EXION:
Exploiting Inter- and Intra-Iteration Output Sparsity for Diffusion Models

HPCA’ 2025

Jaehoon Heo, Adiwena Putra, Jieon Yoon, Sungwoong Yune, Hangyeol Lee, Ji-Hoo

KAIST Daejeon, Republic of Korea

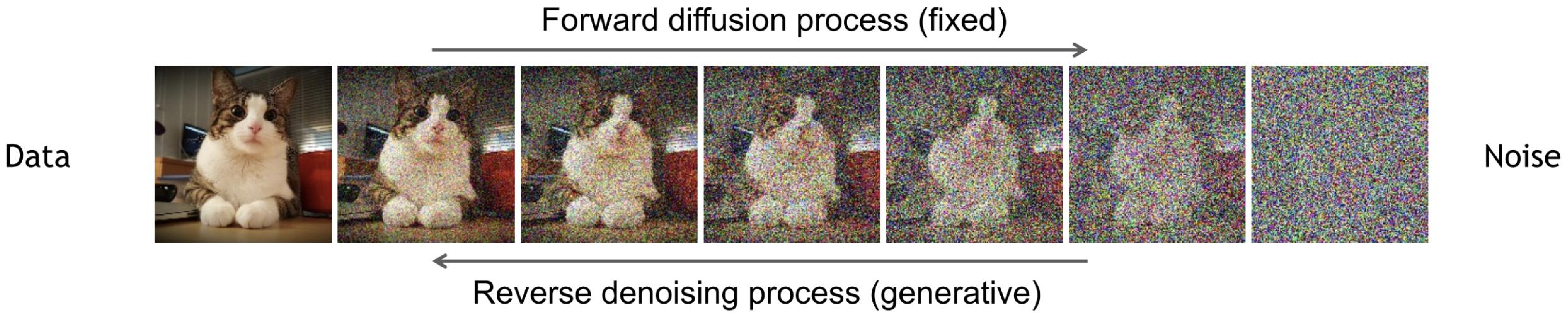
Yuge Cheng, 23/05/2025

Preliminaries

Denoising Diffusion Models: Learning to generate by denoising

Denoising diffusion models consist of two processes:

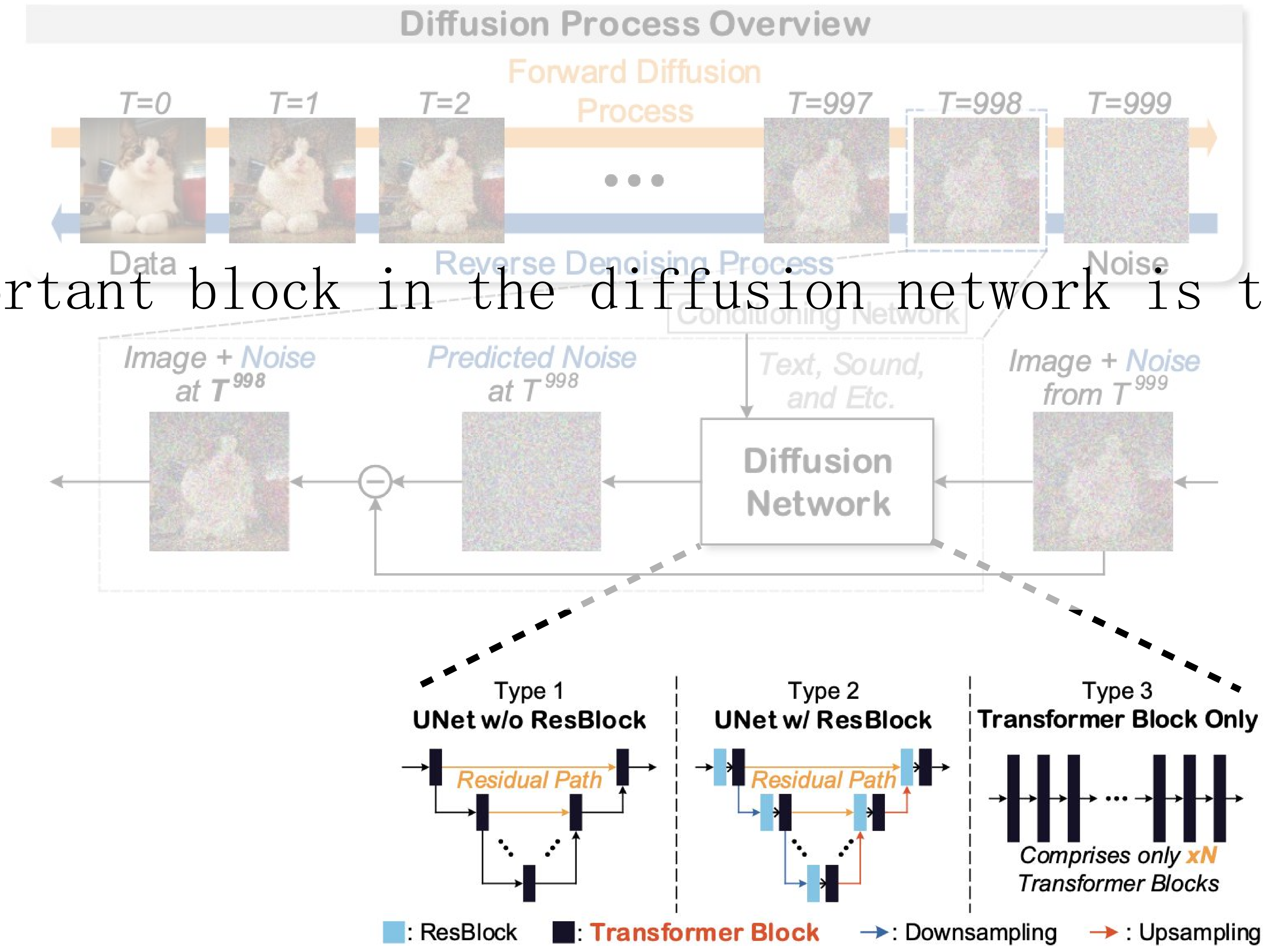
- Forward diffusion process that gradually adds noise to input
- Reverse denoising process that learns to generate data by denoising



Preliminaries

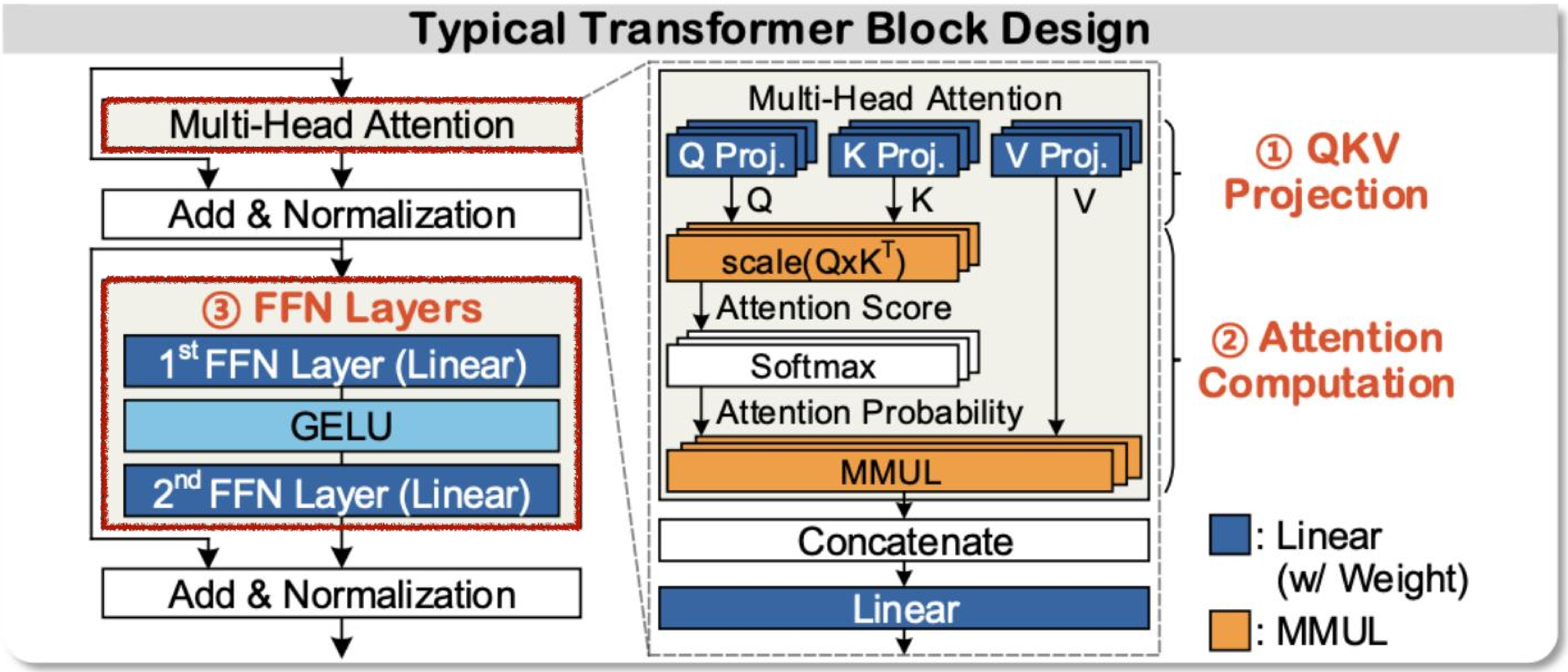
Denoising Diffusion Models: Learning to generate by denoising

The most important block in the diffusion network is the **transformer block !**



Preliminaries

Transformer Block Design



Problems

Better outputs come with *higher* energy consumption and *longer* latency :(

- Each generation requires numerous denoising iterations
- Each iteration evaluates numerous Transformer Blocks
- Each transformer block entails a large number of operations

Metric	StyleGAN-XL	Stable Diffusion
Energy (Joules)	~65.5 J	1546.7 J
Latency (sec)	~1.2 s	11.8 s

X23.6 more energy

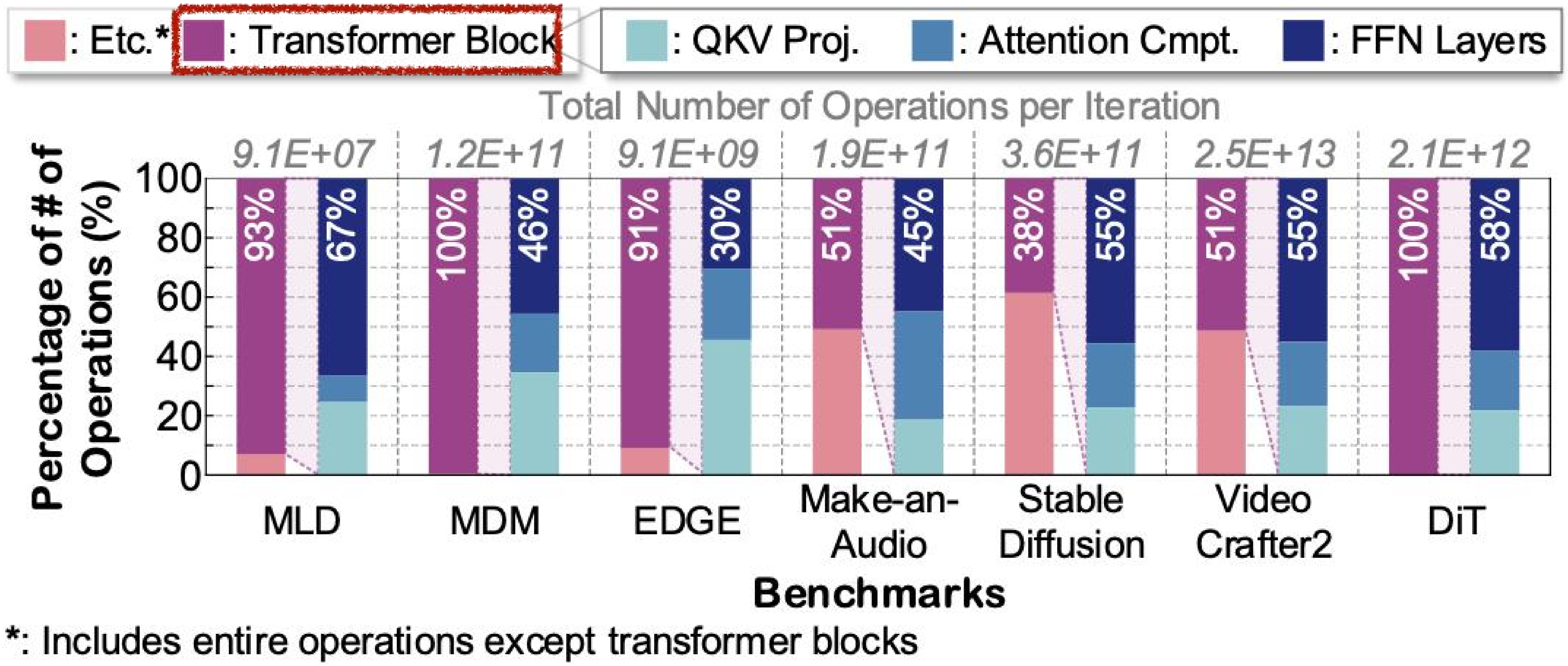
X 9.8 slower

Experiment results on NVIDIA’s RTX 6000 Ada

Problems

Better outputs come with *higher* energy consumption and *longer* latency :(

Fig: Number of Operations Breakdown



- <u>the transformer block</u> accounts for the highest ratio
- <u>the FFN layers</u> are generally the most compute-intensive

Overview

EXION: Exploiting Inter- and Intra-Iteration Output Sparsity for Diffusion Models

Key ideas: exploiting the unique inter- and intra-iteration output sparsity

Overview

EXION: Exploiting Inter- and Intra-Iteration Output Sparsity for Diffusion Models

Key ideas

1. Software optimizations
   - FFN-Reuse across different iterations
   - Sparse attention computation via eager prediction
2. ConMerge: data compaction mechanism
3. Specialized hardware architecture

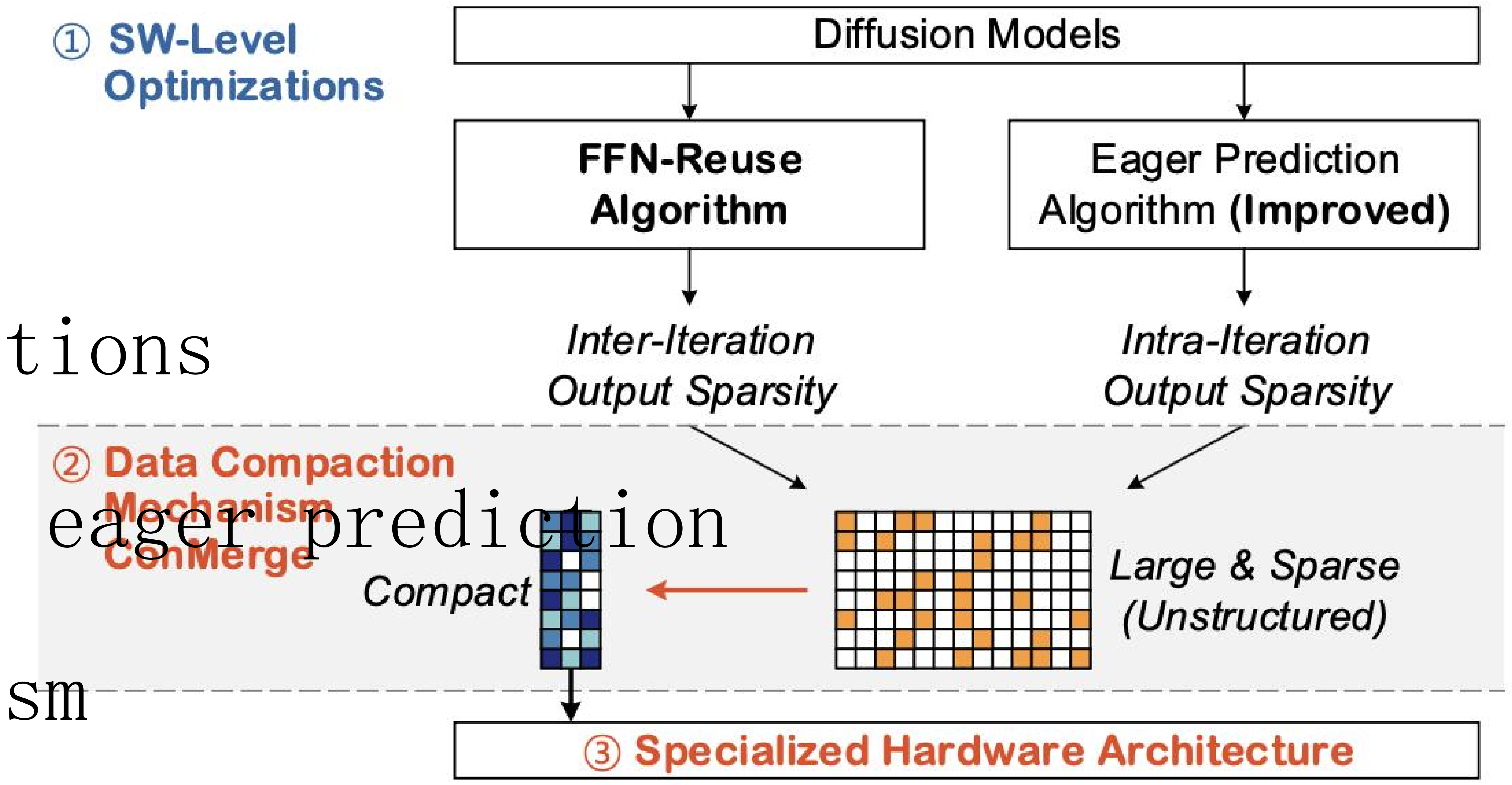
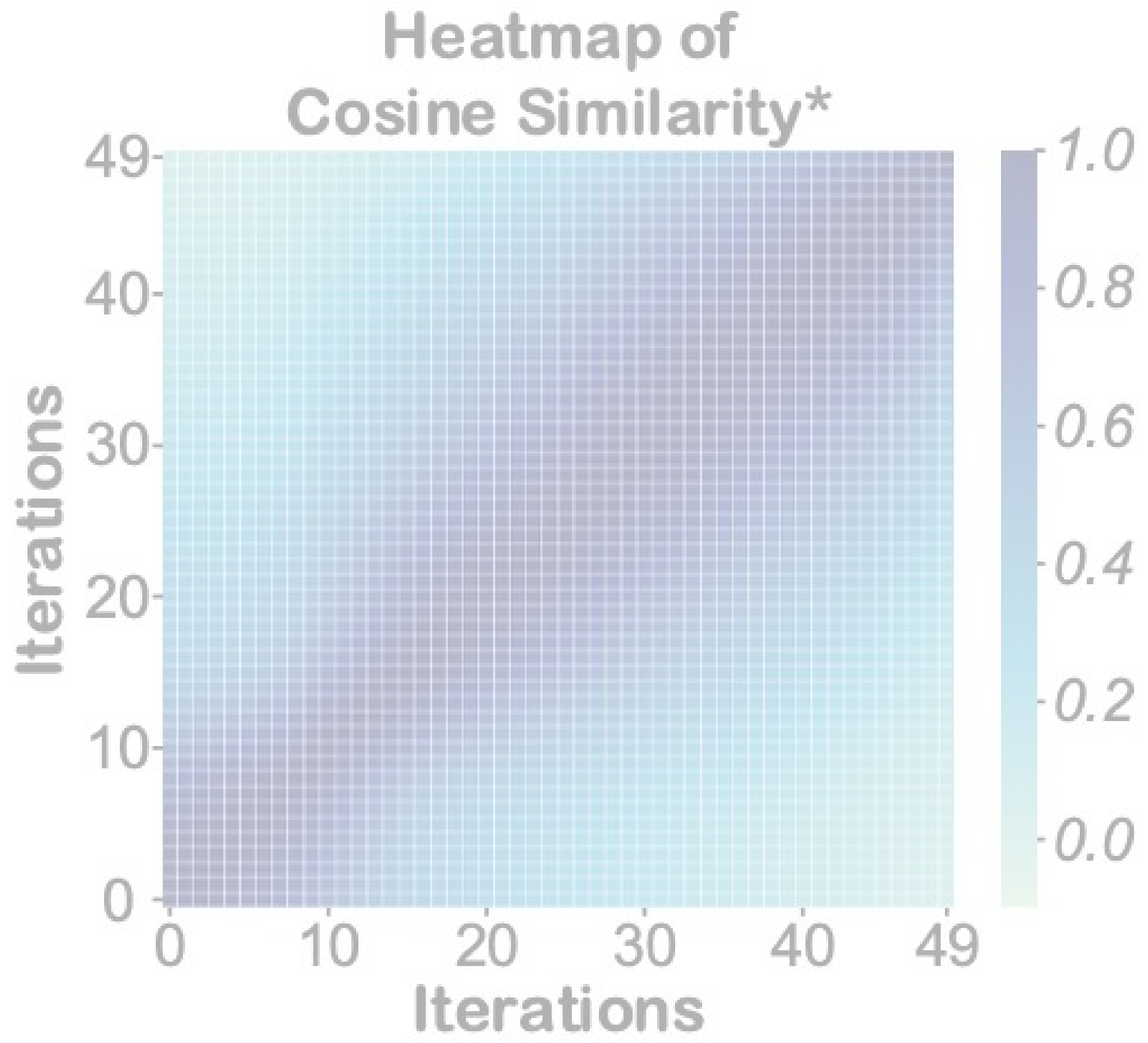


Fig. 1. Overview of EXION Accelerator

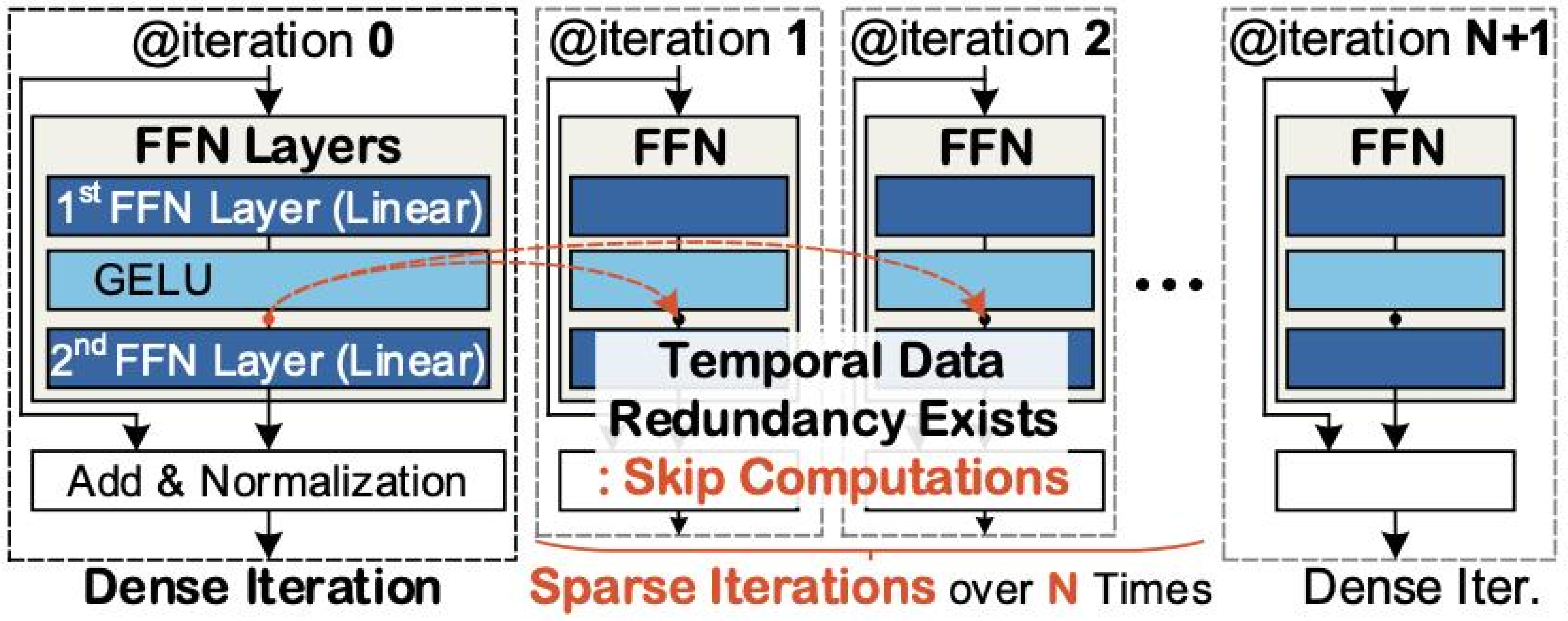
Method

Software optimization 1. FFN-Reuse (inter iteration)

Observation: temporal data redundancy exists across different iterations in



*: Cosine similarity of 2nd block's GELU output in FFN layers across different iterations in DiT model



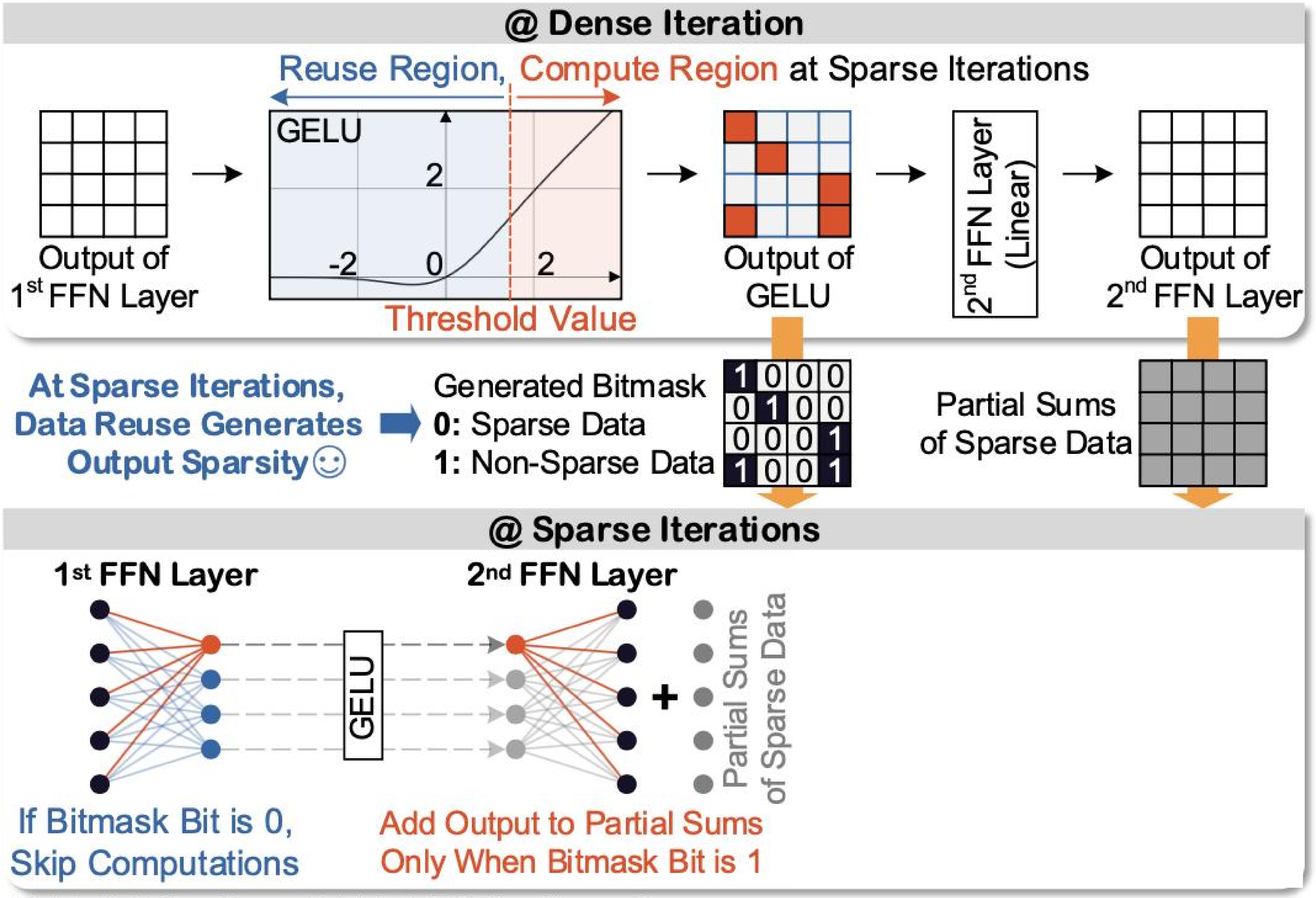
FFN-Reuse Configurations

Models	N
MLD[1]	9
MDM[1]	5
EDGE[1]	5
Make-an-Audio[1]	5
Stable Diffusion[1]	4
VideoCrafter2[1]	3
DiT[2]	2

Method

Software optimization 1. FFN-Reuse (inter iteration)

Fig: FFN-Reuse Algorithm for Inter-iteration Output Sparsity



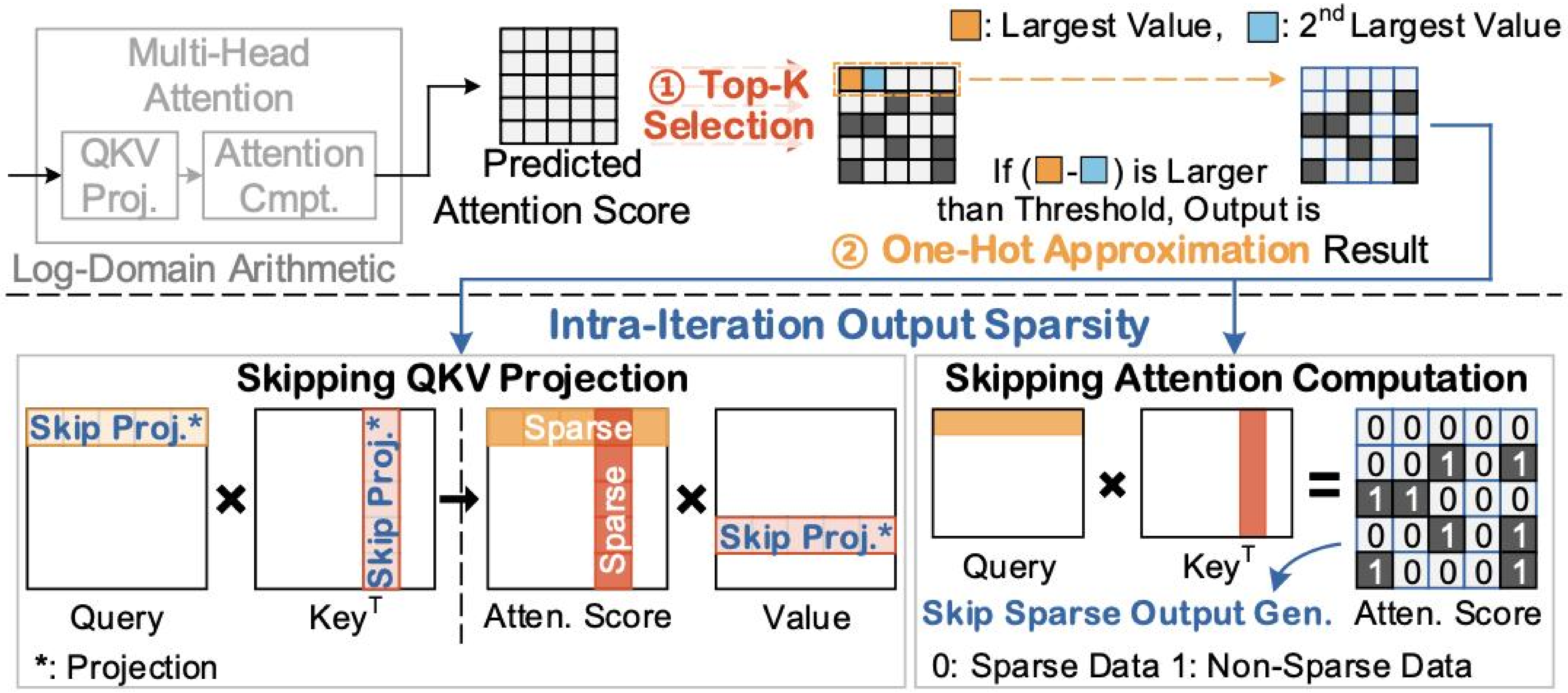
1: Total 50 iterations, *2*: Total 100 iterations, *3*: FFN layers' average reduction percentage

Method

Software optimization 2. Early Prediction (intra iteration)

Goal: predict attention score to skip unnecessary computations

- Skip QKV projection
- Skip attention computation ($Q * K^T$)



Method

Software optimization 2. Early Prediction (intra iteration)

Question: how to estimate the attention early *before* the QKV generation?

FACT: FFN-Attention Co-optimized Transformer Architecture with Eager Correlation Prediction

Yubin Qin*
qyb20@mails.tsinghua.edu.cn
Tsinghua University
Beijing, China

Yang Wang*
wangyang_imec@mail.tsinghua.edu.cn
Tsinghua University
Beijing, China

Dazheng Deng
ddz20@mails.tsinghua.edu.cn
Tsinghua University
Beijing, China

Zhiren Zhao
zhaozr21@mails.tsinghua.edu.cn
Tsinghua University
Beijing, China

Xiaolong Yang
yangxl21@mails.tsinghua.edu.cn
Tsinghua University
Beijing, China

Leibo Liu
liulb@mail.tsinghua.edu.cn
Tsinghua University
Beijing, China

Shaojun Wei
wsj@mail.tsinghua.edu.cn
Tsinghua University
Beijing, China

Yang Hu
hu_yang@mail.tsinghua.edu.cn
Tsinghua University
Beijing, China

Shouyi Yin†
yinsy@mail.tsinghua.edu.cn
Tsinghua University
Beijing, China

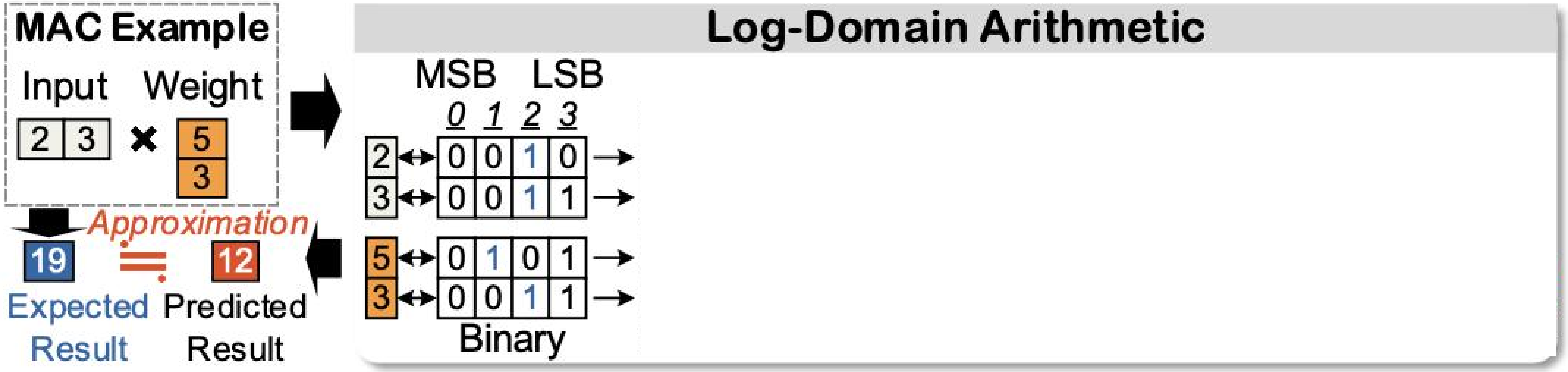
ISCA’ 2023

Method

Software optimization 2. Early Prediction (intra iteration)

Key ideas: use a log-based multiplication-free prediction

- First transforms the data into log-domain by using a *leading-one detector*
- Then substitutes the costly multiplication with low-power shift-and-add op

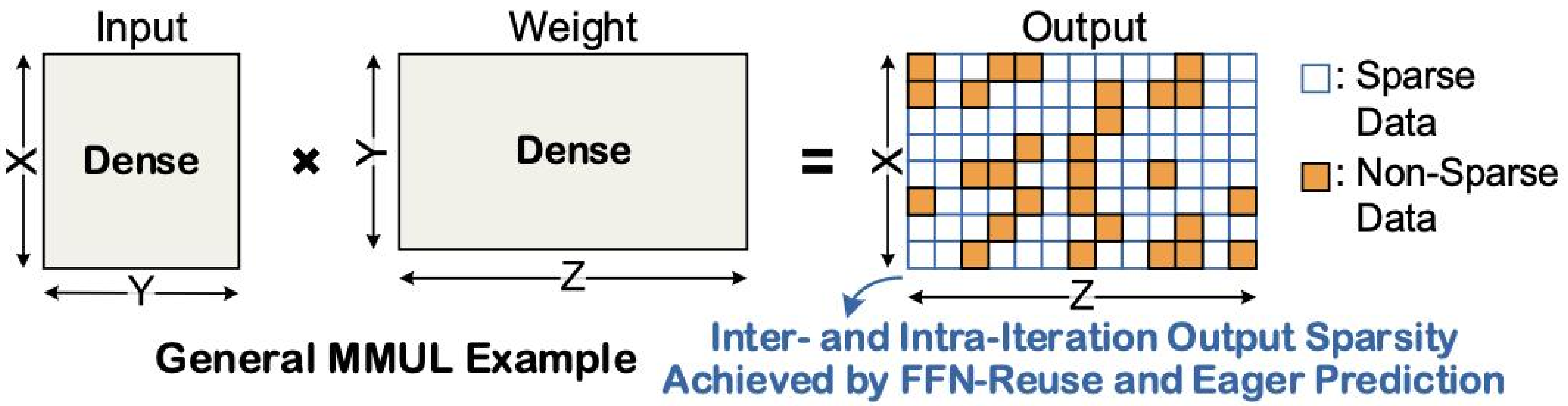


Ps: an INT-type number α can be decomposed as: $\alpha = Sign \times 2^{(W-LO-1)} \times M$

the multiplication of two INT-type numbers: $\alpha \times \beta = XOR(Sign_\alpha, Sign_\beta) \times 2^{(W_\alpha + W_\beta - (LO_\alpha + LO_\beta) - 2)} \times (M_\alpha \times M_\beta)$

Method

Inter- and Intra-Iteration Output Sparsity



Problem: Conventional HW, such as GPUs, cannot utilize it to reduce energy

Method

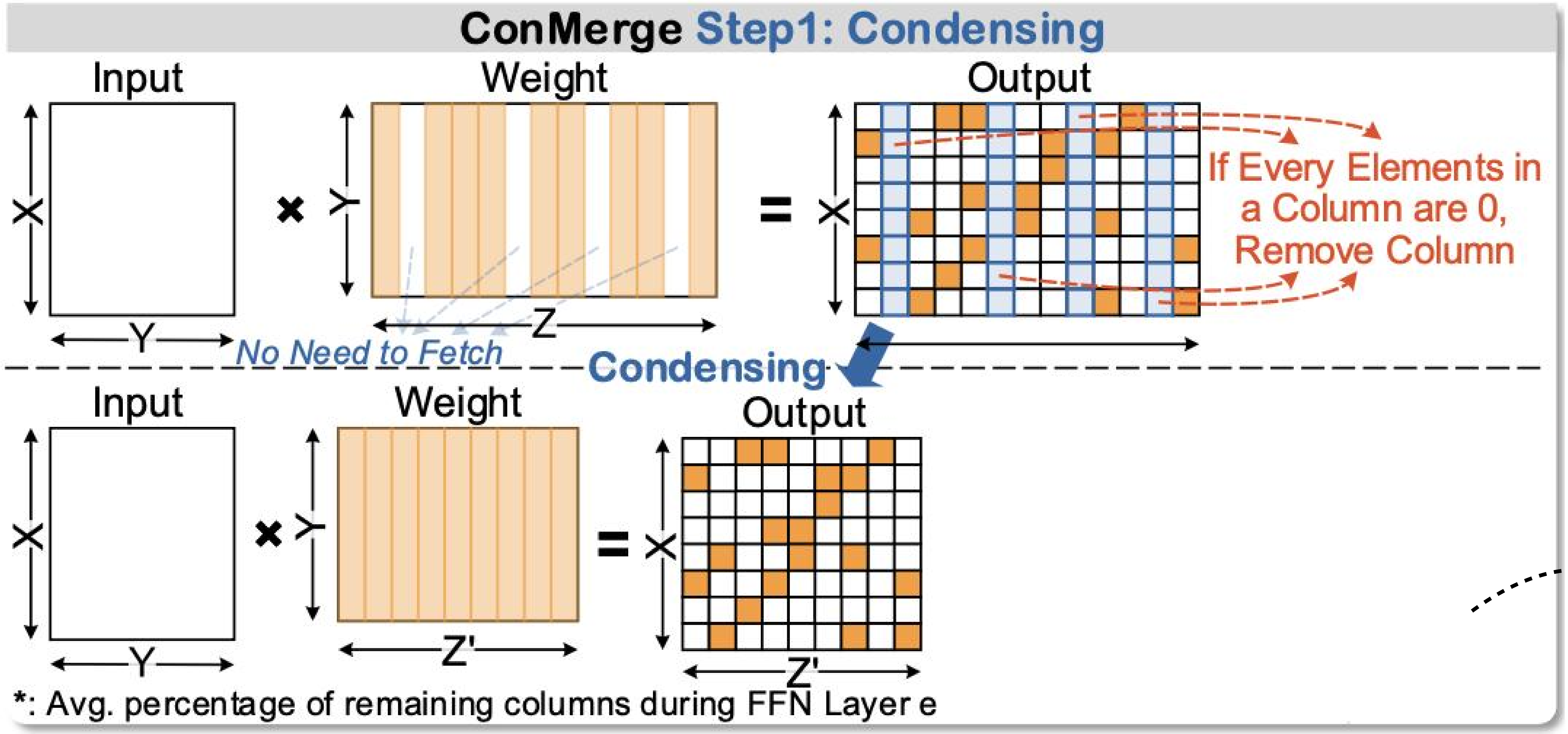
ConMerge: data compaction mechanism

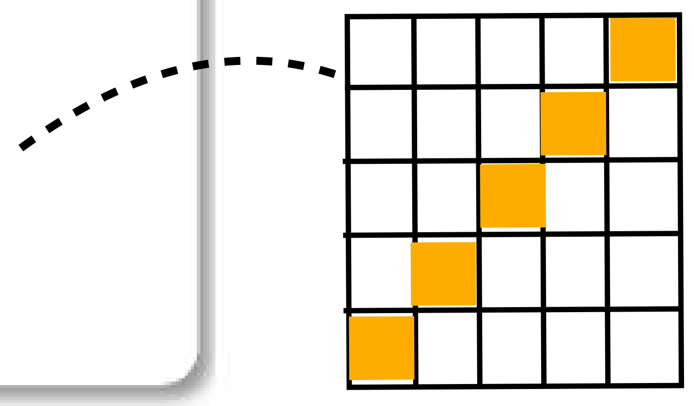
Goal: to **condense** and **merge** large&sparse matrices into small&compact forms

Method

ConMerge: data compaction mechanism

- condense

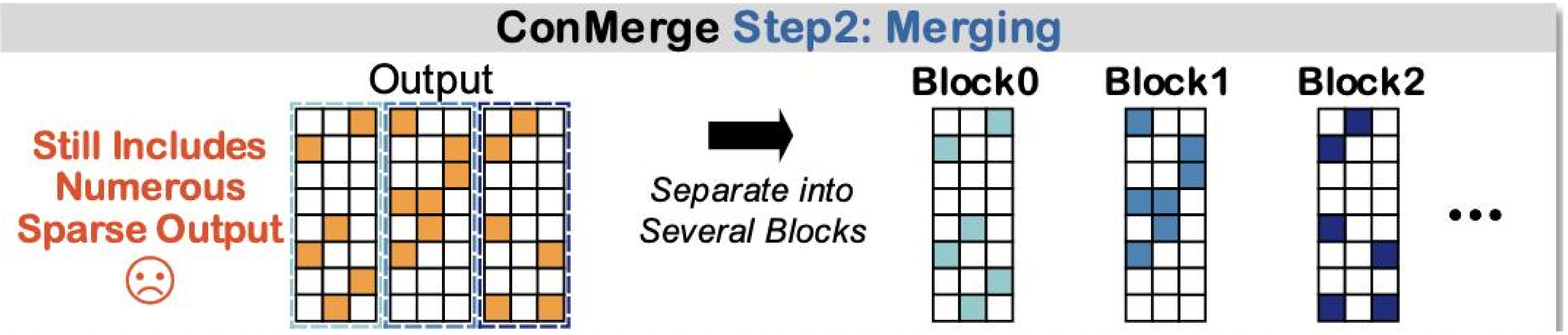




Method

ConMerge: data compaction mechanism

- merge



Method

Hardware co-design: Overview

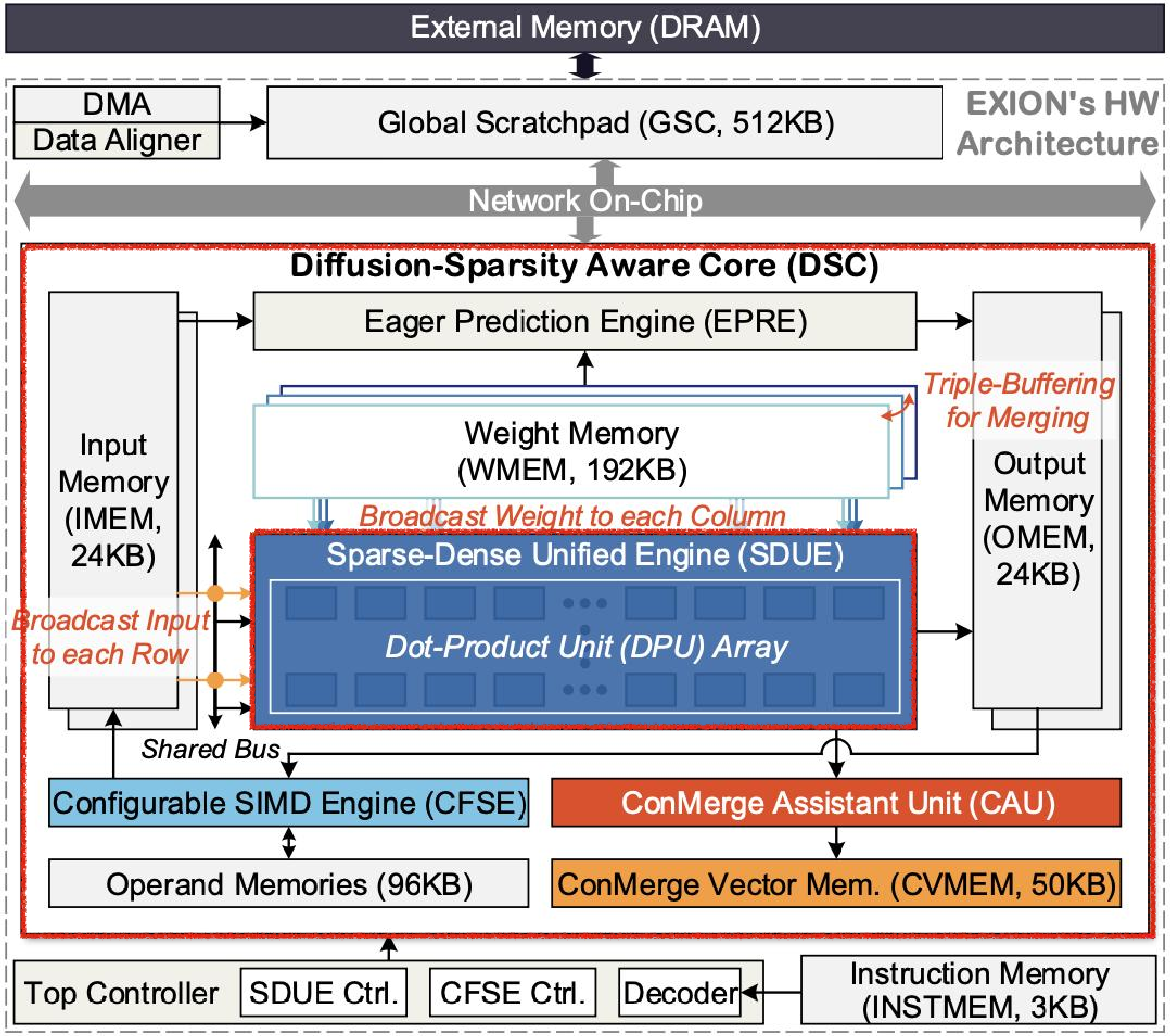
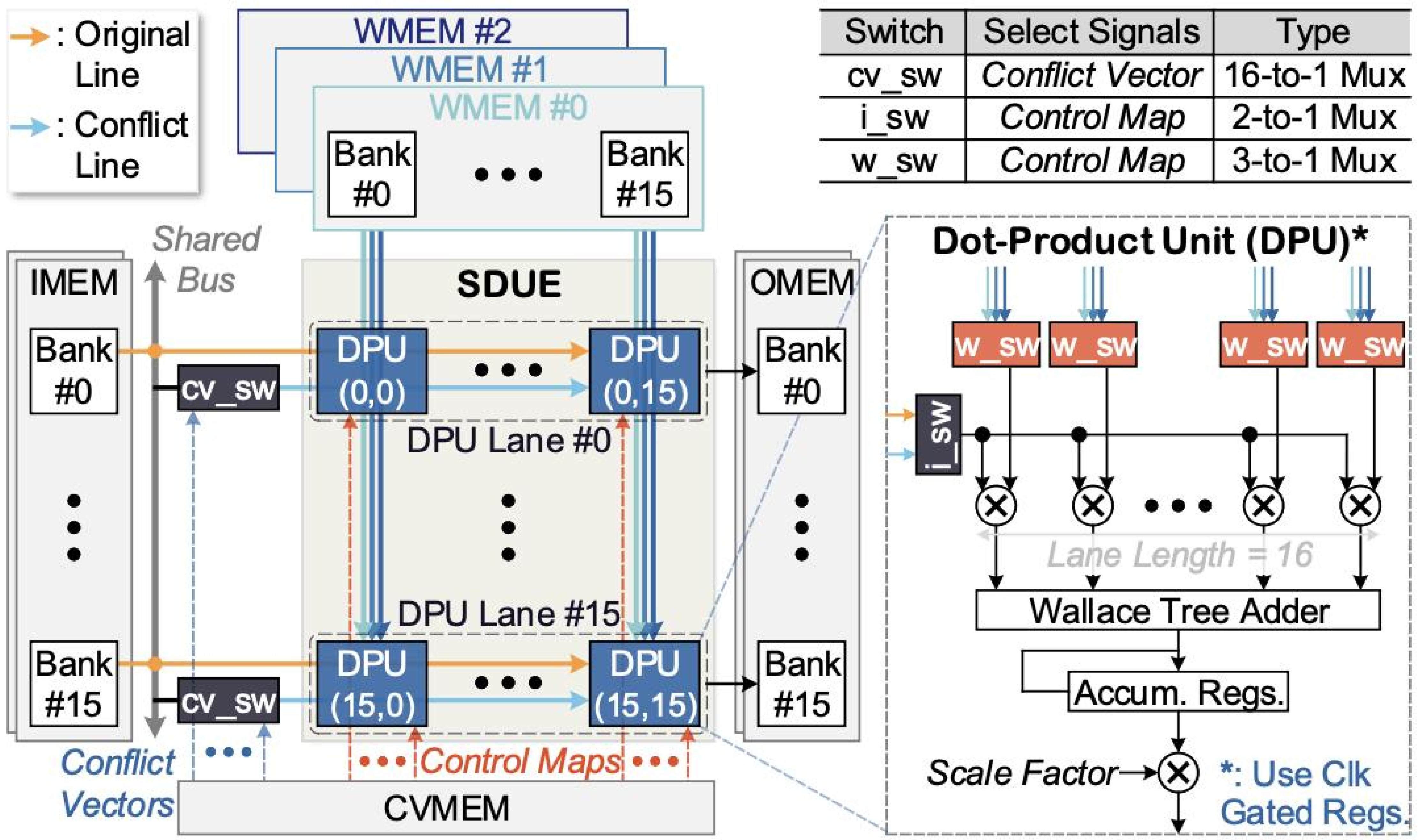


Fig: Hardware architecture overview of EXION

DSC: run the diffusion model’s dense and sparse iterations

Method

Hardware co-design: Sparse-Dense Unified Engine (SDUE)



Switch	Select Signals	Type
cv_sw	*Conflict Vector*	16-to-1 Mux
i_sw	*Control Map*	2-to-1 Mux
w_sw	*Control Map*	3-to-1 Mux

Summary of Hardware Configurations

Toy HW Model (Figure 8-9)		
SDUE		
DPU Array	# of Rows	8
	# of Columns	3
Memory*		
# of Banks	IMEM	8
	WMEM	3
	OMEM	8

EXION Configuration		
SDUE		
DPU Array	# of Rows	16
	# of Columns	16
Memory*		
# of Banks	IMEM	16 (1.5KB per Bank)
	WMEM	16 (12KB per Bank)
	OMEM	16 (1.5KB per Bank)

*: Configuration of a single memory between/among double/triple-buffered memories

cv_switch: conflict vector switch

i_sw: input switch

w_sw: weight switch

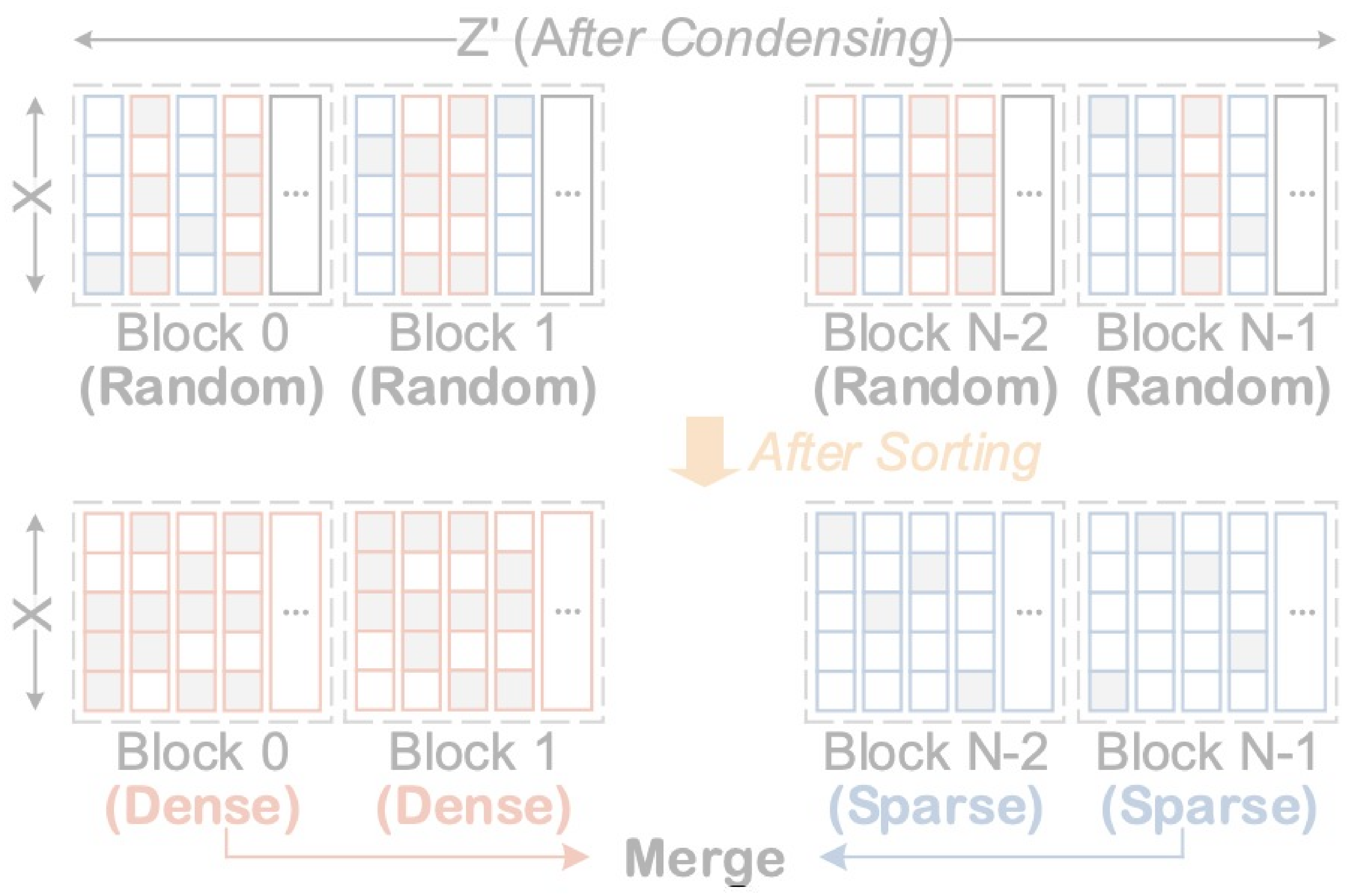
SDUE can compute:

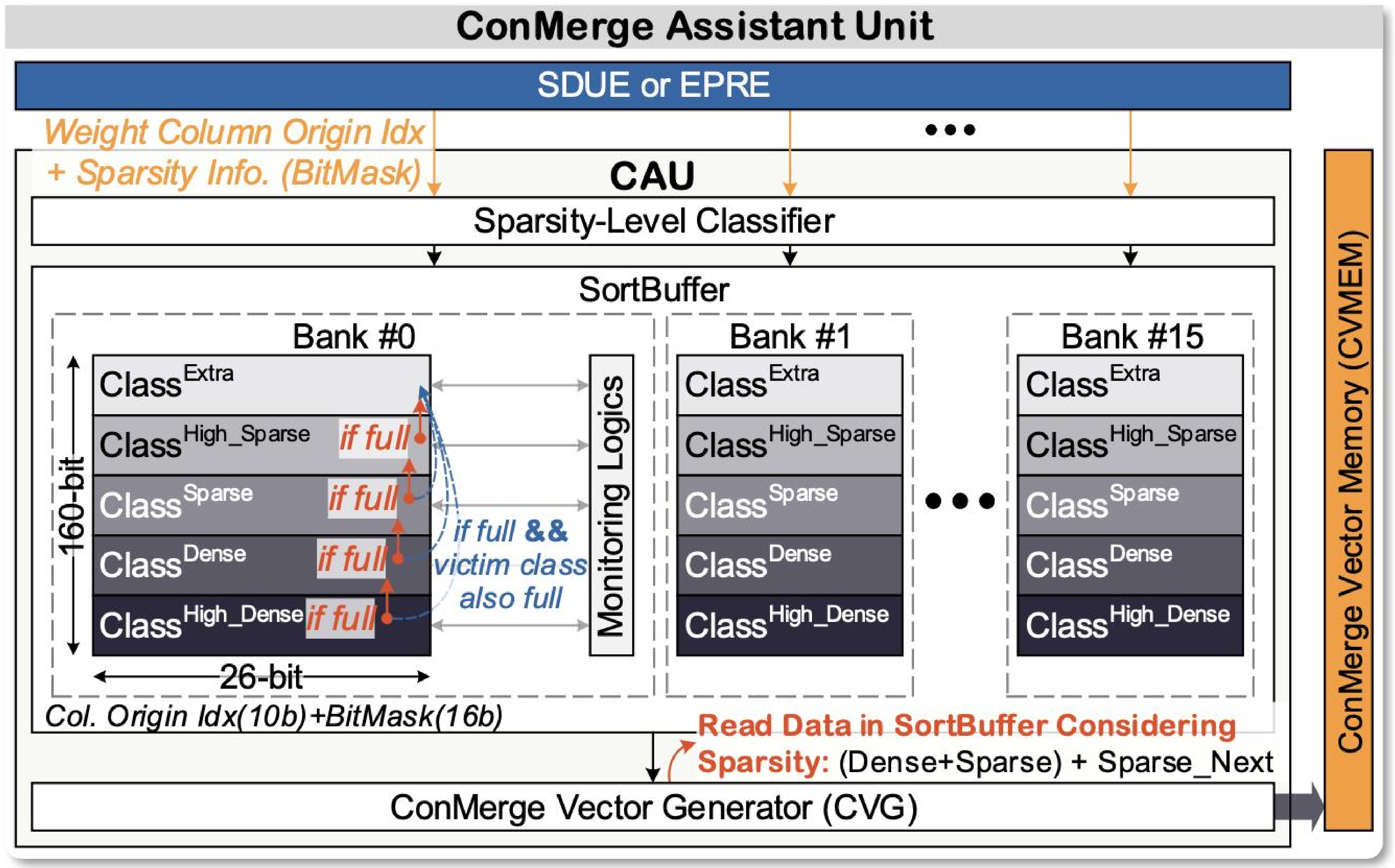
- the normal dense output matrix
- and also the ConMerged block

Method

Hardware co-design: ConMerge Assistant Unit (CAU)

- Sorting strategies for fast merging

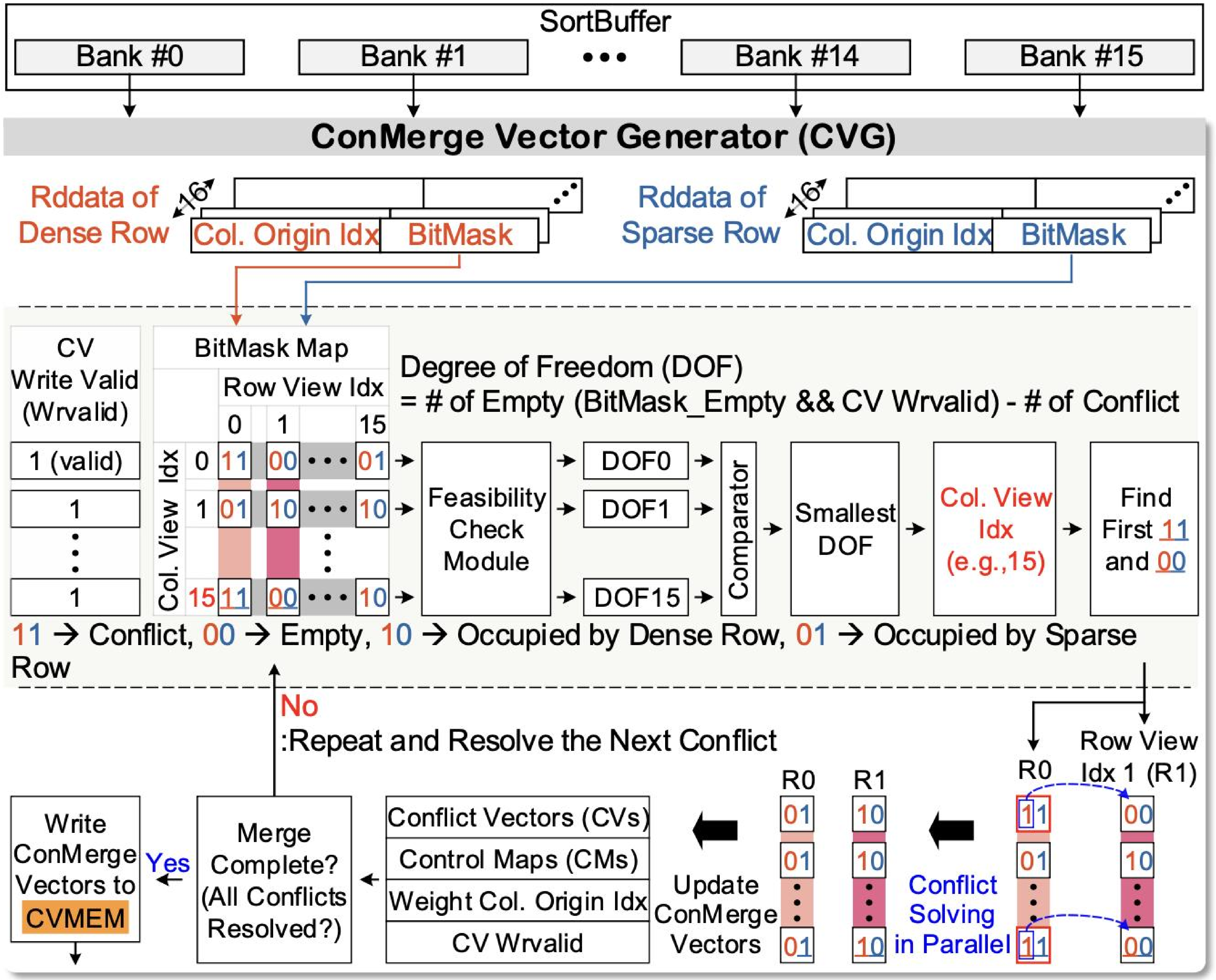




Method

Hardware co-design: ConMerge Assistant Unit (CAU)

- Detailed Merging Process in ConMerge Vector Generator



Evaluation

Experimental setup

- Workloads selected: seven different diffusion models

  text-to-motion (MLD and MDM)

  music-to-motion (EDGE)

  text-to-image (Stable Diffusion)

  class-to-image (DiT)

  text-to-audio (Make-an-Audio)

  text-to-video (VideoCrafter2)

- Hardware specifications of GPUs

	Edge GPU	Server GPU
	NVIDIA Jetson Orin Nano	NVIDIA RTX 6000 Ada
Throughput	40.0 TOPS	91.1 TFLOPS[1]
Memory Bandwidth	68 GB/s	960 GB/s
Power Consumption	-15W	-300W

Evaluation

Accuracy Evaluation

	Models	MLD	MDM	EDGE	Make-an-Audio	Stable Diffusion	DiT	VideoCrafter2
	Task	Text-to-Motion	Text-to-Motion	Music-to-Motion	Text-to-Audio	Text-to-Image	Image Generation	Text-to-Video
	Dataset	HumanML3D	HumanML3D	AIST++	AudioCaps	COCO 2014	ImageNet 2012	ECTV
	Total Iterations	50	50	50	50	50	100	50
Output Sparsity Achieved by EXION's Software-Level Optimizations								
FFN-Reuse	Inter-Iter. Sparsity	95%	95%	95%	97%	97%	80%	70%
FFN-Reuse	N[1]	9	5	5	5	4	2	3
EP[2]	Intra-Iter. Sparsity (q_th[3], k[4])	30% (q_th=0.3, k=0.7)	95% (q_th=0.3, k=0.05)	50% (q_th=0.9, k=0.5)	80% (q_th=0.7, k=0.2)	20% (q_th=0.8, k=0.8)	95% (q_th=0.15, k=0.05)	50% (q_th=2, k=0.5)

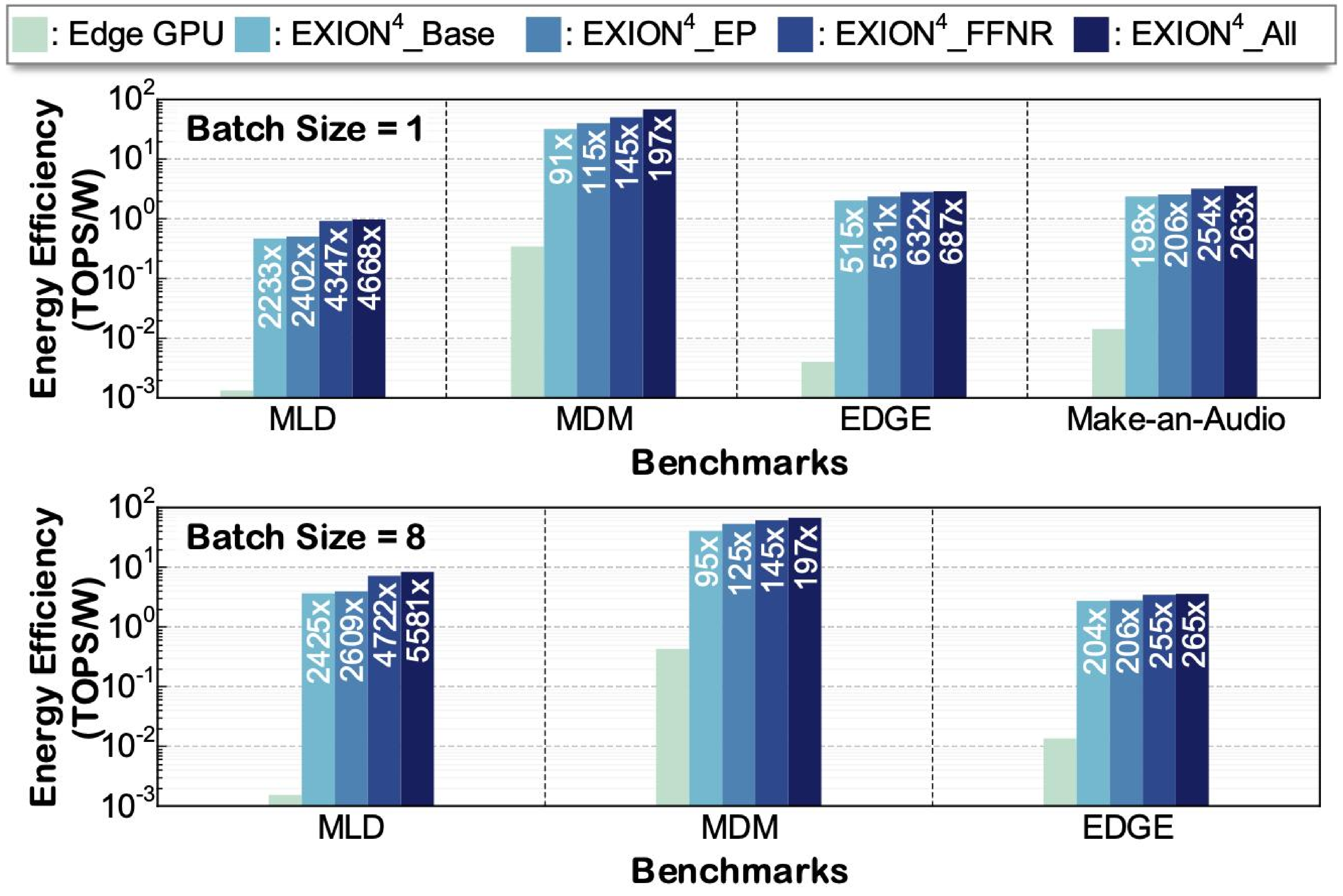
Accuracy Evaluation Metric

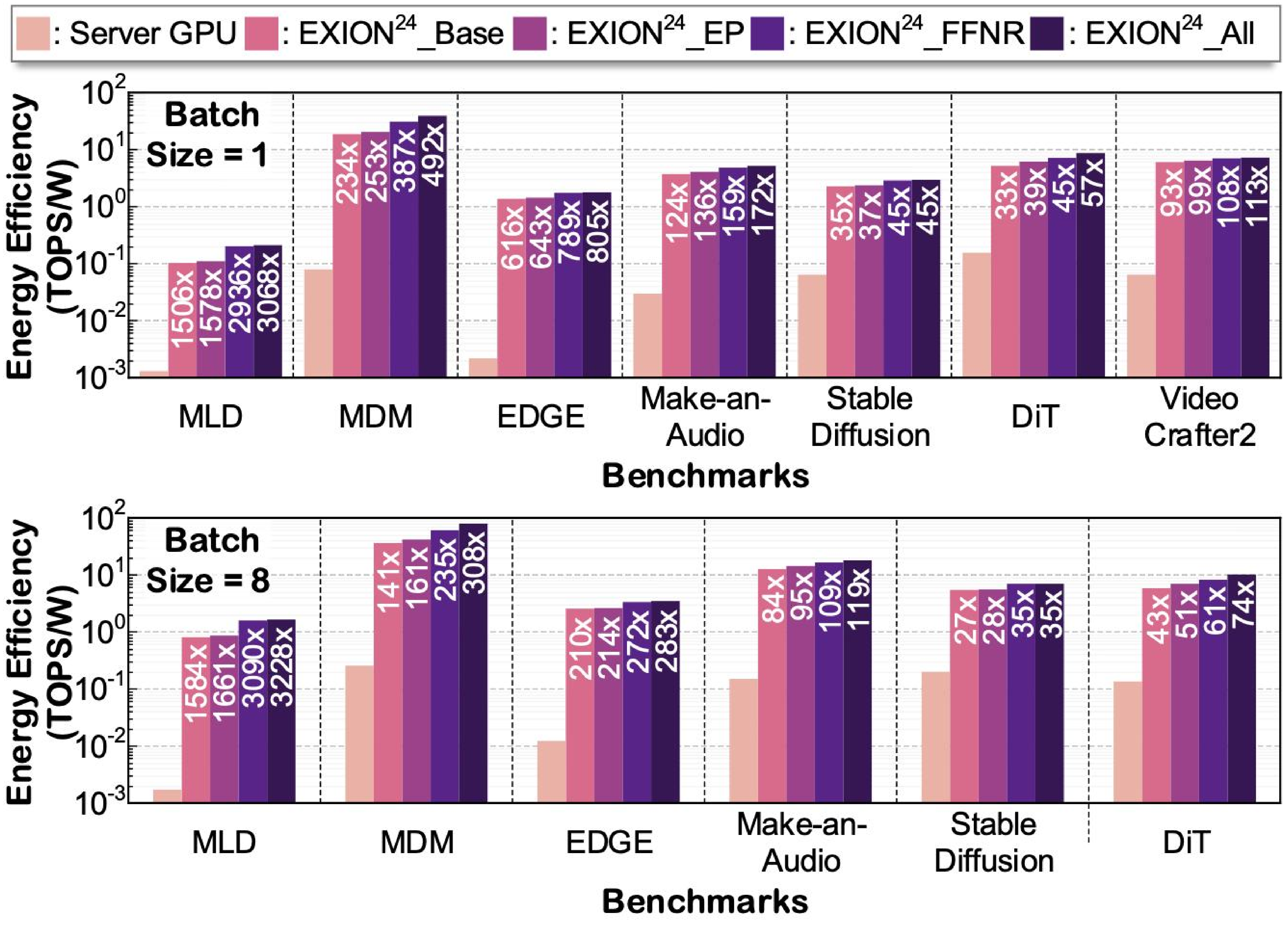
Applied Methods	MLD FID (↓) w/ GT[5]	MLD R-Precision (↑)	MDM FID (↓) w/ GT[5]	MDM PSNR w/ Vanil.[6](↑)	EDGE PFC (↓)	EDGE Beat Align Score (↑)	Make-an-Audio FAD (↓) w/ GT[5]	Make-an-Audio PSNR w/ Vanil.[6](↑)	Stable Diffusion FID (↓) w/ GT[5]	Stable Diffusion IS (↑)	Stable Diffusion PSNR w/ Vanil.[6](↑)	DiT FID (↓) w/ GT[5]	DiT IS (↑)	DiT PSNR w/ Vanil.[6](↑)	VideoCrafter2 VQA_A(↑)	VideoCrafter2 IS (↑)	VideoCrafter2 PSNR w/ Vanil.[6](↑)
Vanilla Model	0.393	0.754	0.406	-	1.352	0.218	4.618	-	26.63	33.11	-	10.63	265.73	-	59.68	17.06	-
FFN-Reuse	0.401	0.752	0.467	19.22	1.389	0.214	4.932	27.14	26.05	33.04	18.20	14.42	265.73	15.99	58.92	16.83	28.25
FFN-Reuse+EP[2]	0.410	0.745	0.968	17.84	2.241	0.195	4.975	26.09	-	-	14.06	-	-	14.60	-	-	27.86
FFN-Reuse+EP[2]+Quant.[7]	0.410	0.744	1.080	17.67	2.411	0.193	5.131	25.72	-	-	13.94	-	-	14.57	-	-	27.56

1: # of sparse iterations between two dense iterations, ***2***: EP w/ TS LOD, ***3***: Threshold for the difference between the largest value and the 2nd largest value, ***4:*** Top-k selection ratio (i.e., k=0.5 selects 50% of the data), ***5***: Ground truth, ***6***: Compute PSNR compared to the vanilla model, ***7:*** After post-training quantization (INT mixed precision, 12b for SDUE/EPRE and 16/32b for CFSE)

Evaluation

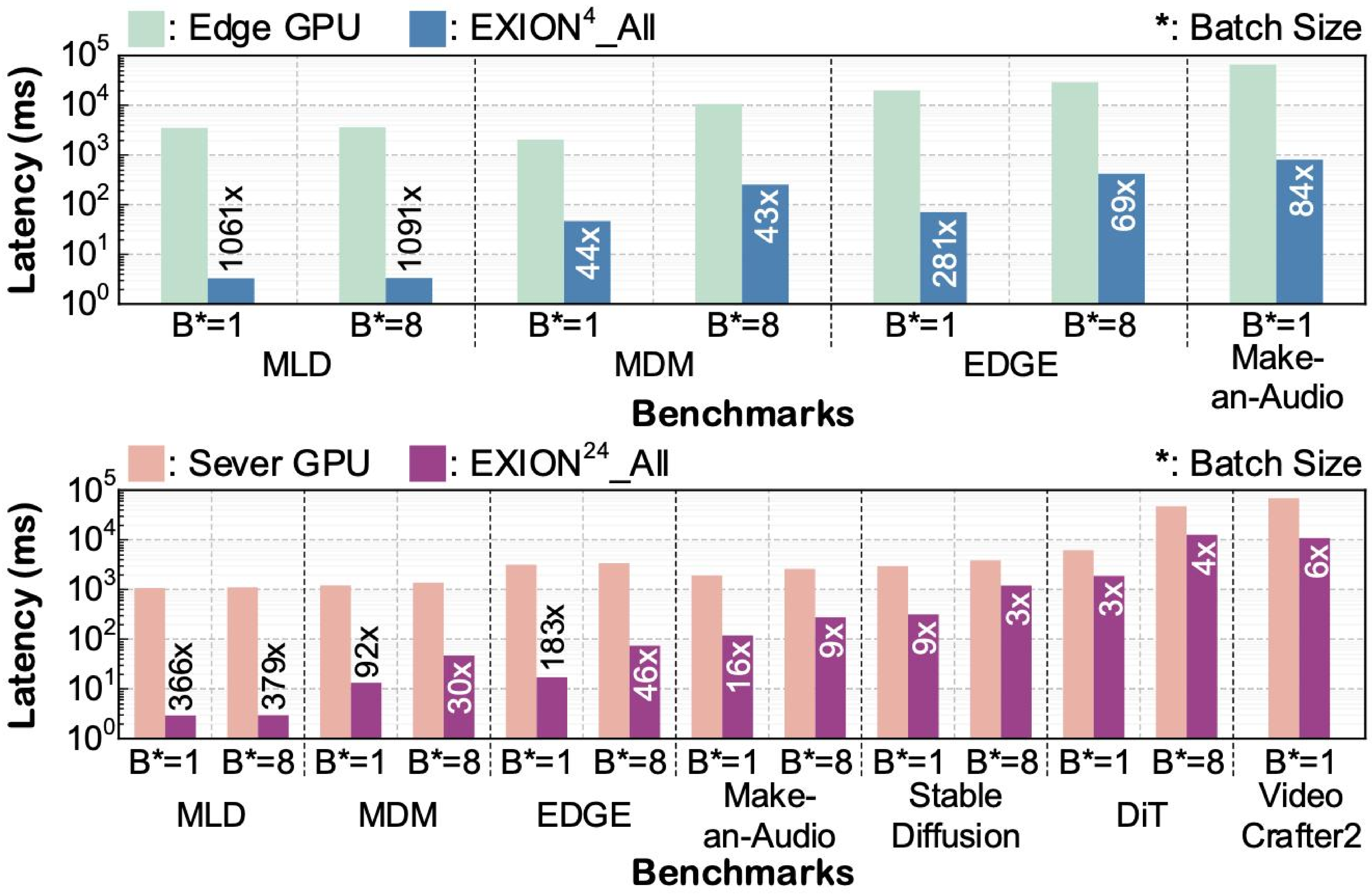
Performance Evaluation: Energy efficiency





Evaluation

Performance Evaluation: Latency



Thanks !

Q&A

Previous Solutions

Software-based approaches

Reduce the large number of iterations (inference steps), e.g. distillation;

Cache and reuse block results;

- Problems: harms accuracy; some of them even require retraining

Hardware accelerators

In each iteration: optimizing QKV projection and attention computation

- Problems: no significantly energy&latency reduce of the overall diffusion