FALQON: Accelerating LoRA Fine-tuning with Low-Bit Floating-Point Arithmetic

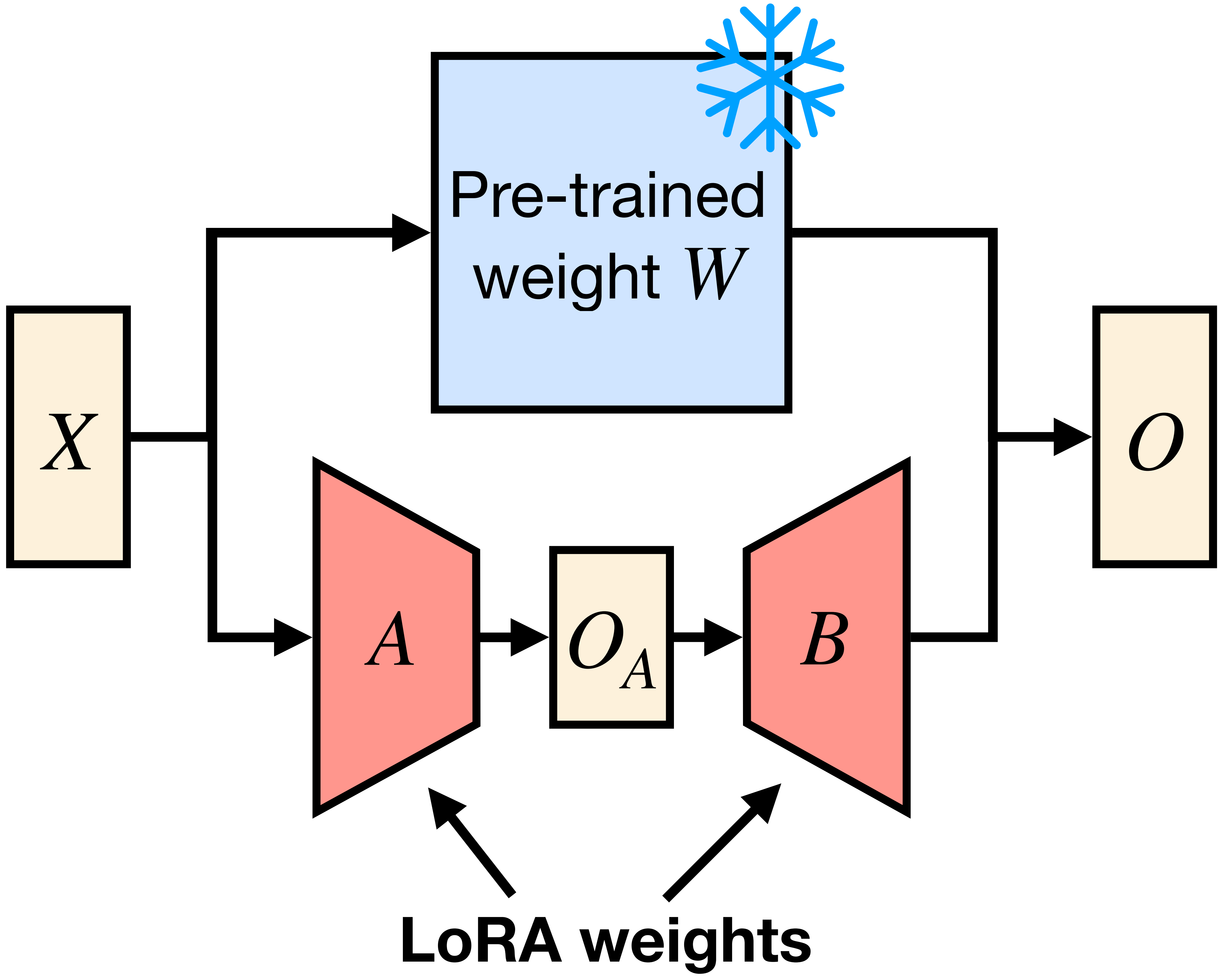
Kanghyun Choi, Hyeyoon Lee, SunJong Park, Dain Kwon, Jinho Lee

Department of Electrical and Computer Engineering
Seoul National University

NeurIPS 2025

Backgrounds

Low-Rank Adaptation (LoRA)

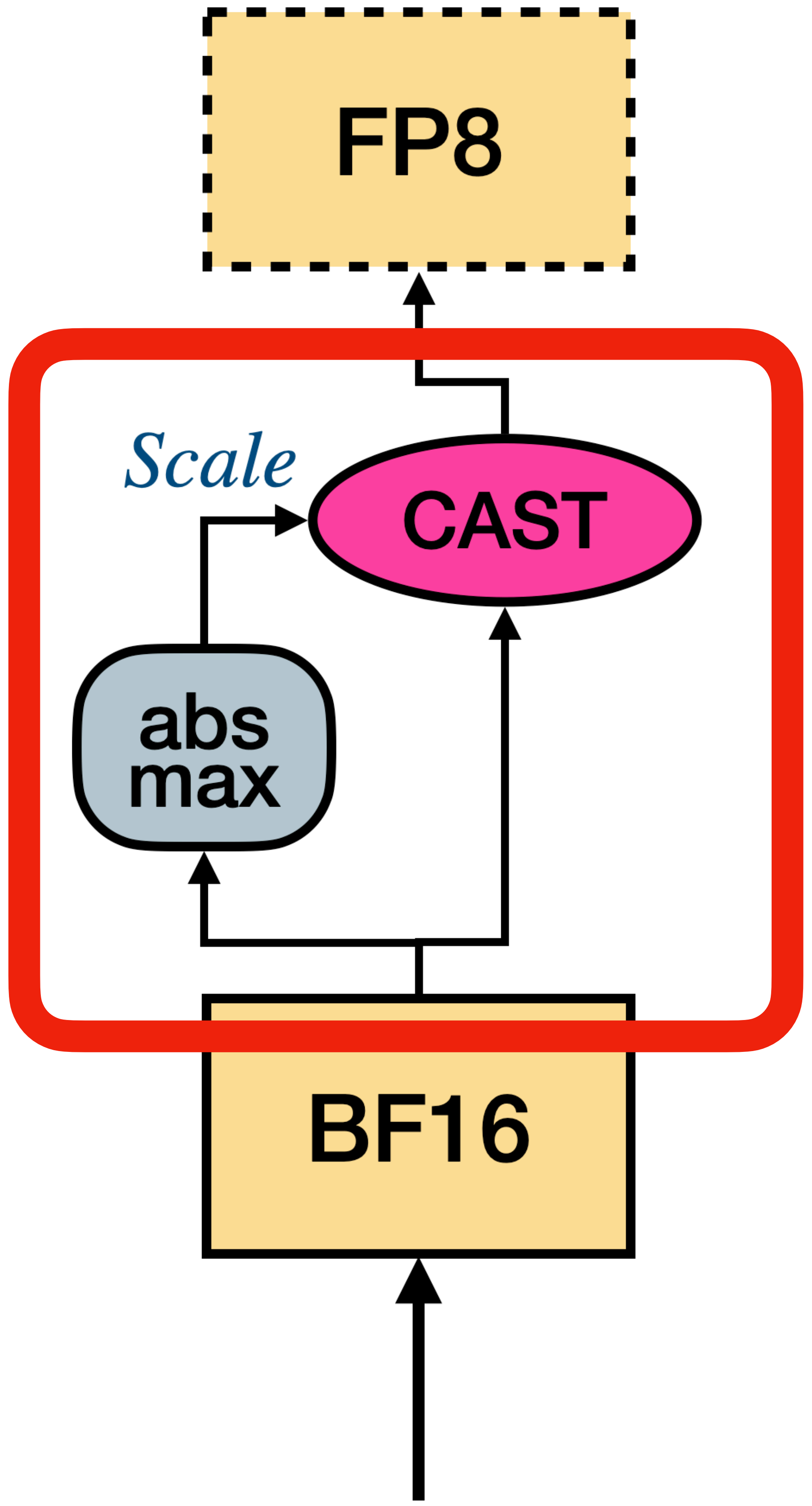


- Low-rank adaptation (LoRA)
  - Freeze pre-trained weights
  - Train LoRA weights only
  - Reduce memory consumption of gradient and optimizer state

$$W_{FT} = W_{orig} + \underbrace{\Delta W}_{\text{weight update}} \approx W_{orig} + \underbrace{BA}_{\text{low-rank projection (LoRA)}}$$

Backgrounds

FP8 Quantization in Linear Layer

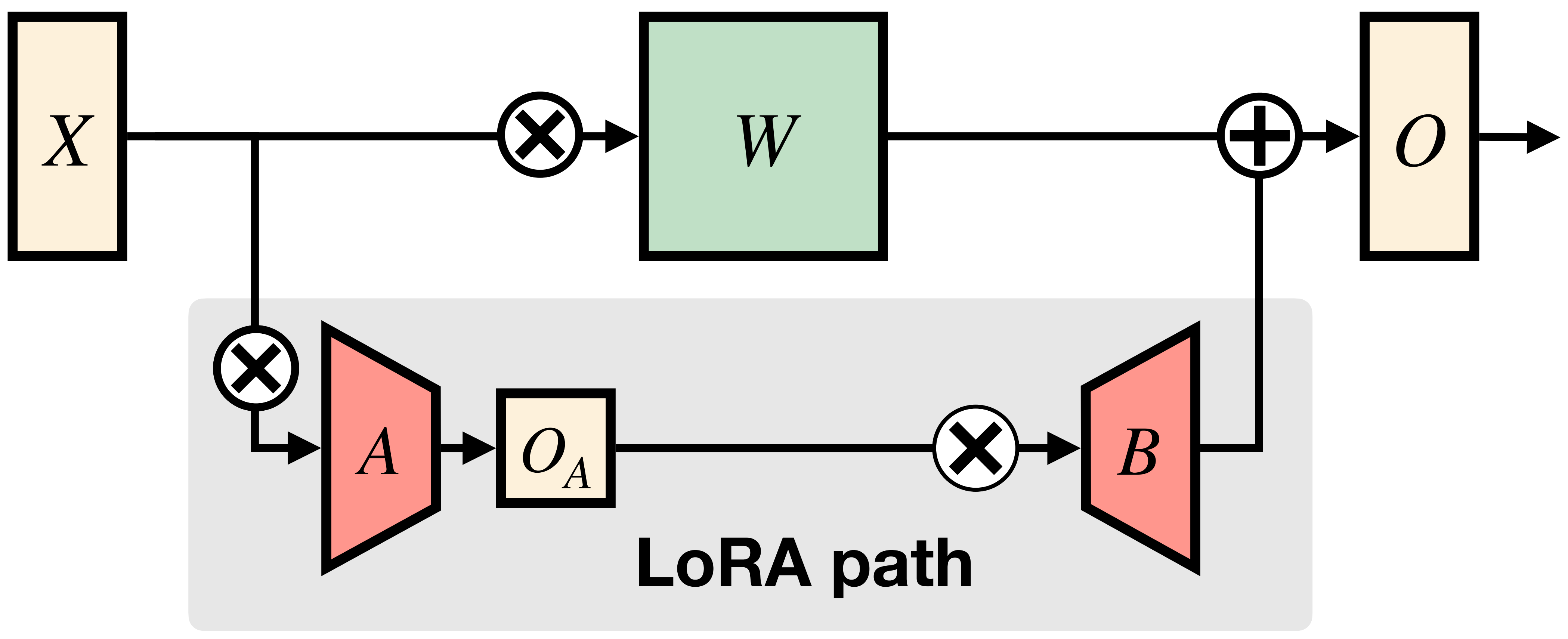


- FP8 quantization (conversion) requires scaling
  - Calculate absolute max (amax) for scaling
  - For quantization,
    we need a **reduction** for amax and **scaling**
  - For small-dimensional MatMul,
    the overhead exceeds the speed up

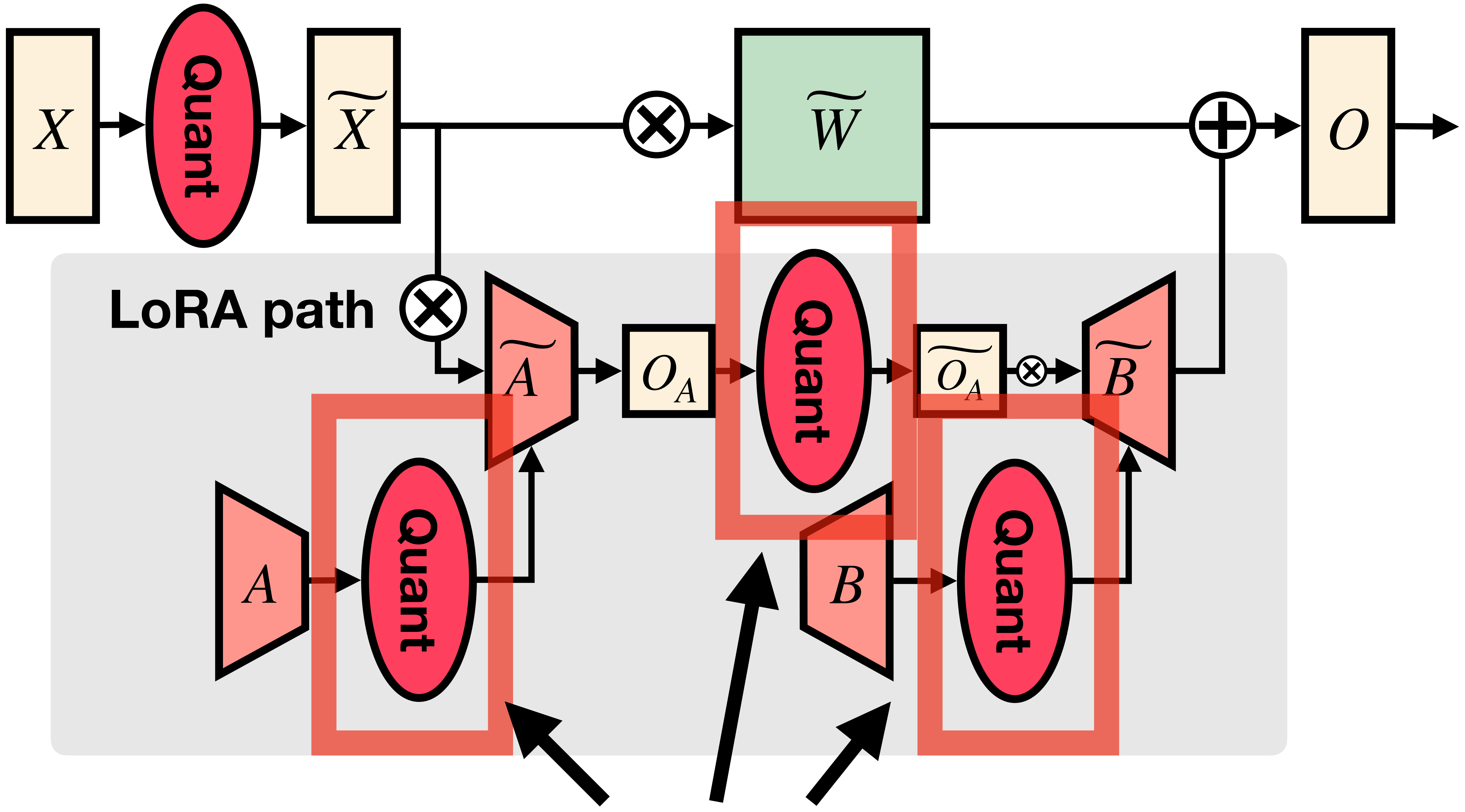
Motivational Study

Quantization Overhead of LoRA Layers

FP16 (No Quantization)



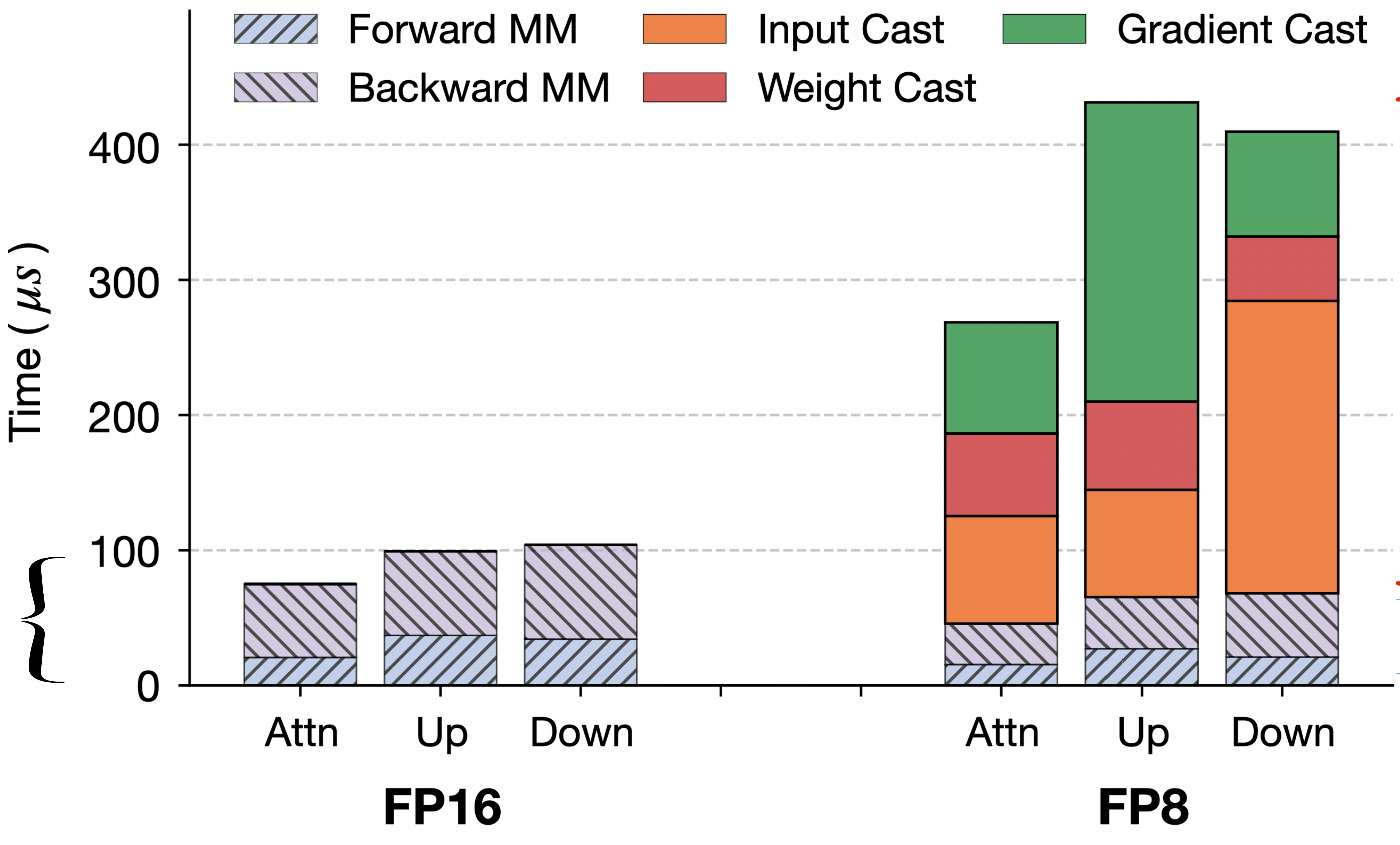
FP8 (Quantization)



Quantization overhead from LoRA path

Motivational Study

FP8 Quantization Overhead of LoRA Layers



FP16 MatMul

Quantization Overheads

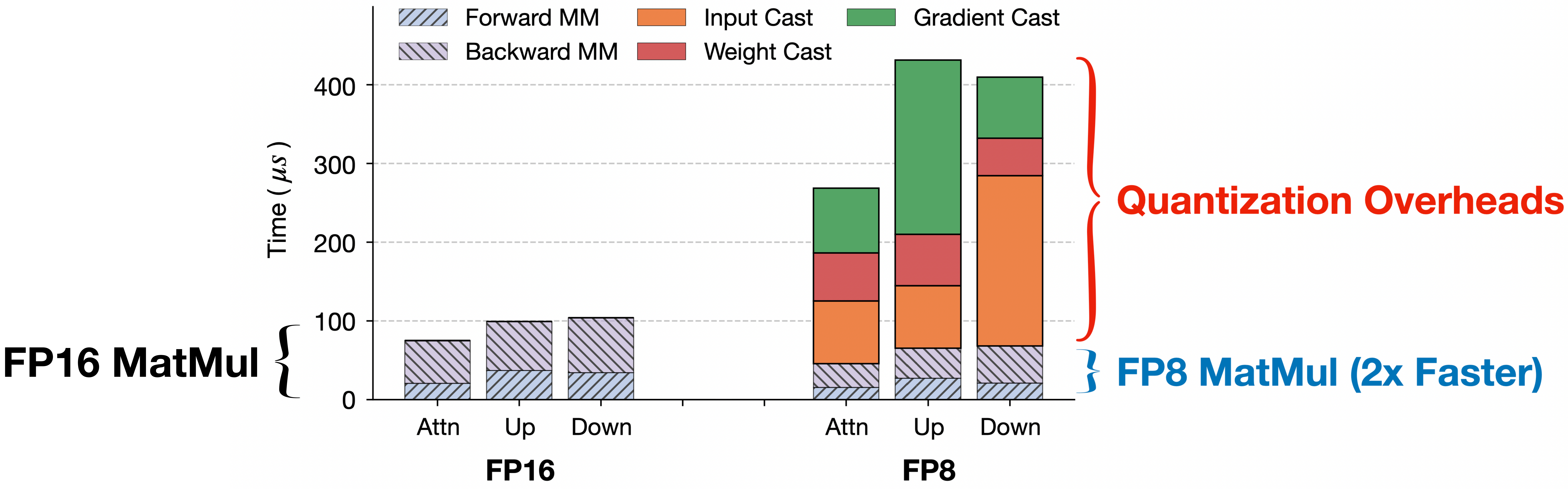
FP8 MatMul (2x Faster)

Motivational Study

FP8 Quantization Overhead of LoRA Layers

Problem: Current FP8 framework suffer from quantization overhead on LoRA

Research Goal: Design a low-overhead FP8 framework for LoRA



Proposed Method

1) Melded LoRA: Merging backbone and LoRA for Forward

Quantization Error

$\widetilde{W} = \text{Quantize}(W)$

$$\widetilde{W} = W_{orig} + \underbrace{\Delta W_Q}_{\boxed{\text{Quantization Error}}}$$

Proposed Method

1) Melded LoRA: Merging backbone and LoRA for Forward

Quantization Error

$\widetilde{W} = \text{Quantize}(W)$

$$\widetilde{W} = W_{orig} + \Delta W_Q$$

Quantization Error

$$W_{orig} + \widehat{B}\,\widehat{A}$$

where, $\widehat{B}\,\widehat{A} \approx \Delta W_Q$

Quantization Error as LoRA

Proposed Method

1) Melded LoRA: Merging backbone and LoRA for Forward

Quantization Error

$\widetilde{W} = \text{Quantize}(W)$

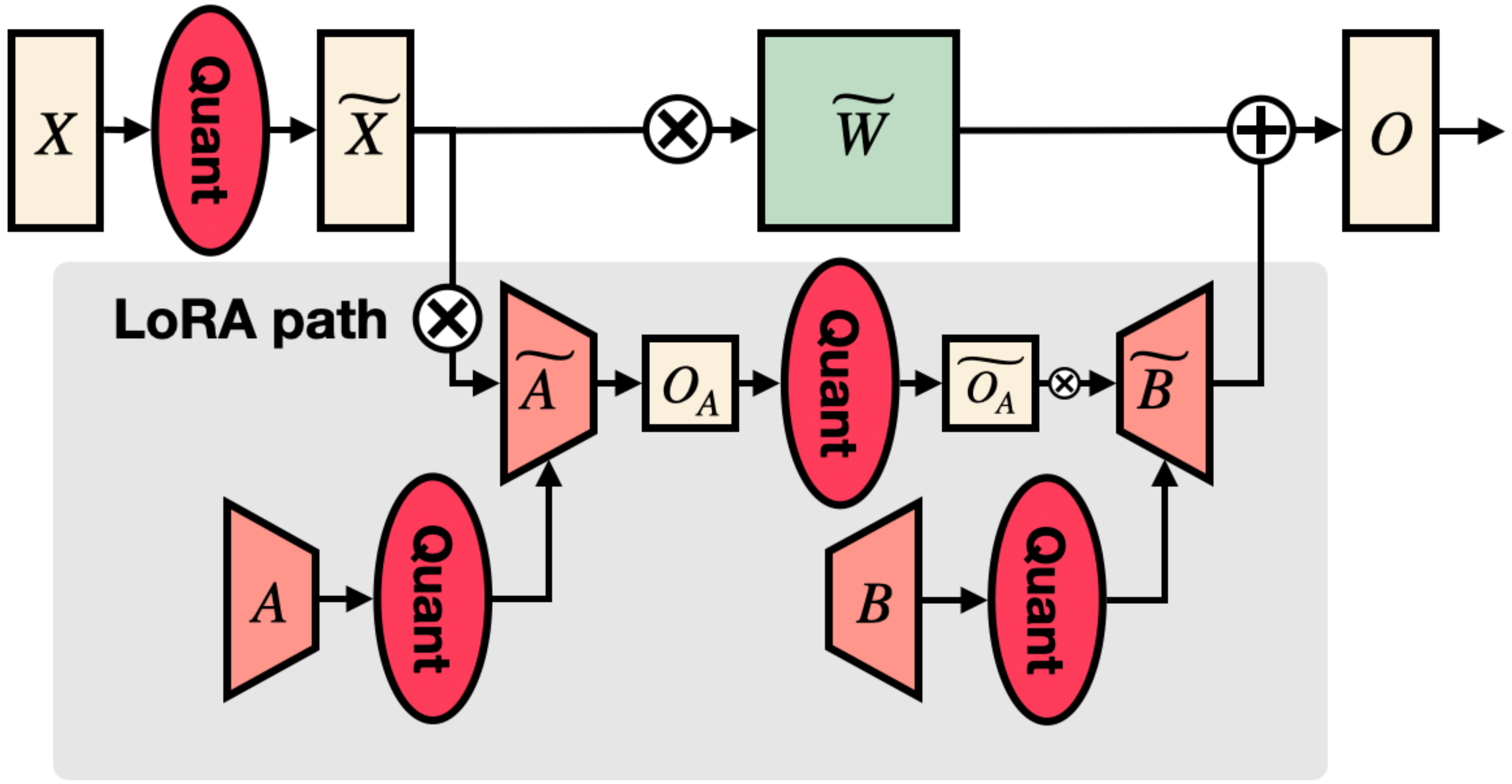
$$\widetilde{W} = W_{orig} + \underbrace{\Delta W_Q}_{\text{Quantization Error}}$$

$$W_{orig} + \widehat{B}\ \widehat{A}$$

where, $\widehat{B}\ \widehat{A} \approx \Delta W_Q$

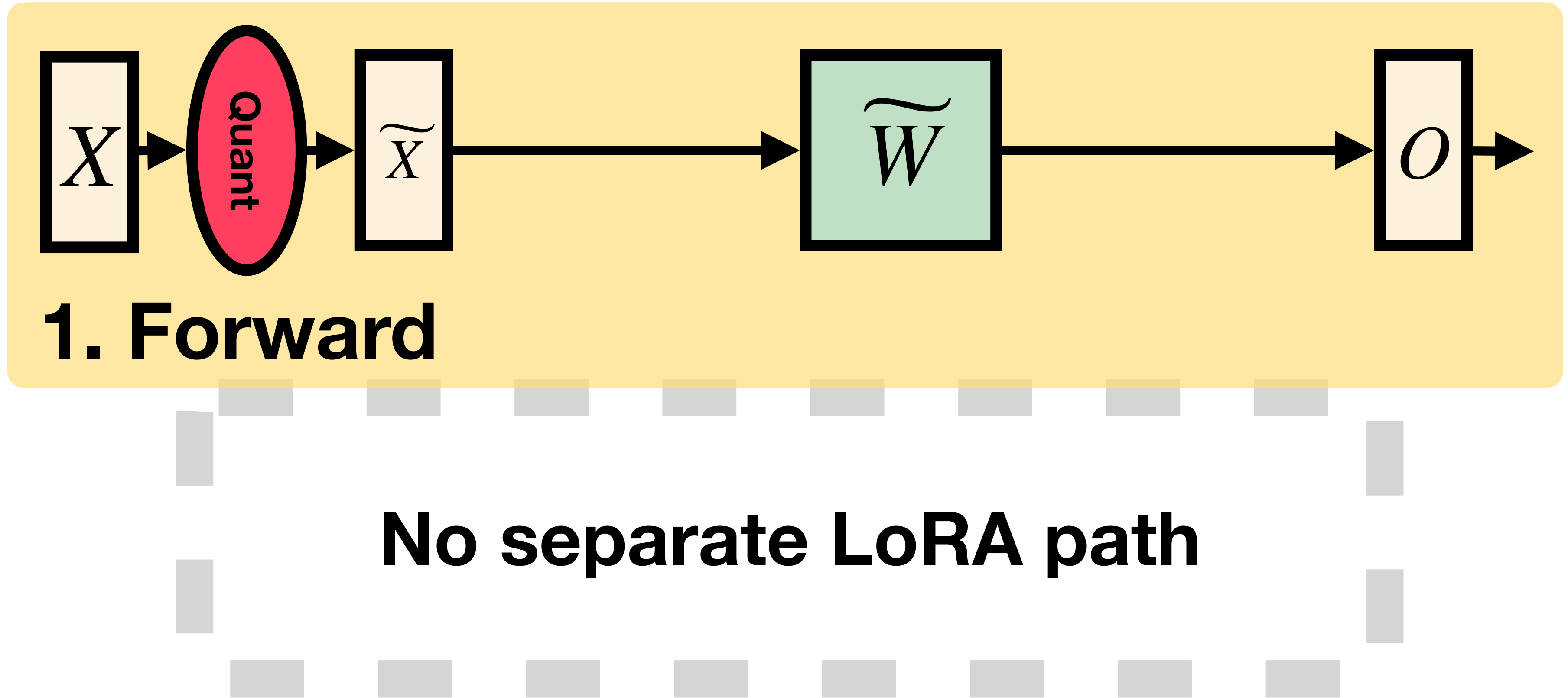
Quantization Error as LoRA

FP8 (Baseline)



FP8 (Ours)

Melded LoRA



Proposed Method

2) Efficient Gradient Computation for Melded LoRA

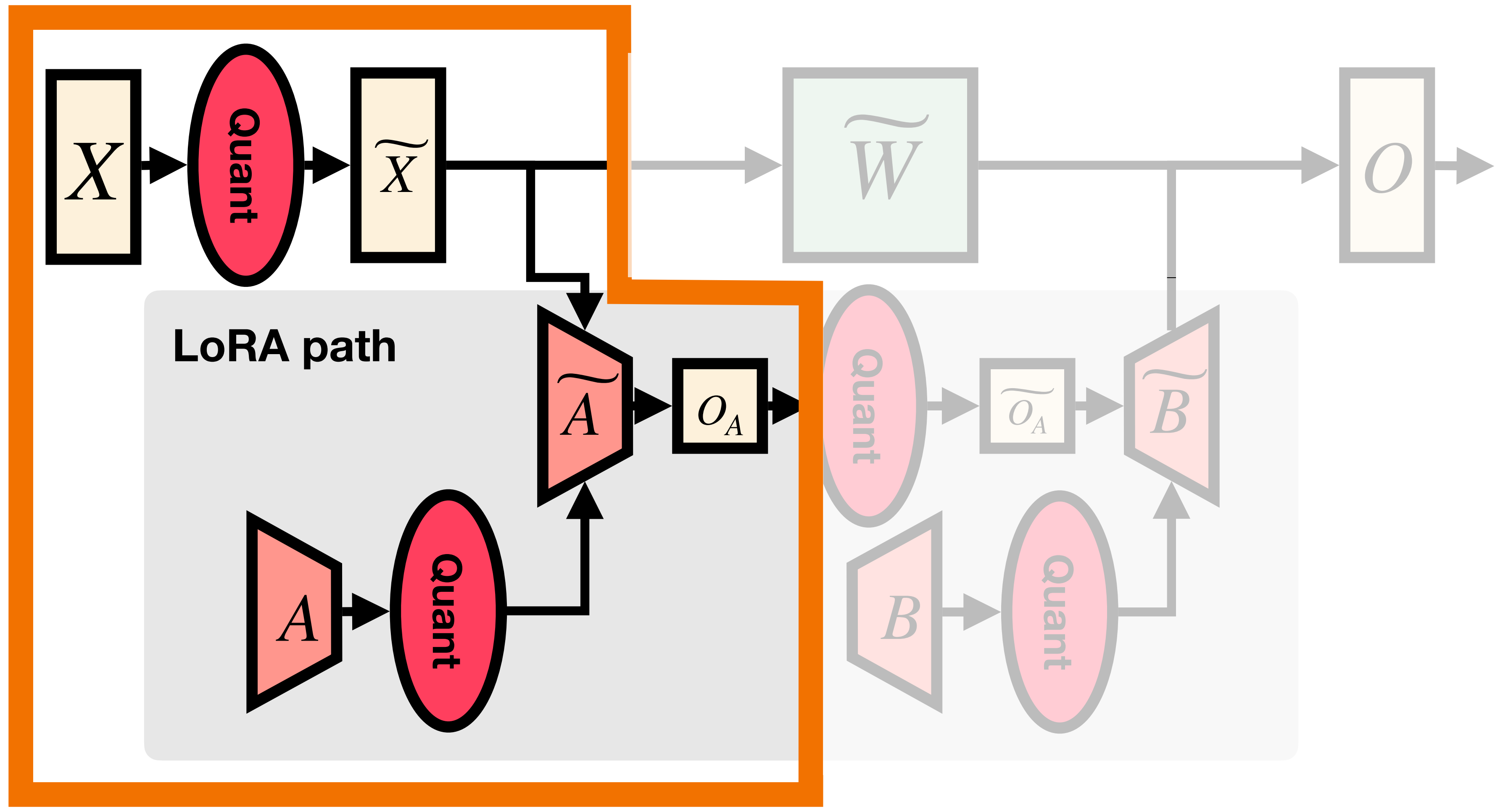
For backward:

(1) We freeze the A matrix

(2) Compute gradient of B matrix

$$\frac{\partial \mathcal{L}}{\partial B} = \frac{\partial \mathcal{L}}{\partial O} x^\top A^\top = \frac{\partial \mathcal{L}}{\partial O} \underline{(Ax)}^\top$$

Naive Ax computation yields further overhead



Proposed Method

2) Efficient Gradient Computation for Melded LoRA

For backward:

(1) We freeze the A matrix

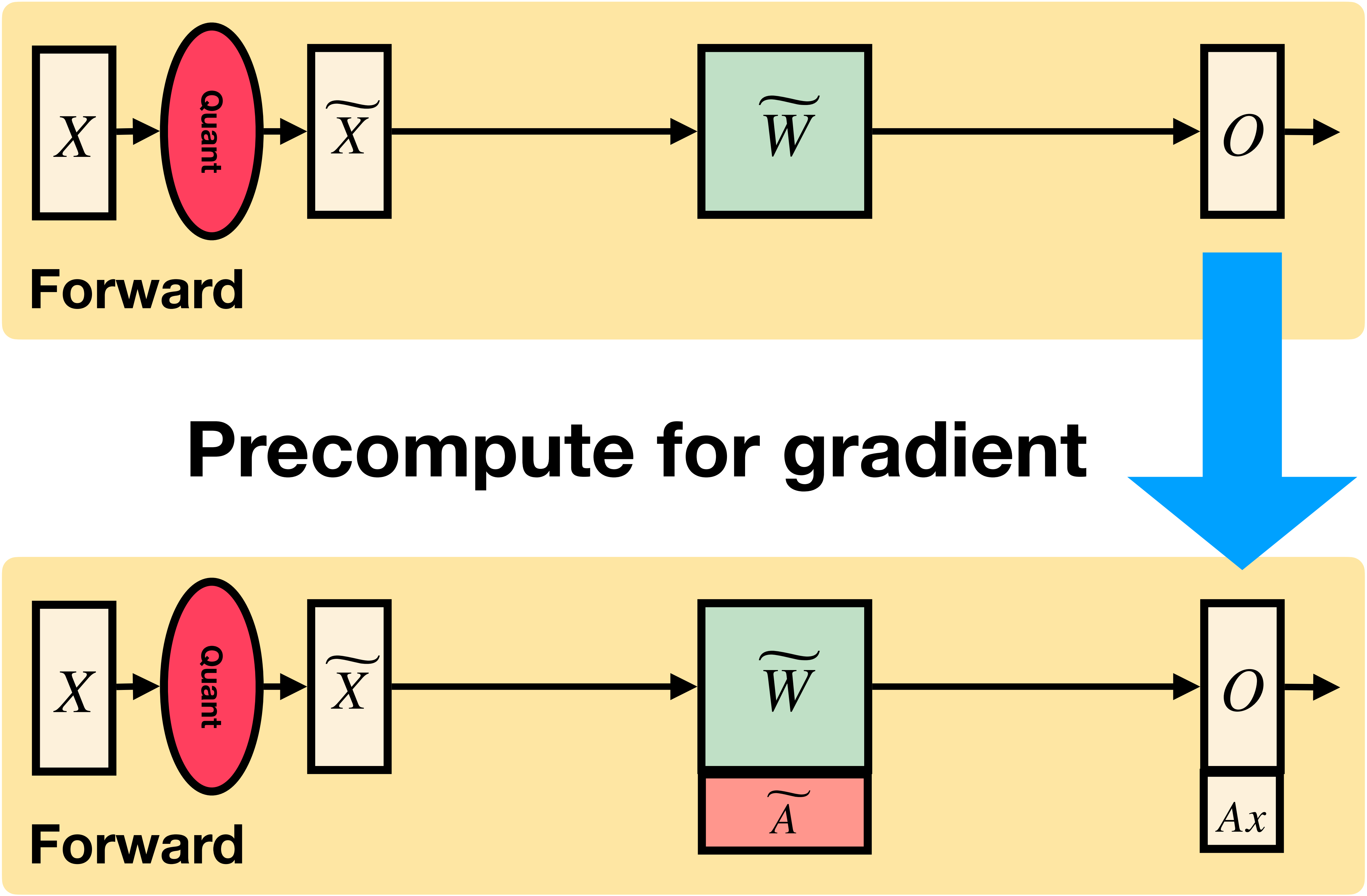
(2) Compute gradient of B matrix

$$\frac{\partial \mathcal{L}}{\partial B} = \frac{\partial \mathcal{L}}{\partial O} x^\top A^\top = \frac{\partial \mathcal{L}}{\partial O} (Ax)^\top$$

(2)-1 Merge A matrix to W:

$$\widetilde{W}' = \begin{bmatrix} \widetilde{W} \\ \widetilde{A} \end{bmatrix} \in \mathbb{R}^{(m+r)\times n}$$

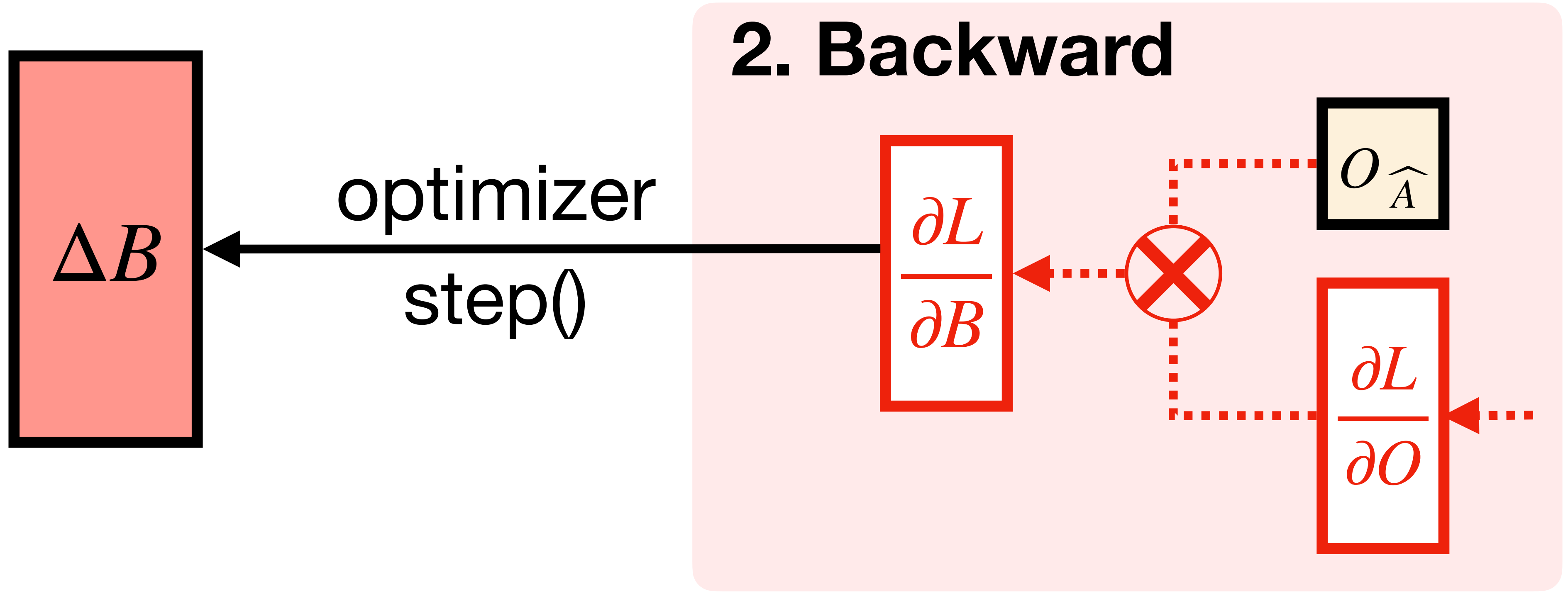
(2)-2 Precompute Ax in forward: $\widetilde{W}'\widetilde{x} = \begin{bmatrix} O \\ Ax \end{bmatrix} \in \mathbb{R}^{(m+r)\times d}$



Proposed Method

3) Row-wise Update of Quantized Weights

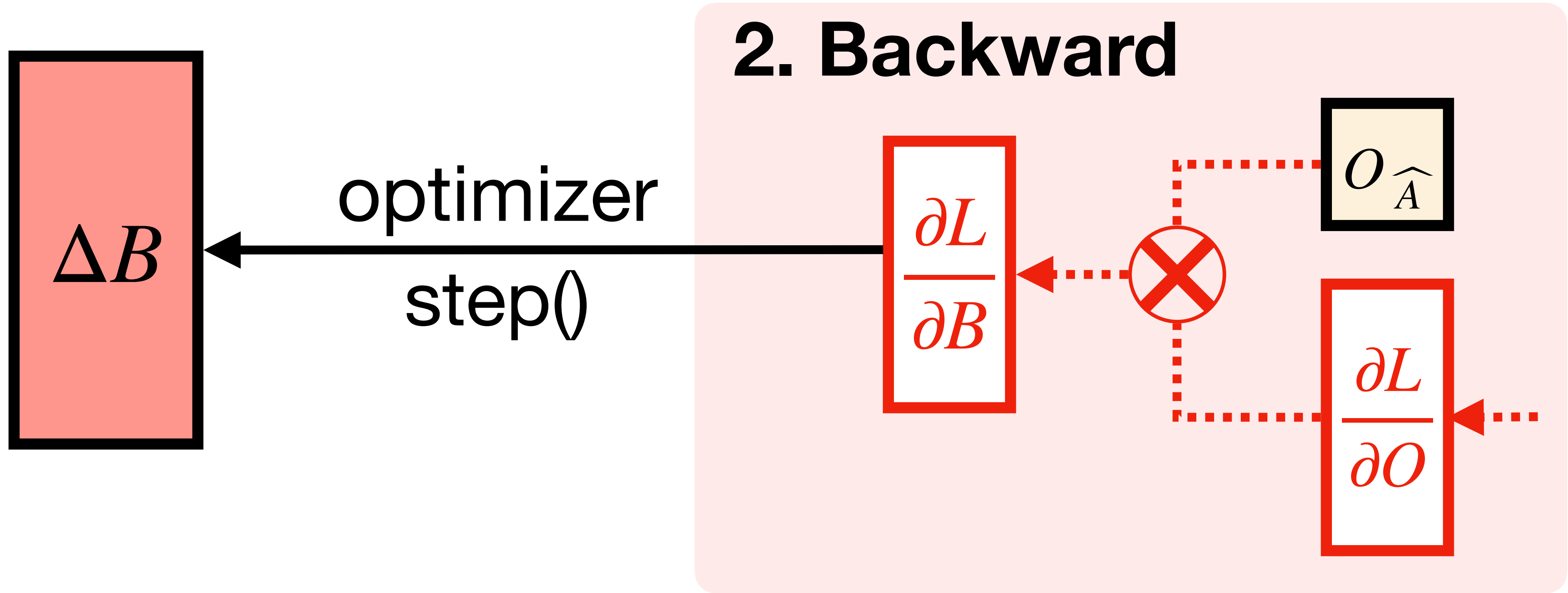
- ΔBuffer: store updates of B
  - Initialized to a zero-matrix



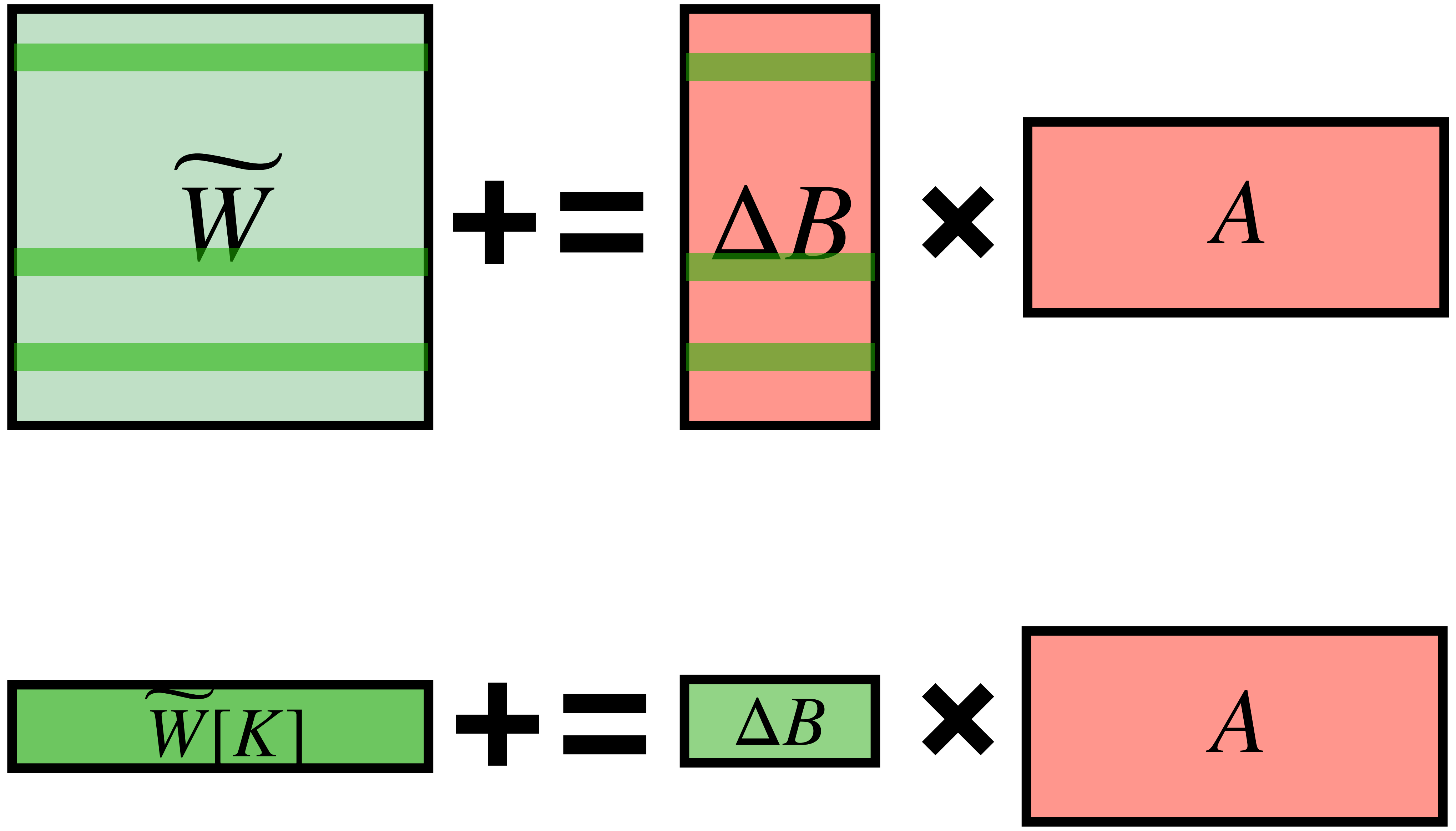
Proposed Method

3) Row-wise **Update** of Quantized Weights

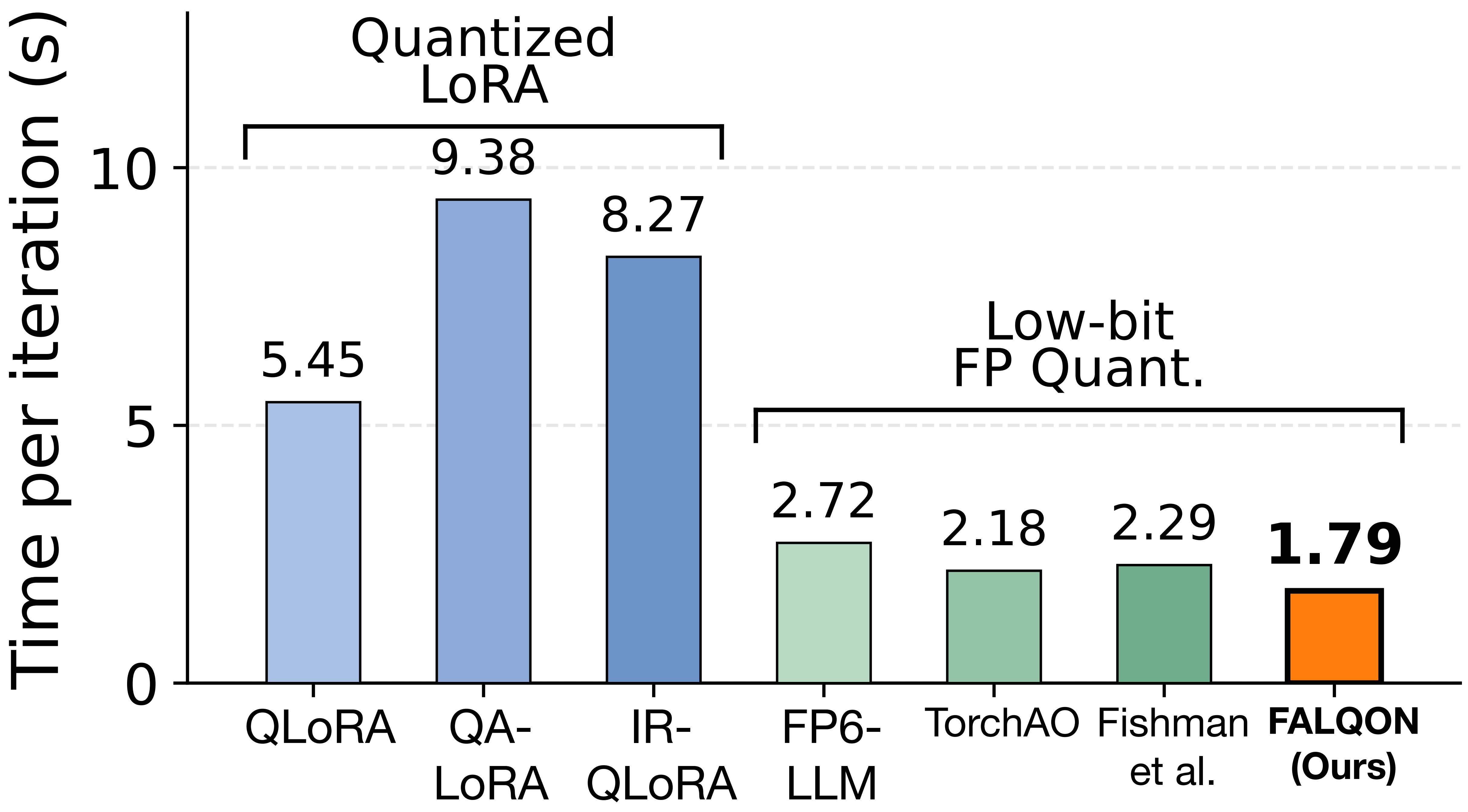
- ΔBuffer: store updates of B
  - Initialized to a zero-matrix



- Top-K Row-wise Update
  - Small updates cannot exceed quantization-grid
  - Apply large update rows only

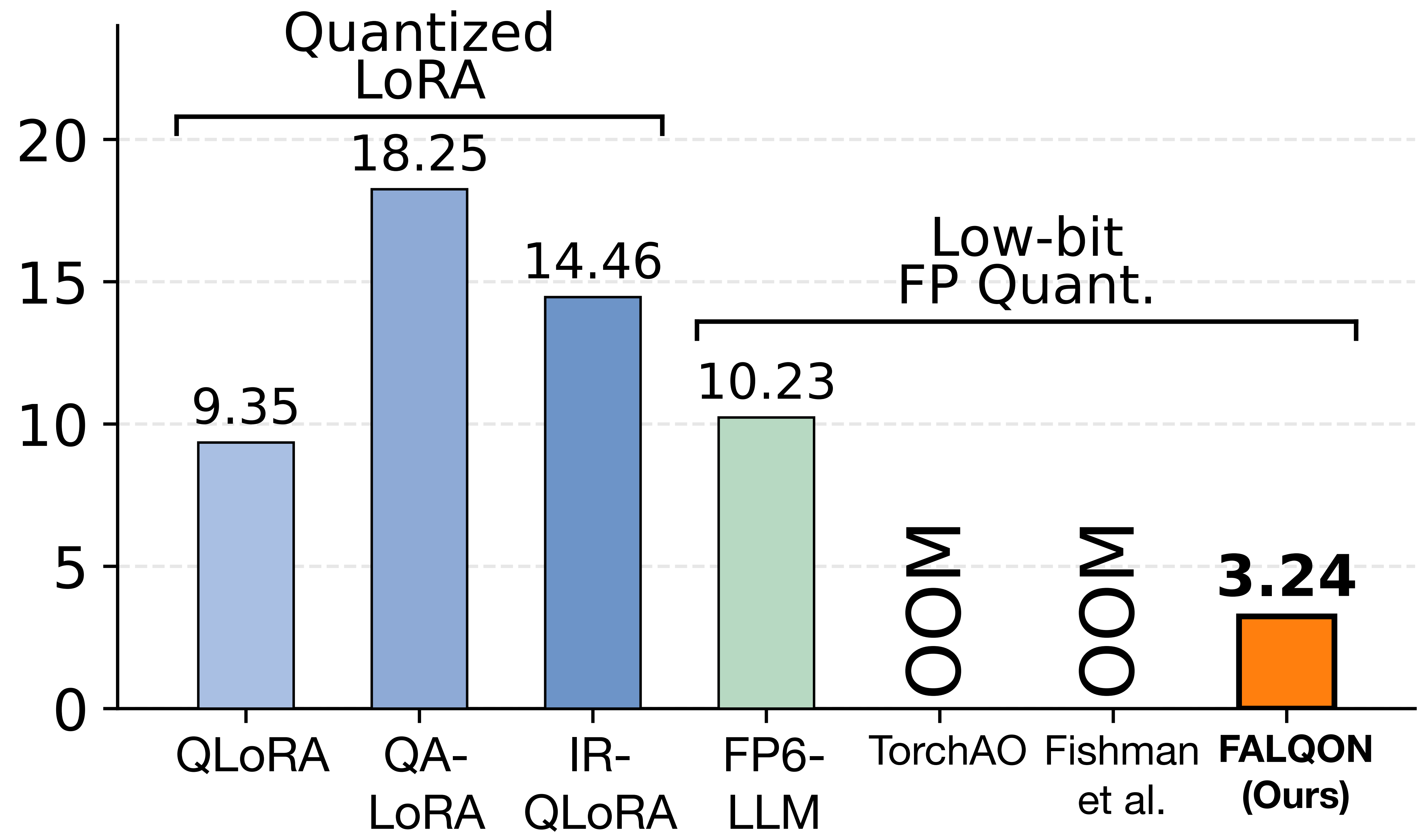


Evaluation



	QLoRA	QA-LoRA	IR-QLoRA	FP6-LLM	TorchAO	Fishman et al.	FALQON (Ours)
5-shot MMLU	0.3272	0.3548	0.3388	0.2295	0.3393	0.3537	0.3491

LLaMA-7B



	QLoRA	QA-LoRA	IR-QLoRA	FP6-LLM	TorchAO	Fishman et al.	FALQON (Ours)
5-shot MMLU	0.4443	0.4729	0.4349	0.2298	OOM	OOM	0.4644

LLaMA-13B

Conclusion

- We show that existing FP8 quantization methods incur substantial overhead with small-dimensional LoRA adapters.
- We propose FALQON, which merges the LoRA adapter in the quantized backbone and significantly reduces quantization overhead.
- FALQON achieves up to three times speedup over existing quantized LoRA methods.