LLM Quantization

Wenxuan Miao 11.22

Outline

- What is Quantization?

- Quantization in LLM

- Current Techniques

- Quantization in Sora

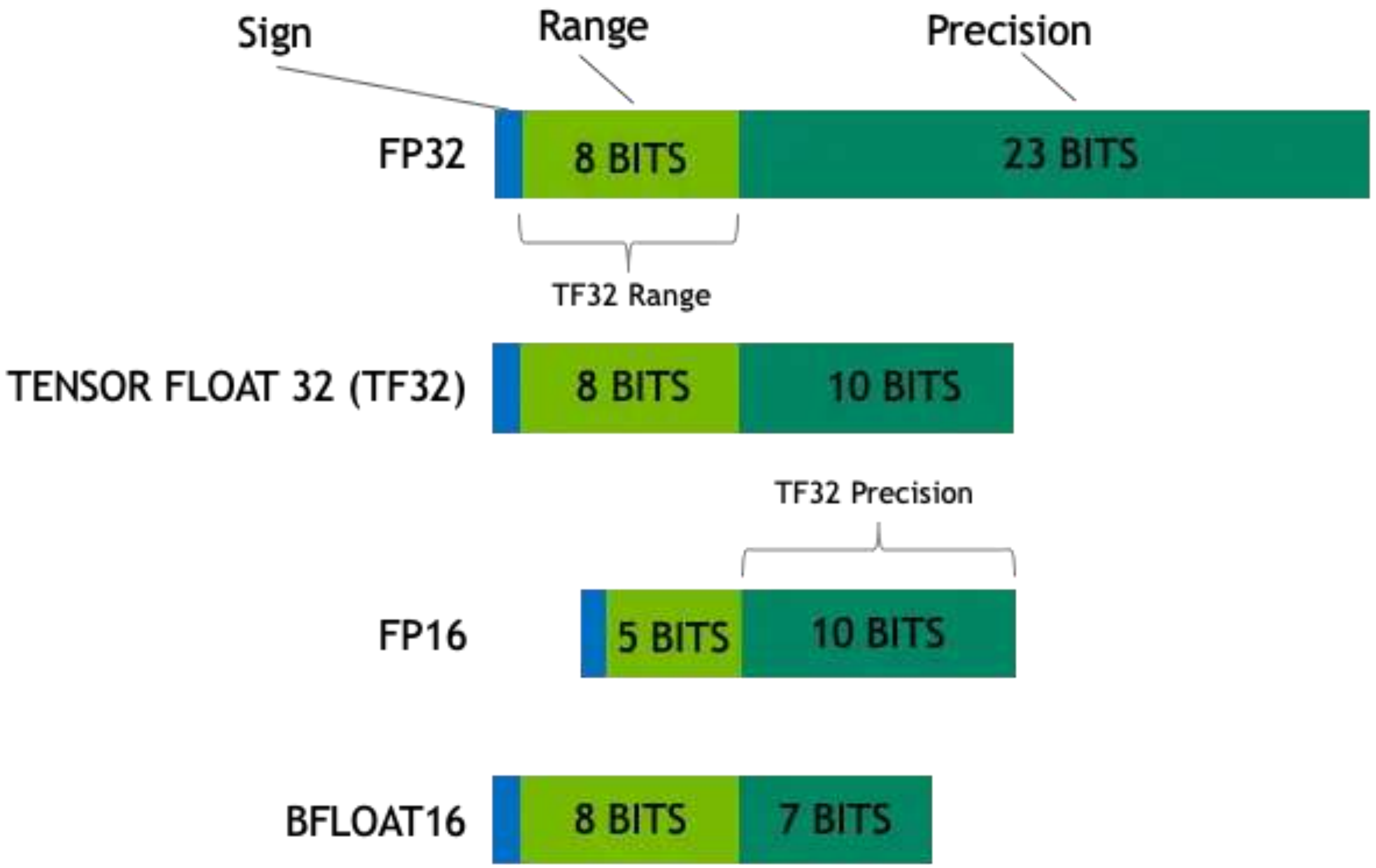
What is Quantization?

What is Quantization?

Why Quantization?

- Quantization compress parameters from high precision (e.g.FP32) to low precision (e.g. INT8) to reduce computational cost and memory usage

- The goal of quantization is to improve inference efficiency and hardware adaptability while maintaining model performance.



What is Quantization?

How to Quantization?

- Asymmetric Quantization

$$\mathbf{Q}_{\mathbf{X}} = \left\lceil \frac{\mathbf{X}}{s} + z \right\rfloor, s = \frac{\mathbf{X}_{\max} - \mathbf{X}_{\min}}{q_{\max} - q_{\min}}, z = \left\lceil q_{\min} - \frac{\mathbf{X}_{\min}}{s} \right\rfloor,$$

- dequantization

$$\hat{\mathbf{X}} = Q(\mathbf{X}) = (\mathbf{Q}_{\mathbf{X}} - z) \cdot s$$

Quantization in LLM

Quantization in LLM

PTQ and QAT

Feature	PTQ (Post-Training Quantization)	QAT (Quantization-Aware Training)
Workflow	Quantize the model after training	Simulate quantization during training
Applicability	Fast and applicable to most pre-trained models	Achieves high accuracy but requires retraining
Impact on Accuracy	May result in significant accuracy degradation	Maintains high accuracy by optimizing during training
Computational Cost	Low, no extra training required	High, involves retraining with quantization

Quantization in LLM

W and A

W (Weights)

- Fixed parameters in the model, such as W_q, W_k, W_v in attention layers and weights in feed-forward networks (FFN).
- Do not change with input data; responsible for mapping and transformations.

A (Activations)

- Intermediate computed results, such as Q, K, V matrices and attention outputs.
- Dynamic, dependent on input data.

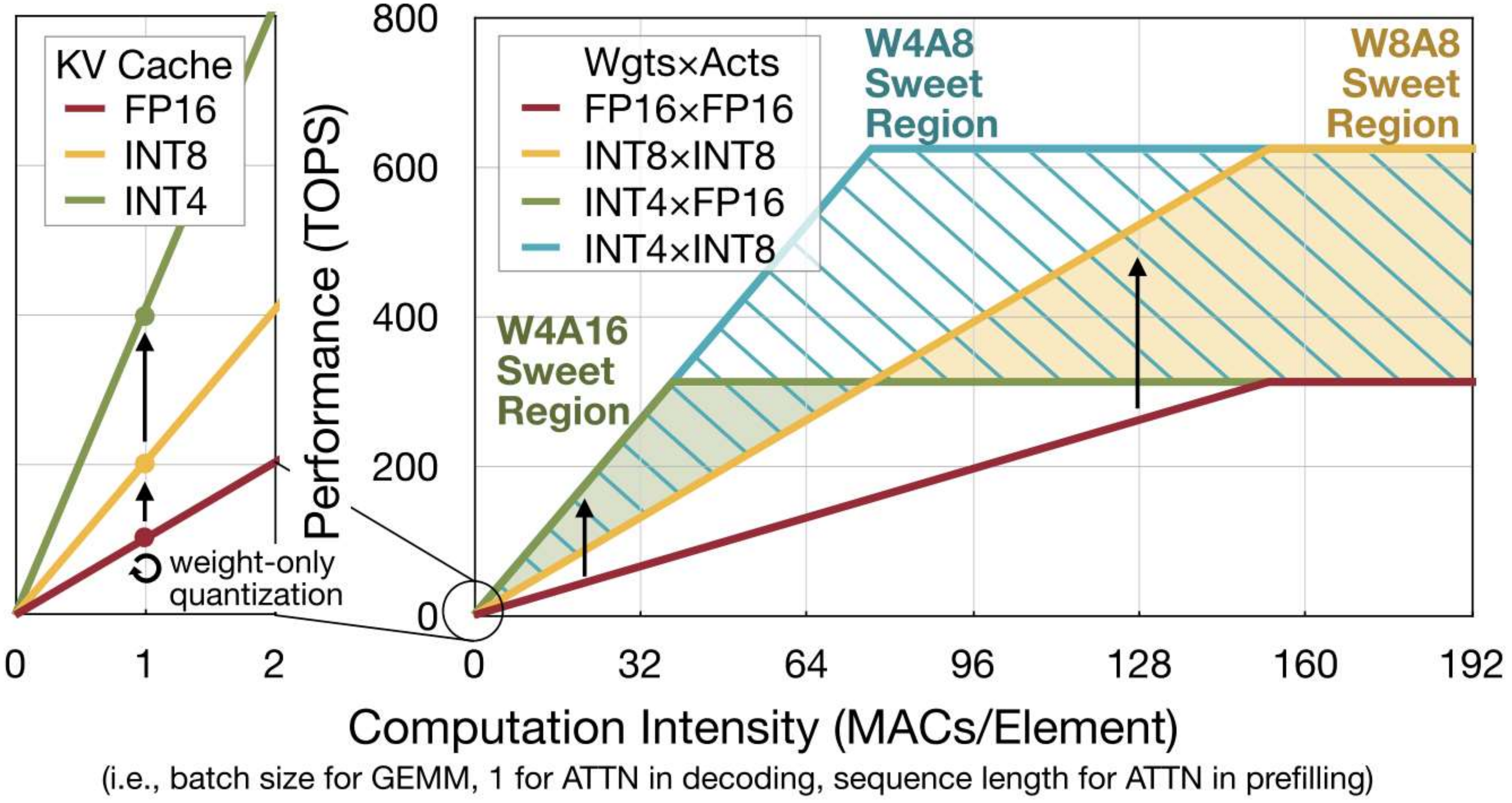
W: Fixed parameters, major focus on storage optimization.

A: Dynamic values, key to computtaion efficiency.

Quantization in LLM

W8A8 and W4A16

- W8A8: Both activation and weights are quantized to INT8.
  - LLM.int8(), smoothQuant
- W4A16: Low-Bit Weight only quantization
  - AWQ, GPTQ

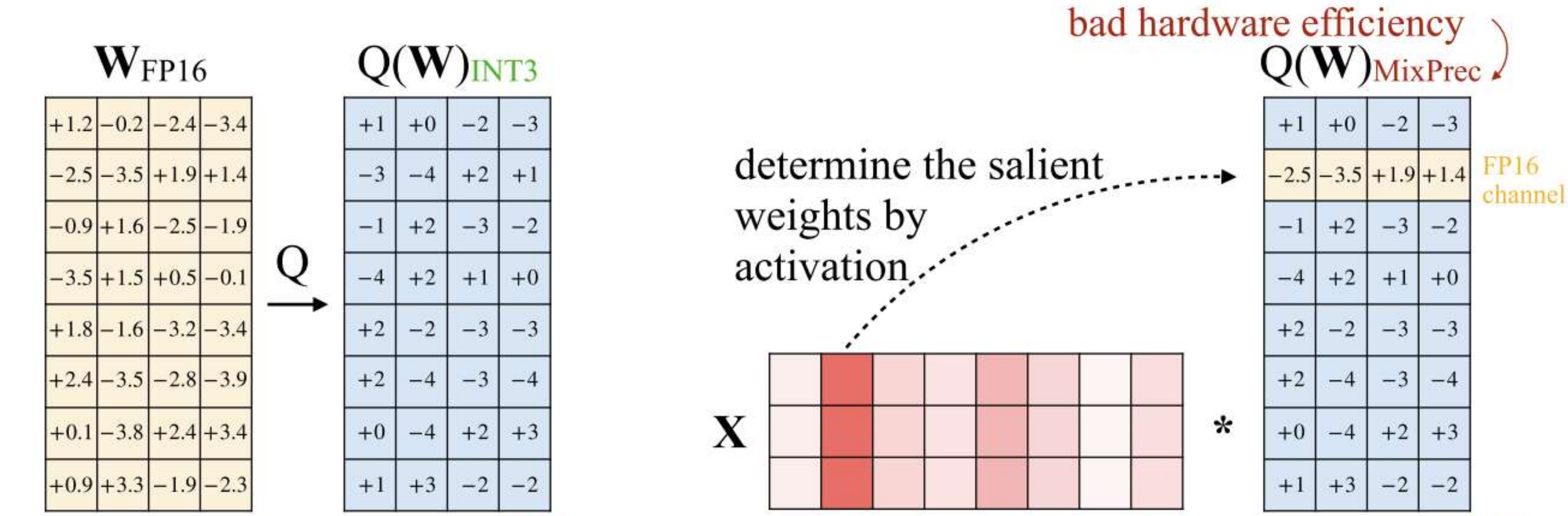


说明

Current Techniques

Current Techniques

Activation-Aware Weight Quantization(AWQ)



(a) RTN quantization (**PPL** 43.2) (b) Keep 1% salient weights in FP16 (**PPL** 13.0)

Current Techniques

Activation-Aware Weight Quantization(AWQ)

- Keeping a small fraction of weights (0.1%-1%) in FP16 significantly improves the performance of the quantized models.
- Selecting weights based on activation magnitude or L2-norm can significantly improve the performance

PPL ↓	FP16	RTN (w3-g128)	FP16% (based on act.)			FP16% (based on W)			FP16% (random)		
			0.1%	1%	3%	0.1%	1%	3%	0.1%	1%	3%
OPT-1.3B	14.62	119.00	25.03	16.91	16.68	108.71	98.55	98.08	119.76	109.38	61.49
OPT-6.7B	10.86	23.54	11.58	11.39	11.36	23.41	22.37	22.45	23.54	24.23	24.22
OPT-13B	10.13	46.04	10.51	10.43	10.42	46.07	48.96	54.49	44.87	42.00	39.71

Current Techniques

Quantization error in AWQ

If we scale the parameter per-channel?

$$Q(\mathbf{w}) = \Delta \cdot \mathrm{Round}(\frac{\mathbf{w}}{\Delta}), \quad \Delta = \frac{\max(|\mathbf{w}|)}{2^{N-1}},$$

$$Q(w \cdot s) \cdot \frac{x}{s} = \Delta^{'} \cdot \mathrm{Round}(\frac{ws}{\Delta^{'}}) \cdot x \cdot \frac{1}{s},$$

Scale is done per-channel, Δ' may not increase if s is not very large

So the quantization error will be:

$$\texttt{Err}(Q(w)x) = \Delta \cdot \texttt{RoundErr}(\frac{w}{\Delta}) \cdot x$$

$$\texttt{Err}(Q(w \cdot s)(\frac{x}{s})) = \Delta^{'} \cdot \texttt{RoundErr}(\frac{ws}{\Delta^{'}}) \cdot x \cdot \frac{1}{s}$$

Current Techniques

Algorithm in AWQ

- So we use scale based method to avoid mixed-precision to be more hardware-friendly. But how to train the scale factor?
- Here is the “Activation-aware”:

$$\mathbf{s} = \mathbf{s_X}^{\alpha}, \quad \alpha^* = \arg\min_{\alpha} \mathcal{L}(\mathbf{s_X}^{\alpha})$$

- s_X is the average magnitude of activation(per channel). Only one single hyper-parameter α need to be trained

Current Techniques

Smooth Quant

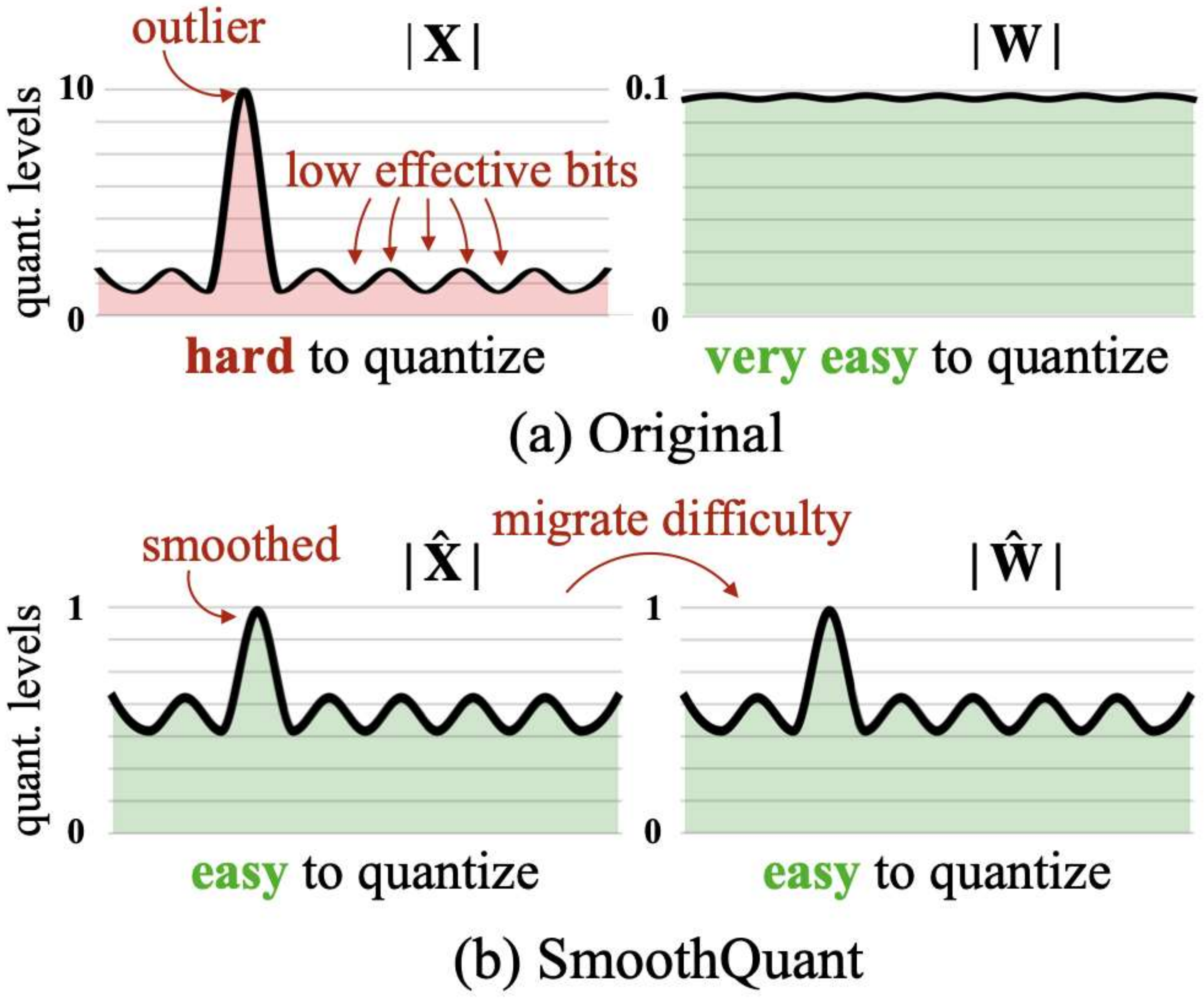
- Outliers lies in activation mainly.

$$\mathbf{Y} = (\mathbf{X}\mathrm{diag}(\mathbf{s})^{-1}) \cdot (\mathrm{diag}(\mathbf{s})\mathbf{W}) = \hat{\mathbf{X}}\hat{\mathbf{W}}$$

- Migrate the scale variance from activations to weights.

$$\mathbf{s}_j = \max(|\mathbf{X}_j|)^{\alpha} / \max(|\mathbf{W}_j|)^{1-\alpha}$$

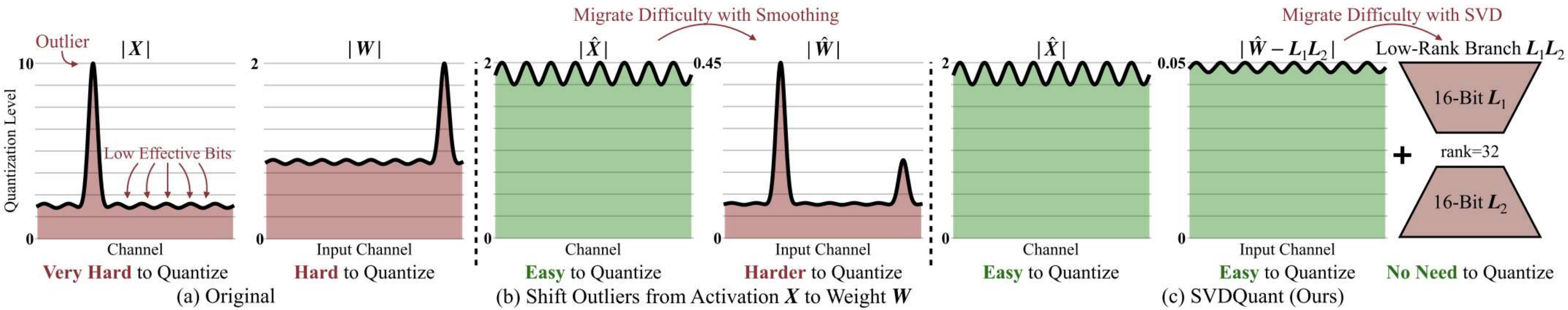
- j is the input channel



Current Techniques

SVDQuant

- If both activations and weights are hard to quantize:



- SVDQuant introduce Singular Value Decomposition to deal with W

Current Techniques

SVDQuant

Singular Value Decomposition (SVD) is a mathematical technique that factorizes a m×n matrix into three component matrices:

$$A = U\Sigma V^T$$

U: $m \times m$ orthogonal matrix

Σ: $m \times n$ diagonal matrix, containing the singular values in descending order.

V^T:$n \times n$ orthogonal matrix

Current Techniques

SVDQuant

$\hat{W}$ is the weight after smooth quant($m \times n$).

Assume $\hat{W} = U\Sigma V$

If $L_1 = U\Sigma_{[:,:r]}$, $L_2 = V_{[:r]}$, $\hat{W}$ can be approximated as $L_1 L_2 + R$. $L_1 L_2$ are low-rank branch since $r \ll min\{m, n\}$

Current Techniques

SVDQuant Algorithm

$\hat{W} = L_1 L_2 + R$, where $L_1 \in R^{m \times r}, L_2 \in R^{r \times n}$.

$$\boldsymbol{X}\boldsymbol{W} = \hat{\boldsymbol{X}}\hat{\boldsymbol{W}} = \hat{\boldsymbol{X}}\boldsymbol{L}_1\boldsymbol{L}_2 + \hat{\boldsymbol{X}}\boldsymbol{R} \approx \underbrace{\hat{\boldsymbol{X}}\boldsymbol{L}_1\boldsymbol{L}_2}_{\text{16-bit low-rank branch}} + \underbrace{Q(\hat{\boldsymbol{X}})Q(\boldsymbol{R})}_{\text{4-bit residual}}.$$

It can be proved that quantization can be bounded by the magnitude of R since X is already smoothed.

$$\mathbb{E}\left[\|\boldsymbol{R} - Q(\boldsymbol{R})\|_F\right] \leq \frac{\sqrt{\log\left(size(\boldsymbol{R})\right)\pi}}{q_{\max}} \mathbb{E}\left[\|\boldsymbol{R}\|_F\right],$$

Current Techniques

Error of SVDQuant

- $\| R \|_F = \| \hat{W} - L_1 L_2 \|_F = \sqrt{\sum_{i=r+1}^{min(m,n)} \sigma_i^2}$,

  where σ_i is the i-th singular value of $\hat{W}$

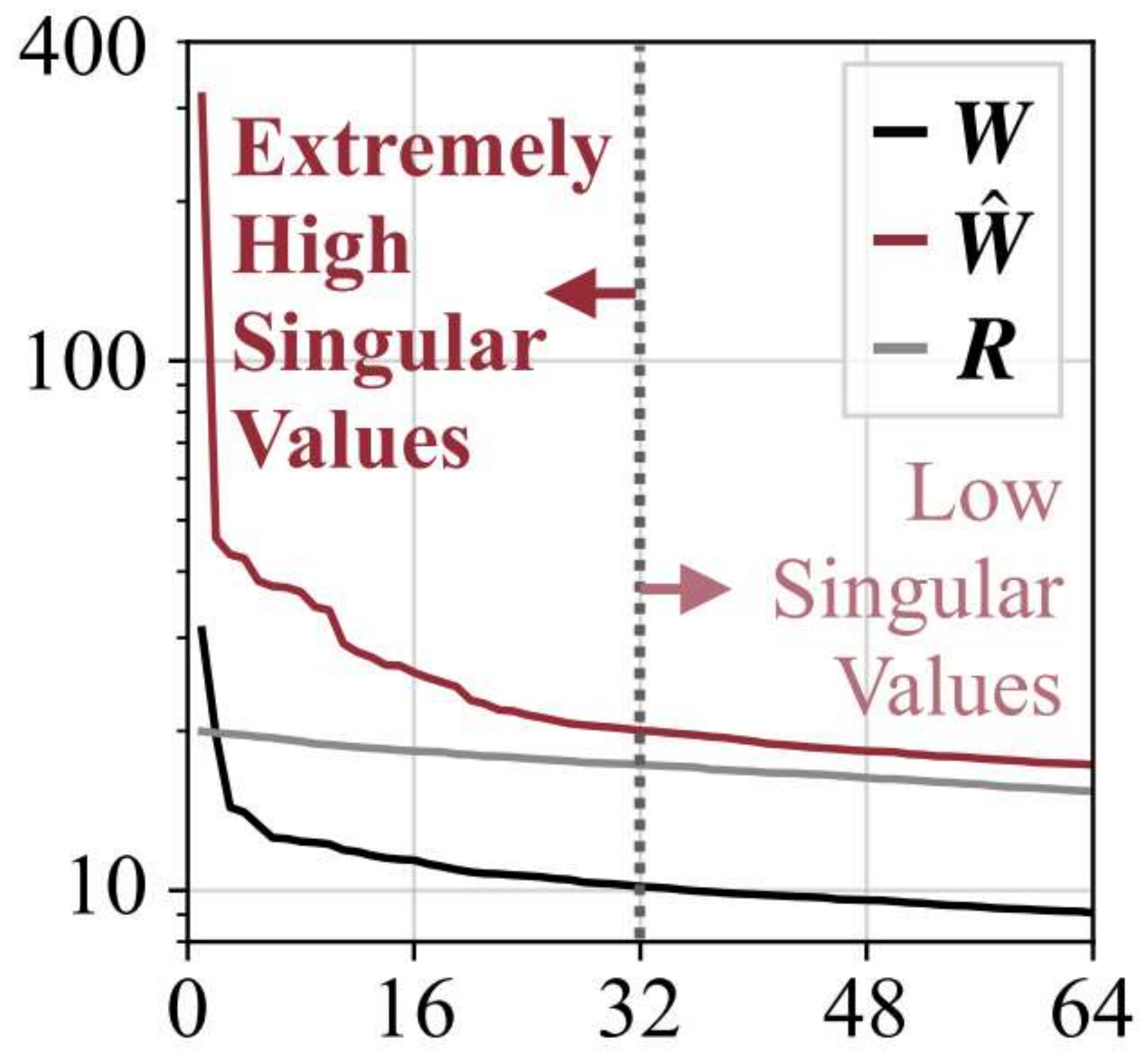
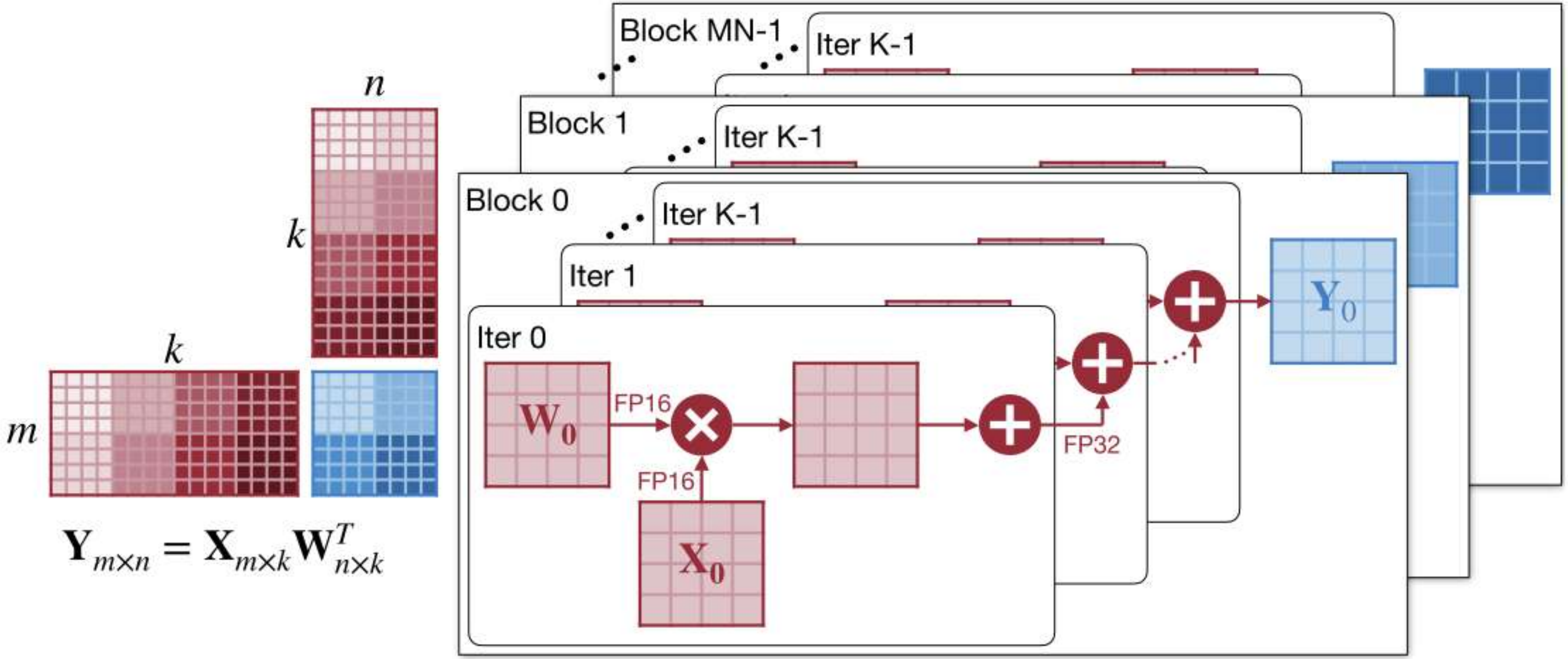


Figure 5: First 64 singular values of $\boldsymbol{W}$, $\hat{\boldsymbol{W}}$, and $\boldsymbol{R}$. The first 32 singular values of $\hat{\boldsymbol{W}}$ exhibit a steep drop, while the remaining values are much more gradual.

Qserve

GEMM on GPU



The matrix is divided in several tiles and computed in a sequential loop.

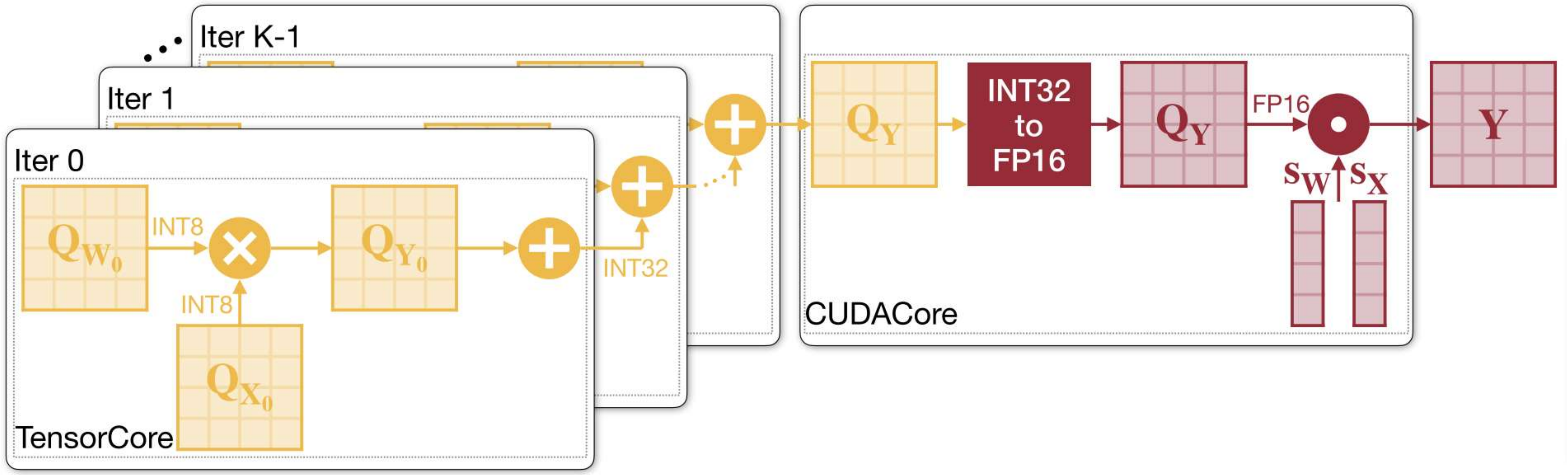
Qserve

W8A8 vs W4A4

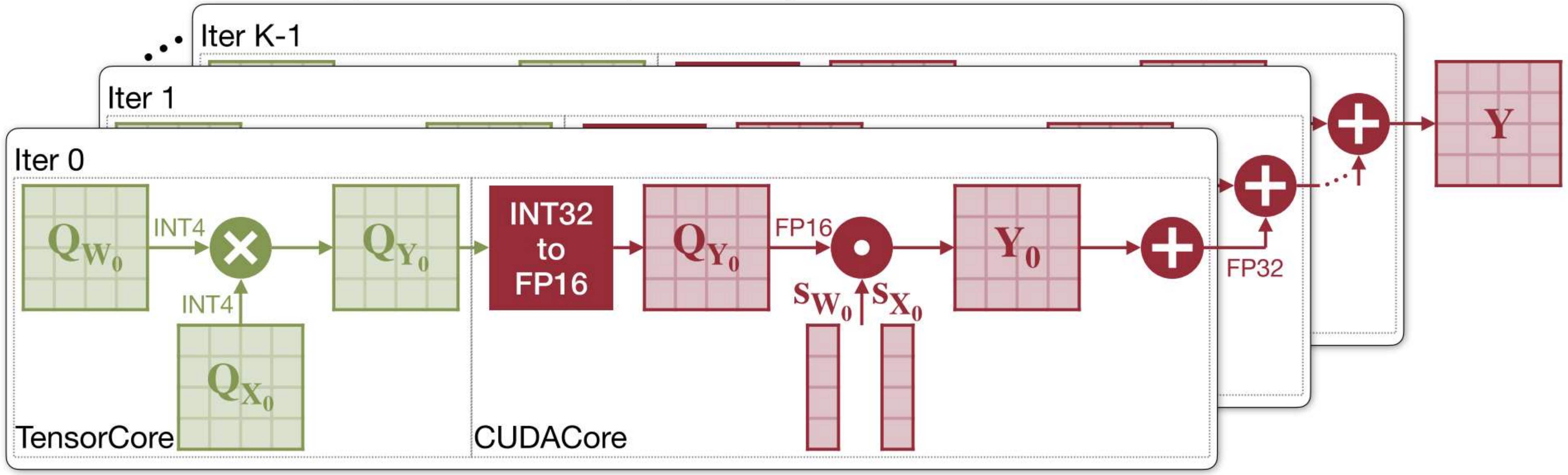
Compare W8A8 and W4A4:
In W8A8, dequantization is performed in the accumulation.

However, W4A4 performs dequantization in the loop.CUDACore is about 2% the performance of tensor core.

So, W4A4 is even slower than W8A8 due to dequantization overhead.



(a) TensorRT-LLM (INT8 Weights and INT8 Activations)



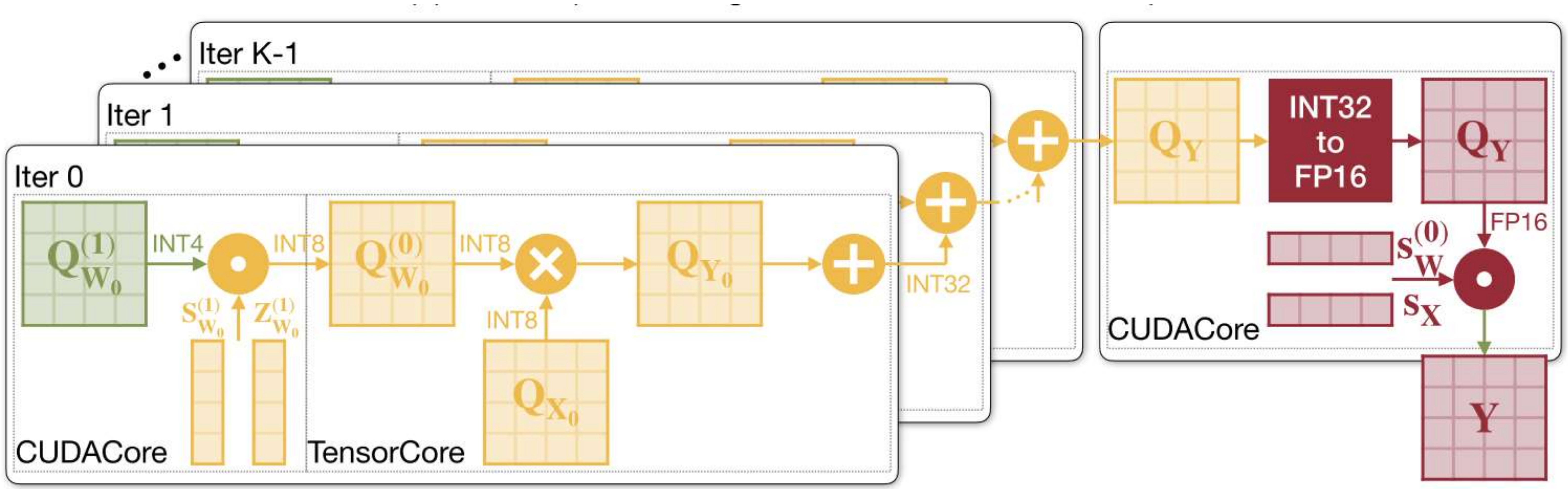
(c) ATOM (INT4 Weights and INT4 Activations)

Qserve

Progressive Quantization (W4A8KV4)

In the algorithm, Qserve introduce progressive group quantization.

This approach ensures that all GEMMs are performed on INT8 tensor cores.



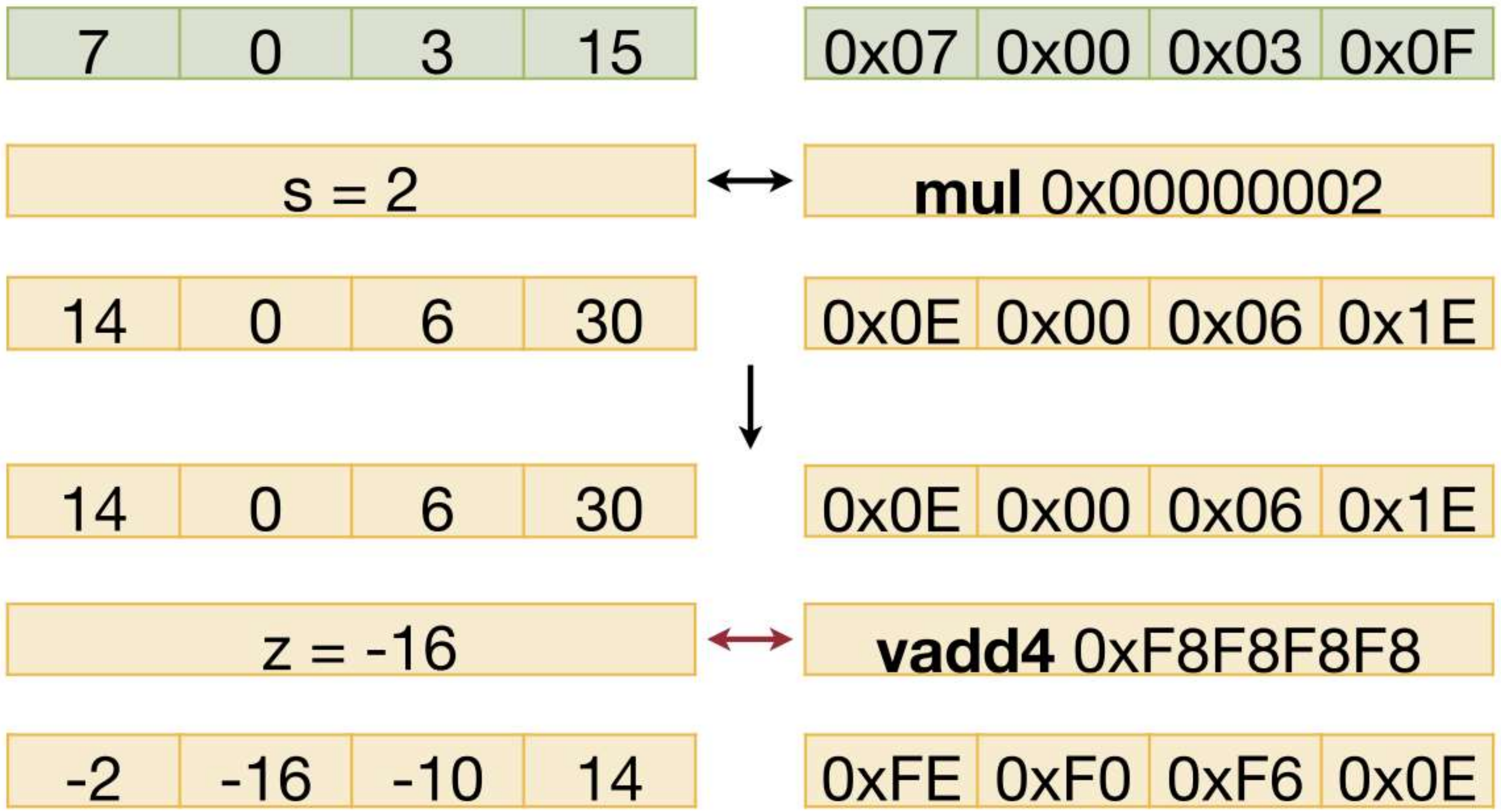
(d) Ours (INT4 Weights and INT8 Activations)

Qserve

INT4 to INT8

Qserve introduce register level parallelism to accelerate.

$$\mathbf{Qw}^{(0)}_{\mathrm{s8}} = (\mathbf{Qw}_{\mathrm{u4}} - \mathbf{z}_{\mathrm{u4}}) \cdot \mathbf{s}^{(1)}_{\mathrm{u8}},$$

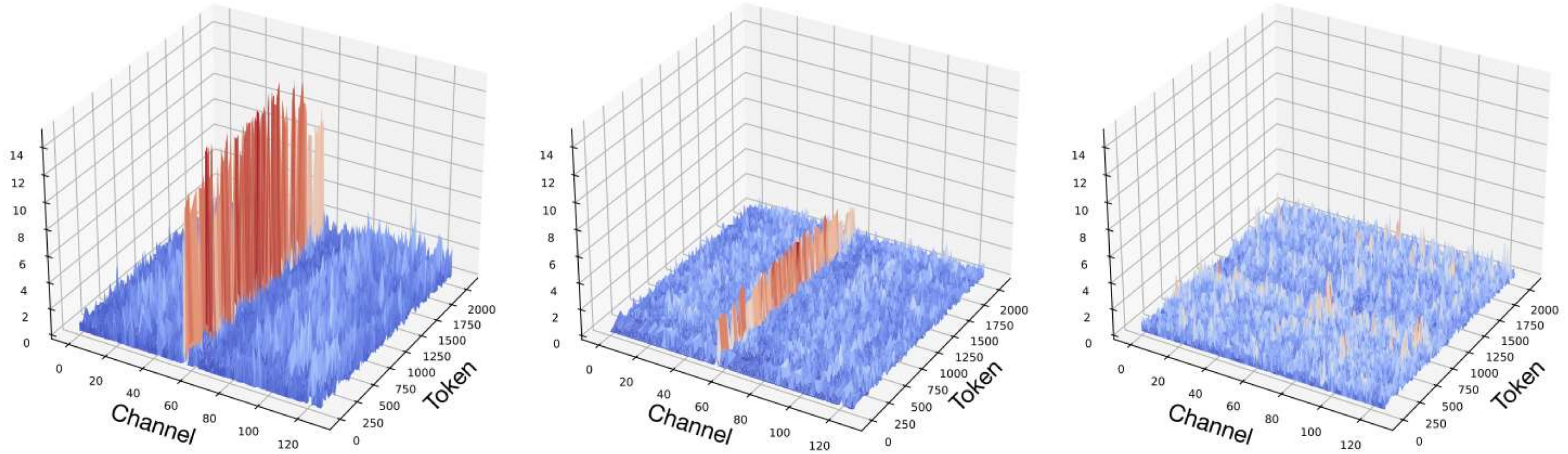


Subtraction **after** multiplication

- Sum can be achieved by vadd4
- Multiplication is achieved by padding 24 zeros to scale factor

Qserve

Smooth Attention



Layer 24 post-RoPE Keys (Original)   Layer 24 post-RoPE Keys (SmoothAttention)   Layer 24 Values

Key suffers from outliers while values are not. To quantize KV cache, we use:

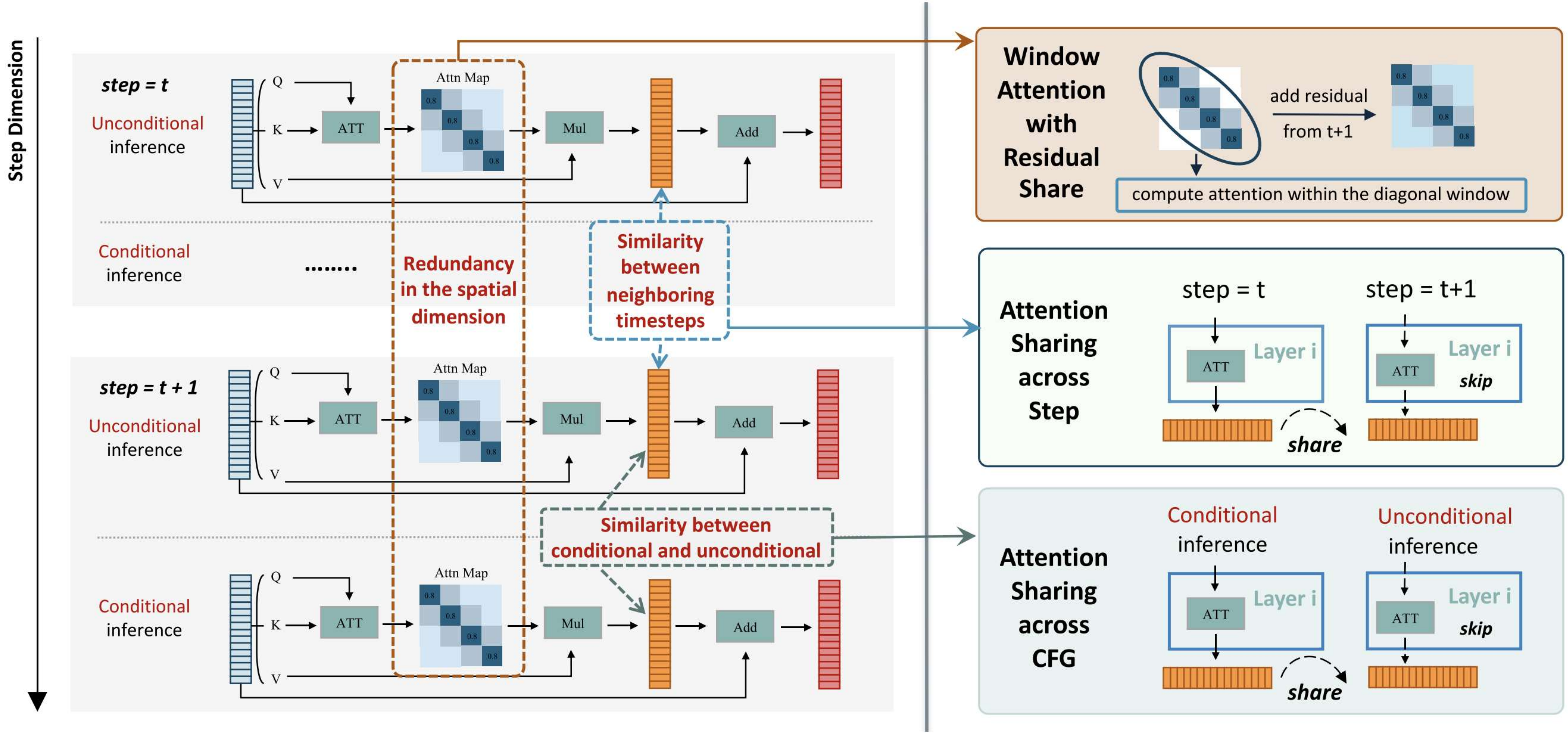
$$\mathbf{Z} = (\mathbf{Q}\boldsymbol{\Lambda}) \cdot \left(\mathbf{K}\boldsymbol{\Lambda}^{-1}\right)^{T}, \quad \boldsymbol{\Lambda} = \mathrm{diag}\,(\lambda)$$

$$\lambda_i = \max\left(|\mathbf{K}_i|\right)^{\alpha}.$$

Quantization in Sora

Quantization in Sora

Redundancy in Diffusion model

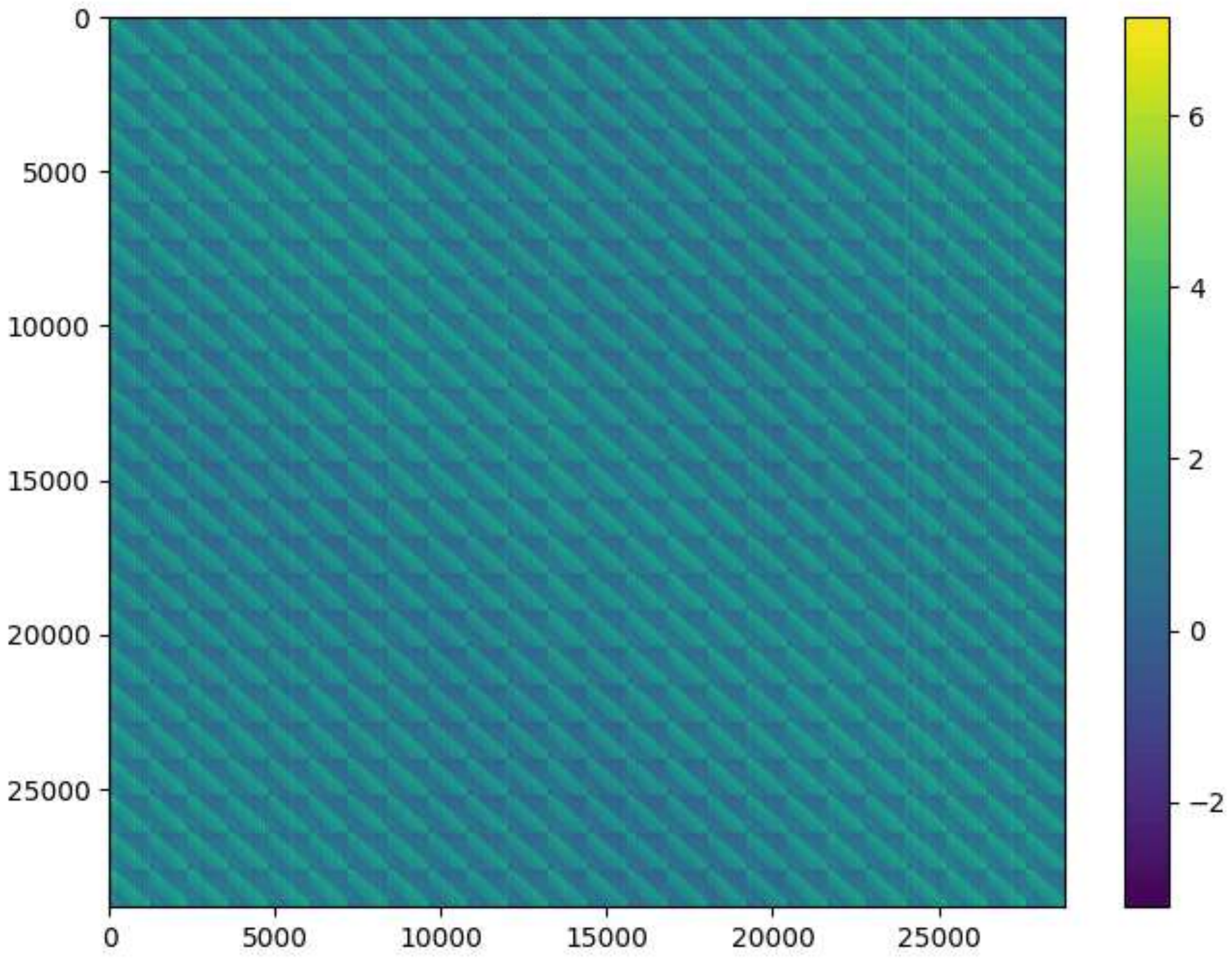


Quantization in Sora

Similarity in Attention heatmap

- We observe that the attention in different frames changes little in particular layers.

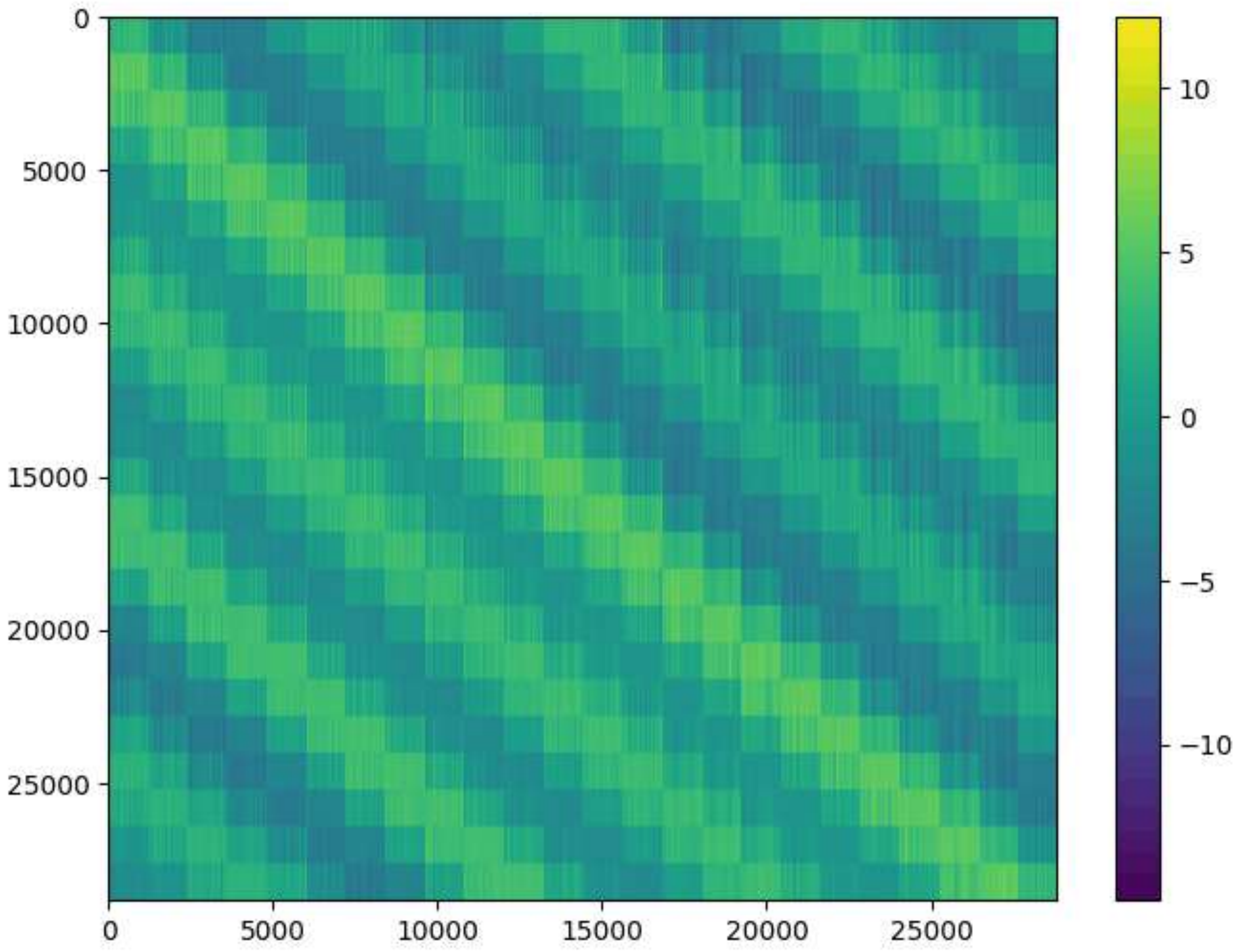
- We want to quantize the residual between different frames to reduce computation



Quantization in Sora

Similarity in Attention heatmap

- We also find that in particular layers, the heatmap exhibits a very distinct diagonal characteristic.
- Perhaps we can draw inspiration from the approaches used in other papers to handle outliers.



Thanks!

Wenxuan Miao 11.22