Optimizing Large Language Model Training Using FP4 Quantization

ICML' 25

Ruizhe Wang, Yeyun Gong, Xiao Liu, Guoshuai Zhao
Ziyue Yang, Baining Guo, Zhengjun Zha, Peng Cheng

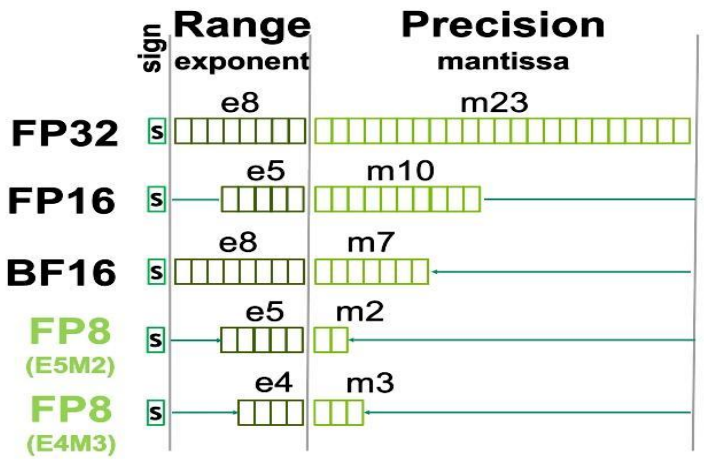
Presenter: YiFan Hu
2025.9.17

Contents

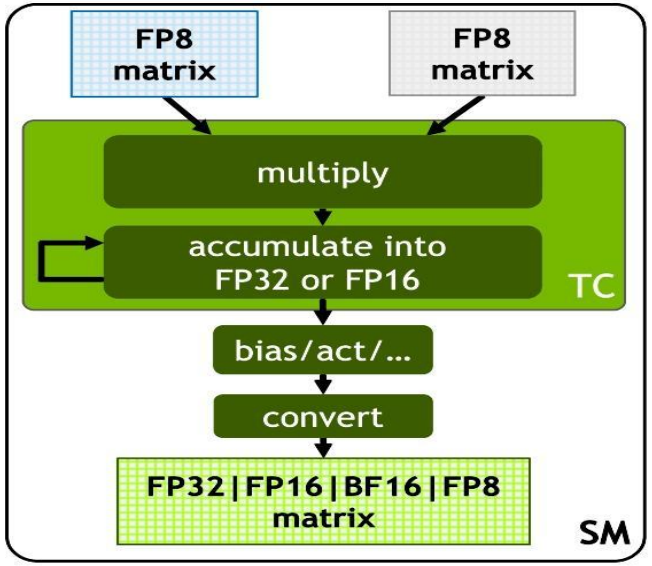
- **Background & Motivation**
- **Methodology**
- **Evaluation**
- **Thinking**

Background & Motivation

- 大模型训练成本高：算力/显存/带宽瓶颈显著；需要更低比特以提升吞吐并降低内存与通信开销
- 硬件趋势：H100 已支持 FP8 张量核（相对 FP16 ≈2×吞吐）；Blackwell（B200）支持 FP6/FP4，理论吞吐更高



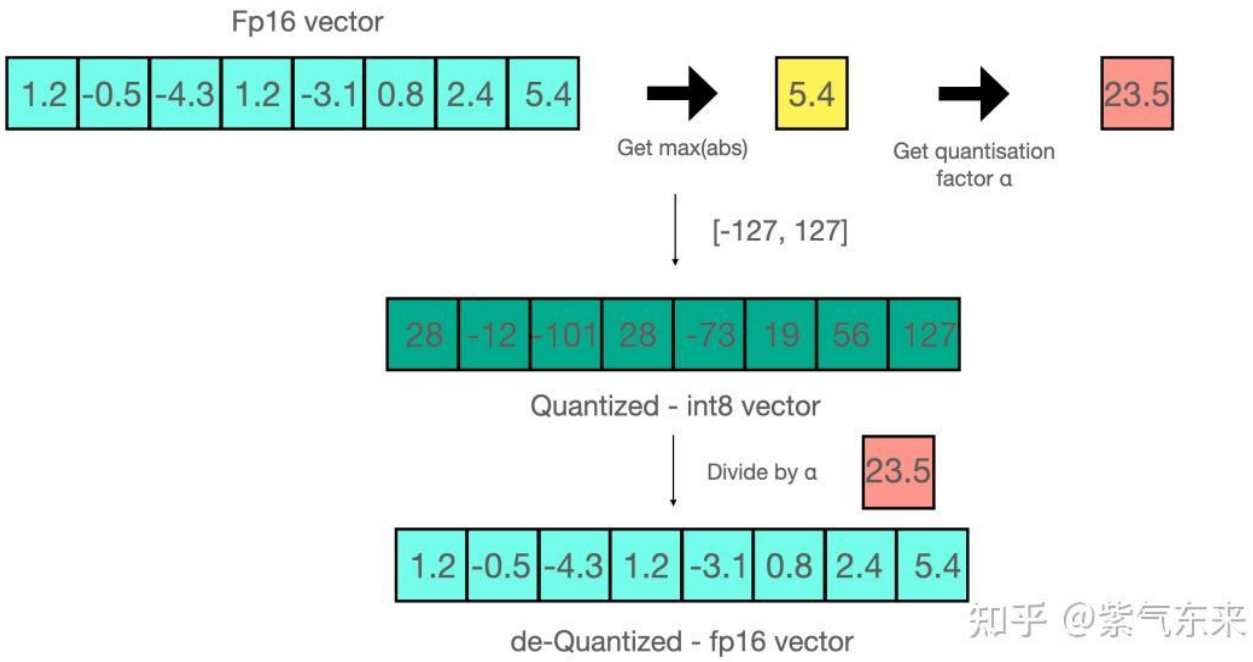
Allocate 1 bit to either range or precision



Support for multiple accumulator and output types

Background & Motivation

量化训练与推理：



显卡在以FP8格式计算矩阵乘法时能达到FP16计算的两倍吞吐量，FP4达到4倍

Background & Motivation

三种不同的FP4格式：E1M2，E2M1，E3M0

Table 4. FP4 Quantization Table under different FP4 formats.

Format	Binary Sequence														
	1111	1110	1101	1100	1011	1010	1101	1000/0000	0001	0010	0011	0100	0101	0110	0111
E1M2	-3.5	-3	-2.5	-2	-1.5	-1	-0.5	±0	0.5	1	1.5	2	2.5	3	3.5
E2M1	-6	-4	-3	-2	-1.5	-1	-0.5	±0	0.5	1	1.5	2	3	4	6
E3M0	-16	-8	-4	-2	-1	-0.5	-0.25	±0	0.25	0.5	1	2	4	8	16

FP16量化FP4公式：absmax

$$x_{\text{fp4}} = Q(x_{\text{fp16}} \cdot \gamma), \quad \gamma = \frac{\text{MAX}_{\text{fp4}}}{\max(|x_{\text{fp16}}|)}$$

FP4计算公式

$$\text{Value} = (-1)^S \times (1.M) \times 2^{E-\text{bias}} \qquad 1.m = 1 + \frac{M}{2^p} \qquad \textbf{bias} = 2^{E-1} - 1$$

Background & Motivation

问题：

- 精度缺乏：使用更少的比特来表示一个数，其所能表示的动态范围一定会受限，可能导致上溢出或者下溢出
- 反向传播过程中，量化函数导数为0或无，影响反向传播，STE方法效果较差

$$x_{\text{fp4}} = Q(x_{\text{fp16}} \cdot \gamma), \quad \gamma = \frac{\text{MAX}_{\text{fp4}}}{\max(|x_{\text{fp16}}|)}$$

下溢出：当max过大，γ 极小，导致更多的样本变为0

上溢出：缩放后的数值超过了fp4的取值范围，则被转换为最大值or最小值

Contents

- **Background & Motivation**
- **Methodology**
- **Evaluation**
- **Thinking**

Methodology

原始方法STE(Straight-Through Estimator)

- 反向传播基础： Y: 输出 A: 激活 W:权重 W_q:量化后的权重 f: 量化函数

$$Y = AW_q = Af(W) \qquad \frac{\partial L}{\partial W} = \frac{\partial L}{\partial W_q}\frac{\partial W_q}{\partial W} = (A^T\frac{\partial L}{\partial Y})\frac{\partial W_q}{\partial W}$$

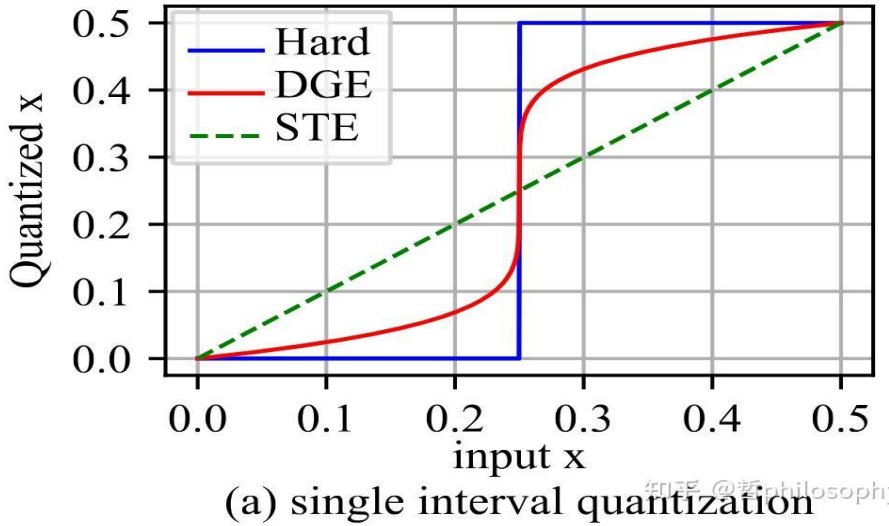
$\frac{\partial L}{\partial W}$ 是损失函数对原权重的导数，即权重梯度

因为 $\frac{\partial W_q}{\partial W}$ 表示f函数的导数

$$\frac{\partial W_q[i,j]}{\partial W[k,l]} = \begin{cases} f'(W[i,j]), & \text{if } (i,j) = (k,l), \\ 0, & \text{otherwise.} \end{cases}$$

$$\frac{\partial L}{\partial W} = \frac{\partial L}{\partial W_q} \odot f'(W)$$

符号“⊙”： 逐元素乘



(a) single interval quantization

Methodology

可微梯度估计（DGE, Differentiable Gradient Estimator）

- STE方法导致计算结果不精确

  1. 在阶梯交替处，导数应较大

  2. 在阶梯平稳处，导数应接近0

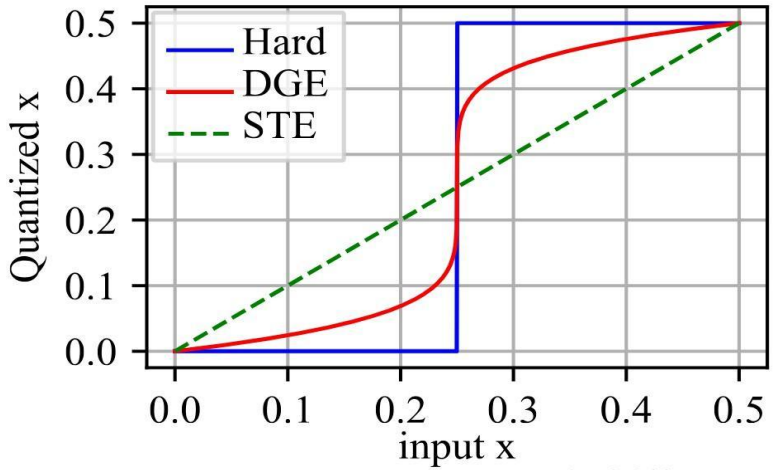
- 设计一种新的f函数

$$f(x) = \frac{\delta}{2} \cdot \left(1 + \text{sign}\left(\frac{2x}{\delta} - 1\right) \cdot \left|\frac{2x}{\delta} - 1\right|^{\frac{1}{k}}\right)$$

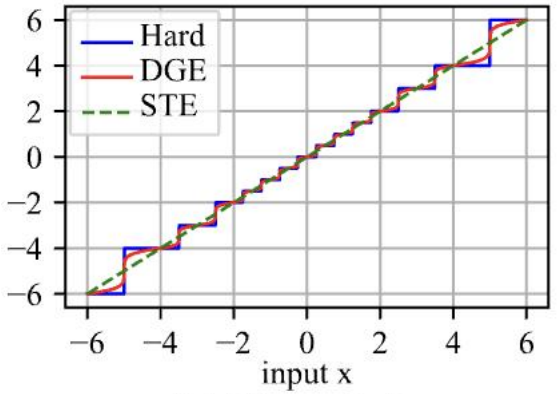
$$f'(x) = \frac{1}{k} \cdot \left|\frac{2x}{\delta} - 1\right|^{\frac{1}{k}-1}$$

k越大，接近硬量化

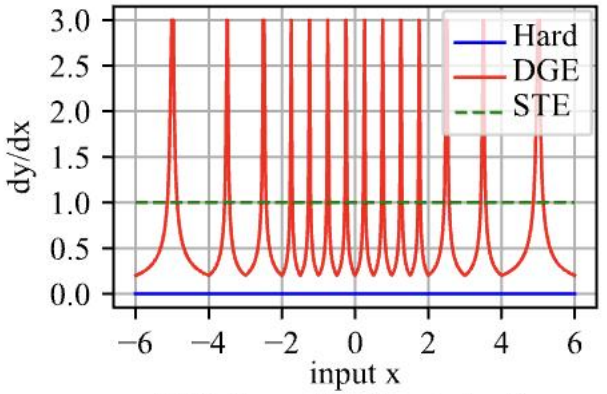
k越小，更平滑



(a) single interval quantization



(b) full quantization



(c) full quantization derivative

Methodology

如何得到量化函数

平滑处理：设置$f(x) = x^a$，x接近1，f(x)趋近于1，x接近0，f(x)趋近0，接近一个阶梯的样式，但更加的平滑

奇偶对称：从-1过渡到1，加上符号函数

$$f(x) = \text{sign}(x) \cdot |x|^a$$

目标布宽：量化里每个步设置为δ，把[-1, 1]的范围缩放到[−δ/2，δ/2]

$$f(x) = \text{sign}\left(\frac{2x}{\delta}\right) \cdot \left|\frac{2x}{\delta}\right|^a$$

$$f(x) = \text{sign}\left(\frac{2x}{\delta} - 1\right) \cdot \left|\frac{2x}{\delta} - 1\right|^a$$

转移中央位置δ/2并缩放

$$f(x) = \frac{\delta}{2}\left(1 + \text{sign}\left(\frac{2x}{\delta} - 1\right) \cdot \left|\frac{2x}{\delta} - 1\right|^a\right)$$

Methodology

异常值钳位和补偿（OCC, Outlier Clamping and Compensation）

问题：异常值导致下溢问题

解决方案：使用OCC钳位+补偿

钳位：$Y_c = \text{clamp}(Y, \max = \alpha, \min = 1 - \alpha)$

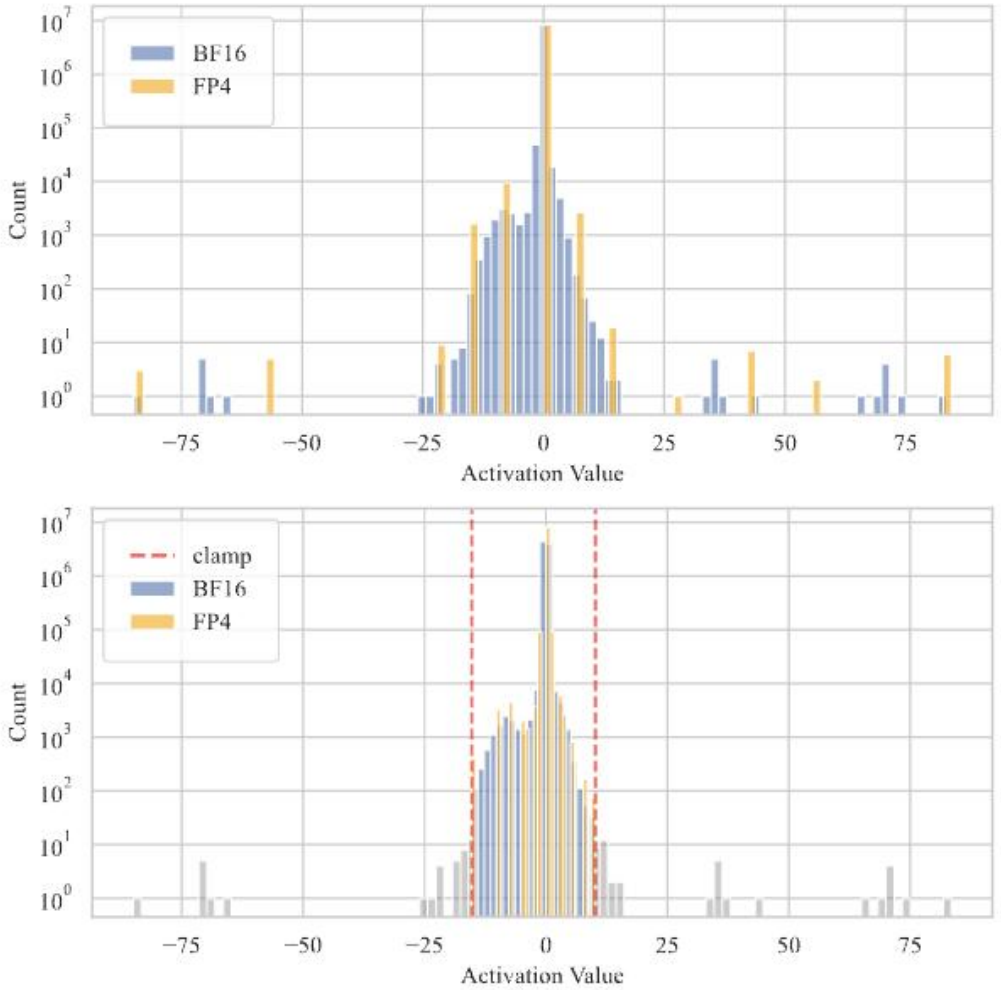
但导致了失去部分数据的误差

补偿：$\Delta Y = Y - Y_c$ 仍然需要参与运算，可使用稀疏矩阵运算

$$A = [-7.0, -2.0, -0.3, 0.1, 0.5, 1.2, 3.0, 8.0]$$

$$[-7, -2, -0.3, 0.1, 0.5, 1.2, 3.0, 7.0]$$

$$\Delta A = [0, 0, 0, 0, 0, 0, 0, 1.0]$$



Methodology

异常值钳位和补偿（OCC, Outlier Clamping and Compensation）

钳位：

1. 按照绝对值排序
2. 查找对应位置数值
3. 钳制最大值
4. 计算残差矩阵

$$Y_c = \text{clamp}(Y, \max = \alpha, \min = 1 - \alpha)$$

$$A = [-7.0, -2.0, -0.3, 0.1, 0.5, 1.2, 3.0, 8.0]$$

$$|A| = [0.1,\ 0.3,\ 0.5,\ 1.2,\ 2.0,\ 3.0,\ 7.0,\ 8.0]$$

$$\alpha = 0.75$$

$$\alpha \times N = 0.75 \times 8 = 6$$

$$\tau = 3.0$$

$$A_c = [-3.0,\ -2.0,\ -0.3,\ 0.1,\ 0.5,\ 1.2,\ 3.0,\ 3.0]$$

$$\Delta A = A - A_c = [-4.0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 5.0]$$

补偿：

1. 使用高精度稀疏矩阵运算
2. 与原始数据相加，得到最终结果

$$Y = A_c W + \Delta A W$$

Contents

- **Background & Motivation**
- **Methodology**
- **Evaluation**
- **Thinking**

Evaluation

1. 实验设置

项目	设置
量化对象	GeMM
混合精度	非 GeMM 占比较小，使用 FP16/BF16 保持稳定与精度
模型	LLaMA 2（作为主要架构）
数据	DCLM 数据集，从零训练
超参数	峰值 LR = 3e-4；权重衰减 0.1；Adam：β1=0.9，β2=0.95，ε=1e-8
FP4超参数	DGE：k = 5；OCC：分位数 α = 0.99
序列与批量	序列长度 2048；全局 batch size 2048（≈ 4.19M tokens/step）
训练 token	100B tokens

◆ **注意事项**

① A按照token进行量化，W按照通道量化

② 使用FP8模拟FP4检验，底层运算还是FP8 。输入/输出张量在软件上先被量化到 FP4 格式（E2M1），存储成 FP4 的值。实际计算时：把 FP4 的数值“升格”成 FP8，在 FP8 核里做乘加，结果再按 FP4 的规则截断/舍入

③ Adam：梯度与一阶动量 m 存 FP8；二阶动量 v 存 FP16；其余操作 FP16/BF16

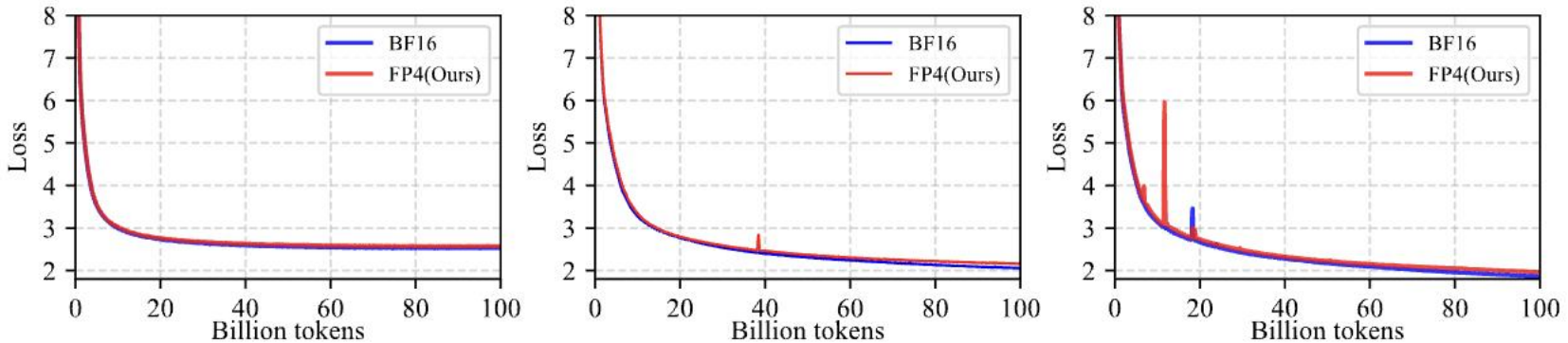
Evaluation

2. 主要结果

Model Size	Precision	Average	PiQA	Hellaswag	ObQA	Arc-C	Arc-E	BoolQ	LogiQA	SciQ	Lambada
1.3B	BF16	**53.23**	71.11	50.80	36.60	36.69	68.60	57.83	30.26	83.30	43.84
	FP4(Ours)	**53.13**	70.89	50.82	36.20	36.86	67.47	58.23	29.49	83.90	44.30
7B	BF16	**53.87**	71.22	52.03	37.40	38.99	67.47	60.55	27.65	85.00	44.56
	FP4(Ours)	**54.42**	71.87	52.97	38.40	39.85	67.97	62.20	27.96	84.70	43.88
13B	BF16	**54.44**	72.80	53.56	38.60	38.82	67.97	57.40	29.65	86.30	44.87
	FP4(Ours)	**54.95**	73.78	54.12	39.60	39.68	67.89	55.90	30.88	85.80	46.89

结论：

1. 模型越大，损失越小
2. 在小模型中，FP4甚至部分超越BF16
3. FP4与BF16差距很小



(a) LLaMA-1.3B (b) LLaMA-7B (c) LLaMA-13B

Size	Precision	Average	Lbd.OAI	Lbd.std	Pile10k	Wikitext
1.3B	BF16	**37.38**	14.98	25.10	82.77	26.65
	FP4(Ours)	**36.86**	15.33	23.07	82.52	26.51
7B	BF16	**35.06**	14.34	23.33	77.72	24.86
	FP4(Ours)	**35.62**	14.29	24.42	78.42	25.36
13B	BF16	**33.69**	12.42	22.45	75.06	24.81
	FP4(Ours)	**33.99**	13.67	21.62	75.84	24.83

Evaluation

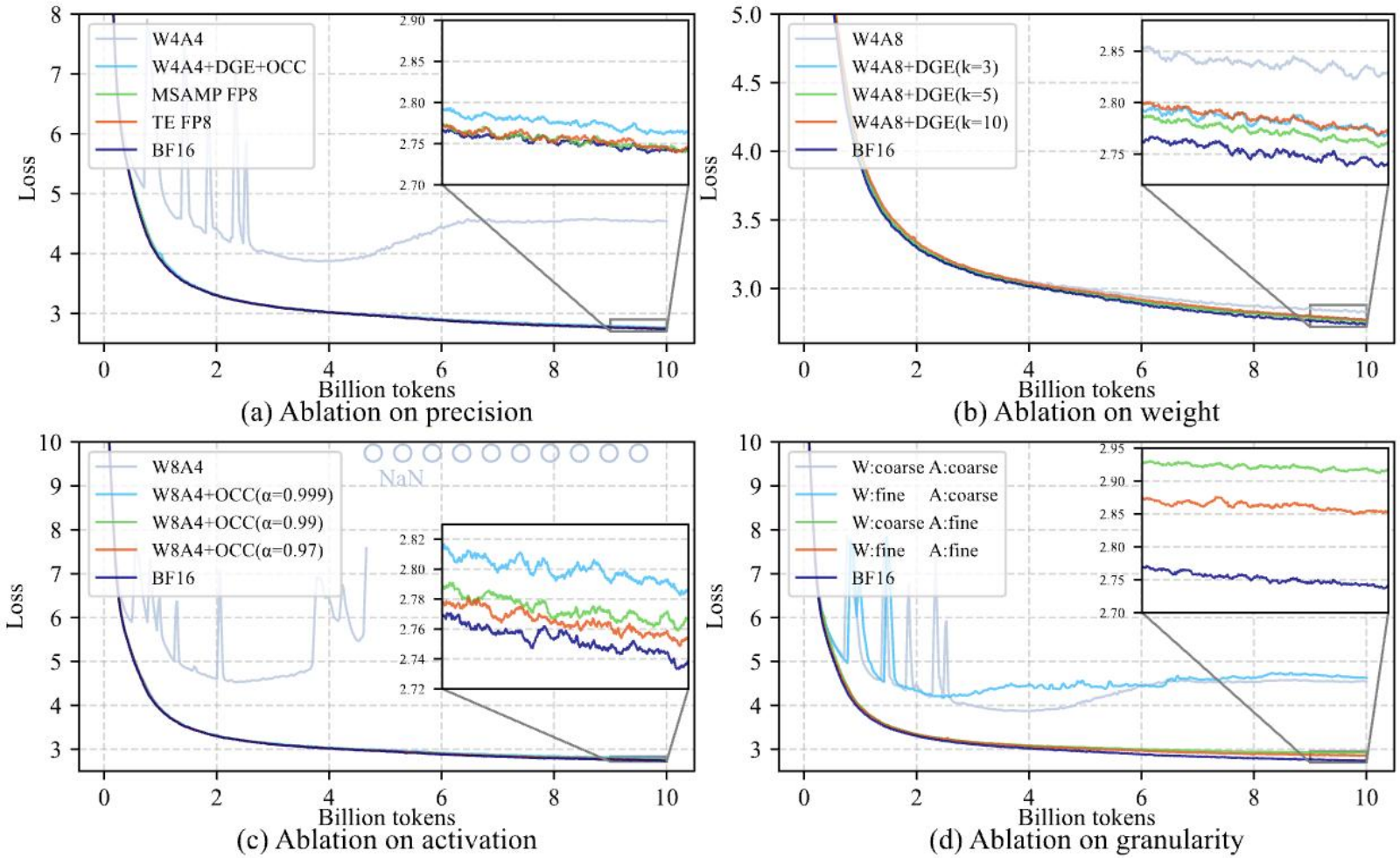
3. 消融实验

① 精度

② 权重

③ 激活值

④ 粒度



Contents

- **Background & Motivation**
- **Methodology**
- **Evaluation**
- **Thinking**

Thinking

- **文章有什么问题，基于这篇 paper 还能做什么优化？**
  - 训练与评测并未在真实 FP4 Tensor Core上完成，而是用 FP8 Tensor Core 模拟 FP4。这会引入与真实硬件不同的舍入模式，导致吞吐/能耗/稳定性和真实 FP4 存在偏差
  - 实验只在 LLaMA 系列模型（1.3B, 7B, 13B）上做，没覆盖到更大的 33B、65B模型
  - Blackwell GPU 出来后，在 FP4 Tensor Core 上跑全流程，看吞吐/能耗/数值稳定性与 FP8 模拟有何差异
  - 使用动态的参数，k与α应根据不同的硬件进行动态调整

Thinking

- **这篇 paper 的 idea 能不能应用在自己的工作上面？**
  - 量化框架”如果要跨硬件，应该支持多种粒度策略（per-tensor / per-channel / per-token）
  - OCC：可以改造成 “动态 clipping + sparse compensation”，和 INT 框架的 PACT 技术结合

- **这篇 paper 能不能泛化？**
  - DGE和OCC和 FP4 不绑定，所有低比特量化（包括 INT4、INT2）都会遇到同样问题，因此思路可以直接迁移
  - 激活比权重更难量化，所以必须细粒度（per-token）这点对 INT 也同样成立，目前很多 INT8/INT4 框架也强调 per-channel/per-token scale

Q & A

Presenter: YiFan Hu

2025.9.17