN

The results are nearly indistinguishable which proves that Mortar could attain faster inference with the maintained Quality of Result.

Evaluation

- **Comparison with SOTA Accelerators**



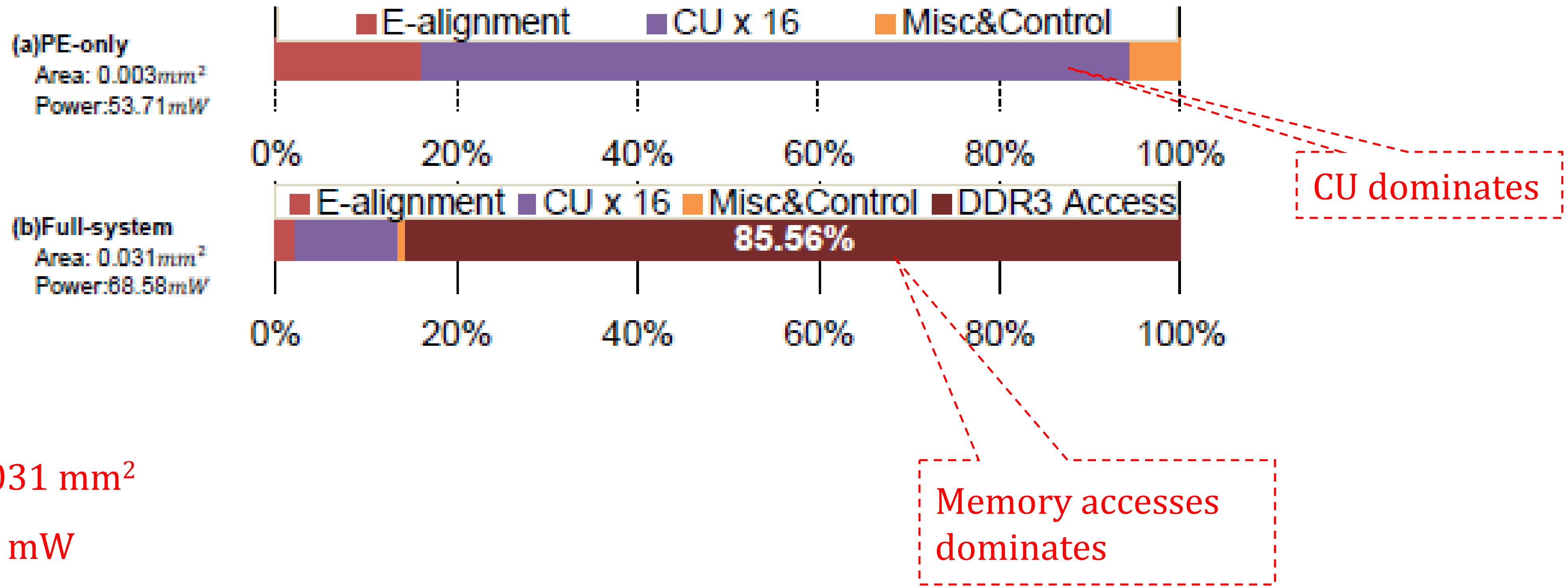
1. The speedup shows better result over other SOTA accelerators.

2. Mortar's performance on ResNet50 is 4.467x that of the baseline. On SqueezeNet1_1, Mortar outperforms the baseline by 6.032x.

Evaluation

- **Accelerator energy and area breakdown**

Mortar' Area & Energy Breakdown for PE-only and full system



1. Area : only 0.031 mm^2
2. Power : 68.58 mW

Recap

The contributions of this work

① Propose a novel off-line mantissa morphing algorithm – "Mortar"

- significantly increase the bit-level sparsity
- accelerate fp32 inference while maintaining high accuracy
- show the strong generalization capabilities on various models

② Propose the associating on-line hardware accelerator "Mortar accelerator"

Applications, and what's more?

Thanks for listening!

Questions?

Yunhung Gao[3], Hongyan Li[12], Kevin Zhang [3], Xueru Yu [4], Hang Lu [12]

[1]State Key Laboratory of Computer Architecture, Institute of Computing Technology, CAS, Beijing, China

[2]University of Chinese Academy of Sciences, Beijing, China

[3]School of Electronics Engineering and Computer Science Peking University Beijing, China

[4]Shanghai Integrated Circuits R&D Center Co. Ltd Shanghai, China

计算机体系结构国家重点实验室
State Key Laboratory of Computer Architecture, ICT, CAS

中国科学院计算技术研究所
INSTITUTE OF COMPUTING TECHNOLOGY, CHINESE ACADEMY OF SCIENCES