# 8-bit Transformer Inference and Fine-tuning for Edge Accelerators


Jeffrey Yu
Stanford University
Palo Alto, United States
jeffreyy@stanford.edu

Kartik Prabhu
Stanford University
Palo Alto, United States
kprabhu7@stanford.edu

Yonatan Urman
Stanford University
Palo Alto, United States
yurman@stanford.edu

Robert M. Radway
Stanford University
Palo Alto, United States
radway@stanford.edu

Eric Han
Stanford University
Palo Alto, United States
erichan2@stanford.edu

Priyanka Raina
Stanford University
Palo Alto, United States
praina@stanford.edu

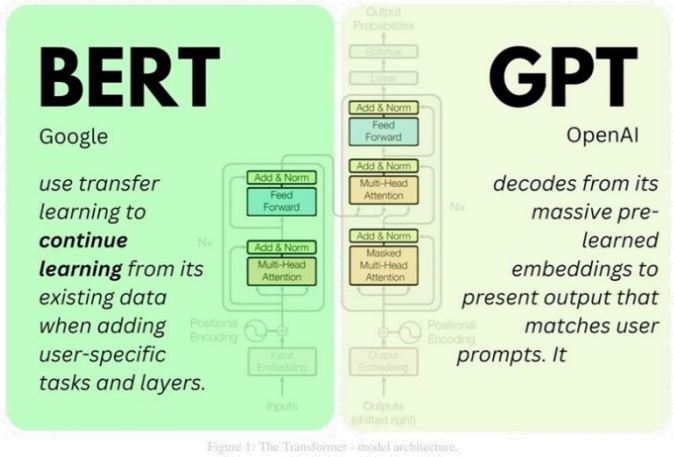

ASPLOS 2024

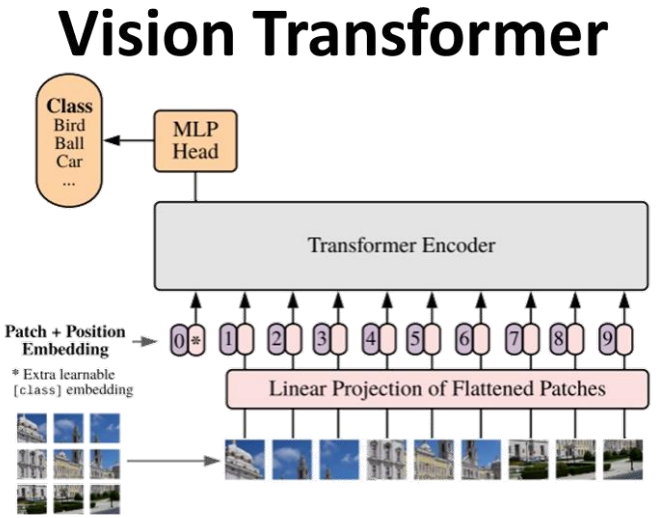
汇报人：姚昌硕
April 11, 2025

# 目录

# 研究背景

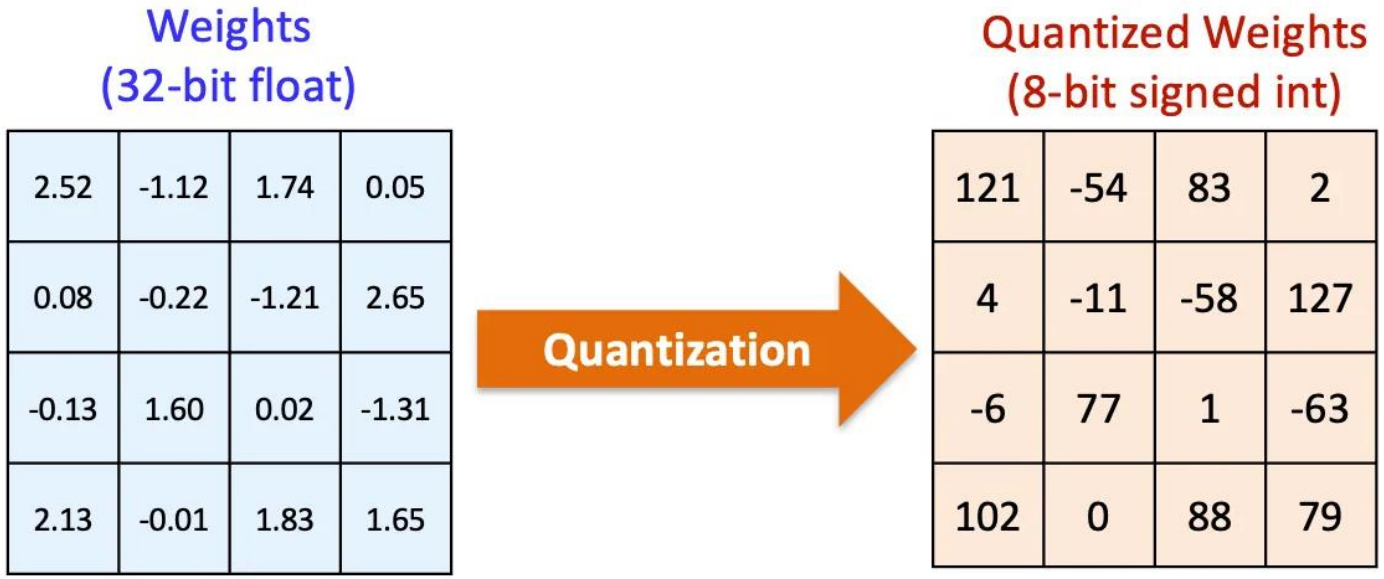
- Transformer：多领域的主流架构
  - NLP领域：BERT、GPT
  - CV领域：ViT、Swin Transformer
- 边缘设备部署Transformer的机遇与挑战
  - 优势：低时延实时推理，高能效比，保护本地数据隐私
  - 挑战：模型参数量庞大，计算复杂度高，内存和带宽需求大
  - 解决方案：**量化**



# 研究背景

- 量化：
  - 定义：将浮点权重/激活值（如FP32）转换为**低比特数**（如INT8/FP8）
  - 优势：
    - 压缩模型存储尺寸
    - 降低内存和带宽需求占用
    - 提升推理速度
    - 优化能耗效率（TOPS/W）



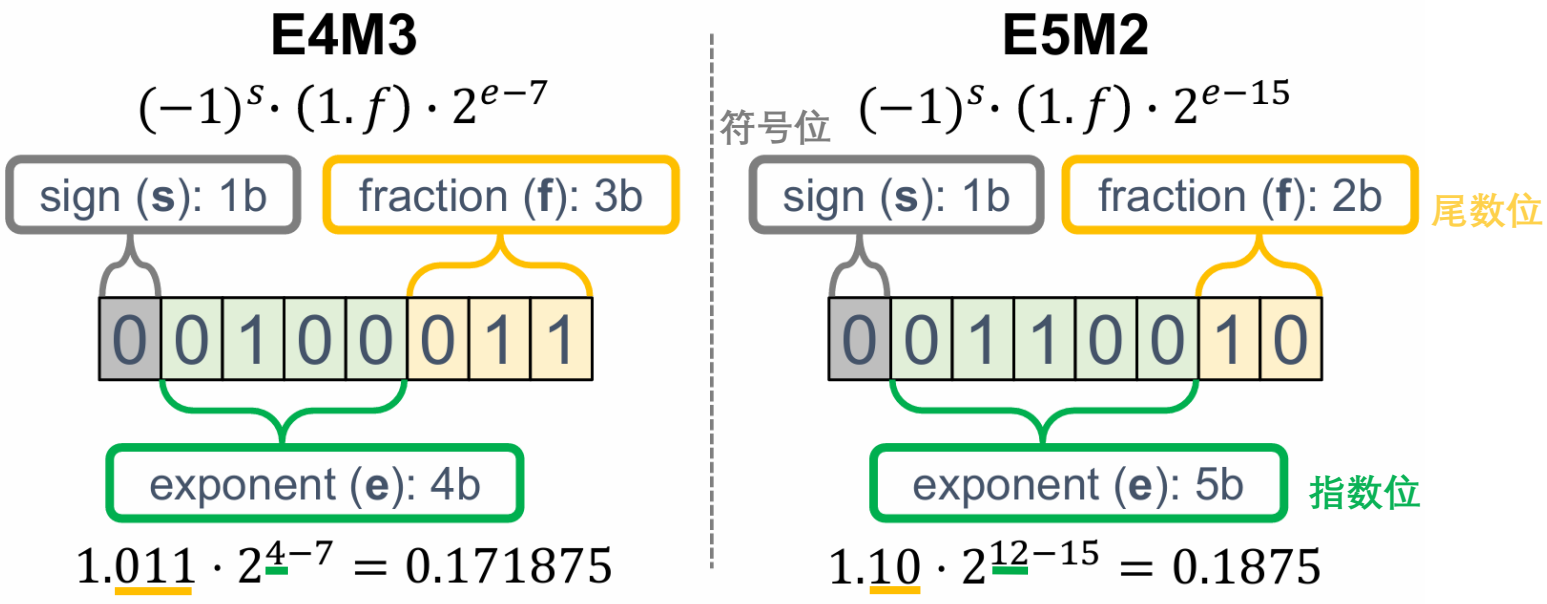
# 相关工作

- **INT8 量化** 如 SmoothQuant、LLM.int8()
  - 仅量化 GEMM 矩阵乘算子；需额外技巧保持精度，专注推理
- **QLoRA**
  - 计算仍使用高精度浮点
- **Nvidia FP8**
  - 限于 GEMM 算子

| | SmoothQuant | LLM.int8() | QLoRA | Nvidia FP8 | 本文方法 |
|---|---|---|---|---|---|
| 数据类型 | Int8 | Int8 | 4-bit NormFloat | E4M3 E5M2 | Posit8, FP8 |
| 推理/训练支持 | 推理 | 推理 | 推理/训练 | 推理/训练 | 推理/训练 |
| 量化的算子 | GEMM | GEMM | 无 | GEMM | 全部 |

# FP8

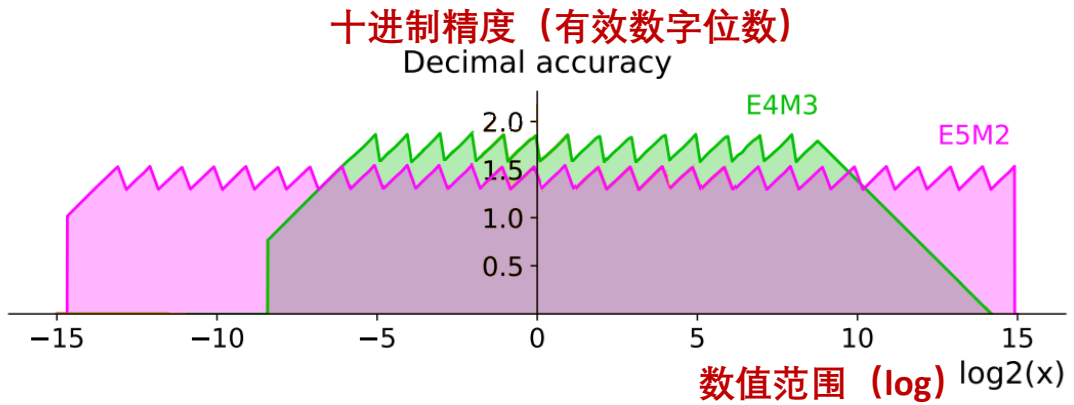
- 当前主流的FP8格式主要有两种变体，E4M3和E5M2
- **E4M3**：1符号位，**4**指数位，**3**尾数位（指数偏置为7 ）
- **E5M2**：1符号位，**5**指数位，**2**尾数位（指数偏置为15 ）



# FP8

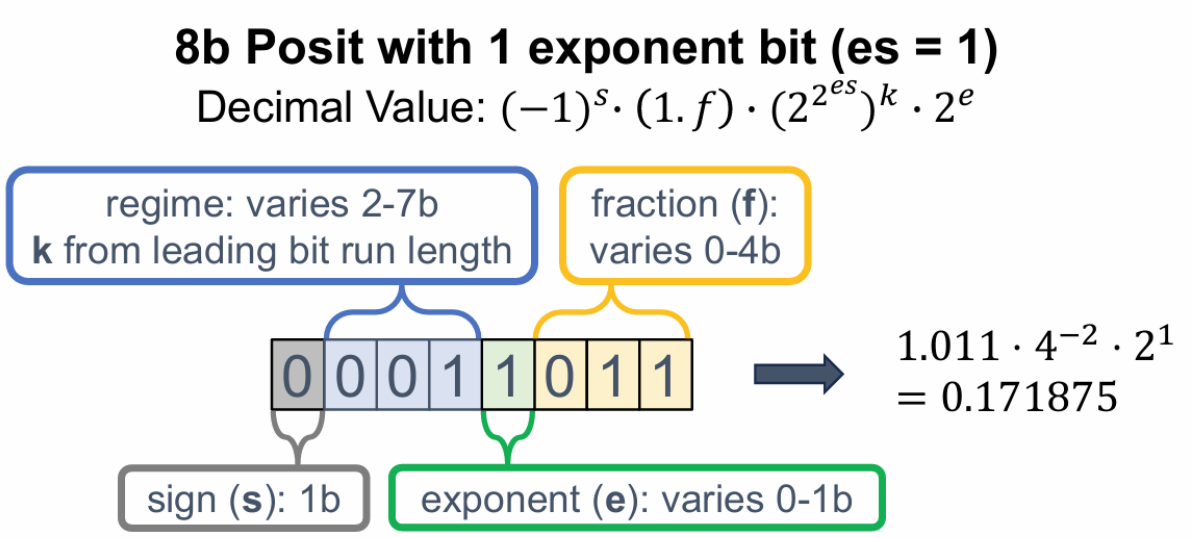
- **E4M3**：精度更高，动态范围更窄 → **推理**
- **E5M2**：精度更低，动态范围更宽 → **训练**

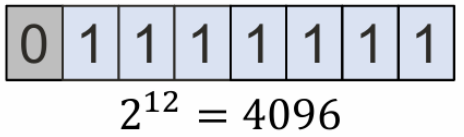
| 特性 | E4M3 | E5M2 |
| --- | --- | --- |
| 最小正数 | $2^{-6}$ | $2^{-14}$ |
| 最大正数 | 448 | 57344 |
| 精度 | 高 | 低 |
| 数值范围 | 窄 | 宽 |
| 适用于 | 前向传播<br>（激活值、权重） | 反向传播<br>（梯度计算） |



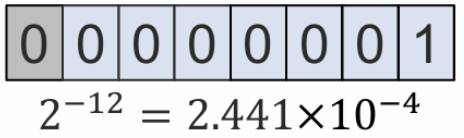
# Posit8

- 增加**可变长度**的指数域（regime）
- “锥形”精度：对极大/小数更强的表达能力，[-1, 1]区间更高的精度
- 适配神经网络中的数值分布

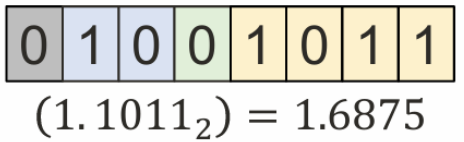



大数：所有bit用于regime




小数：所有bit用于regime




接近1的数：大部分bit用于尾数

# 8-bit Transformer推理

- 相关工作的不足：仅对GEMM进行量化
- 本文方法：量化了**所有算子**
  - 包括Attention Scaling，Softmax, LayerNorm，激活函数，残差连接
  - 通过加速器硬件上的Vector Unit实现

| | SmoothQuant | LLM.int8() | QLoRA | Nvidia FP8 | 本文方法 |
|---|---|---|---|---|---|
| 数据类型 | Int8 | Int8 | 4-bit NormFloat | E4M3 E5M2 | Posit8, FP8 |
| 推理/训练支持 | 推理 | 推理 | 推理/训练 | 推理/训练 | 推理/训练 |
| 量化的算子 | GEMM | GEMM | 无 | GEMM | 全部 |

# 8-bit Transformer推理

- 逐元素算子对量化敏感
  - 量化 → 精度下降
  - 不量化 → 内存读写频繁
- 解决方法：通过**算子融合**提升模型精度，减少内存读写

绿色：GEMM算子
黄色：逐元素算子

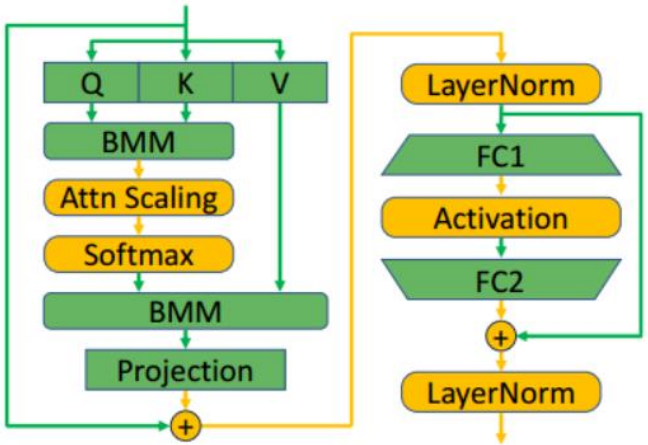
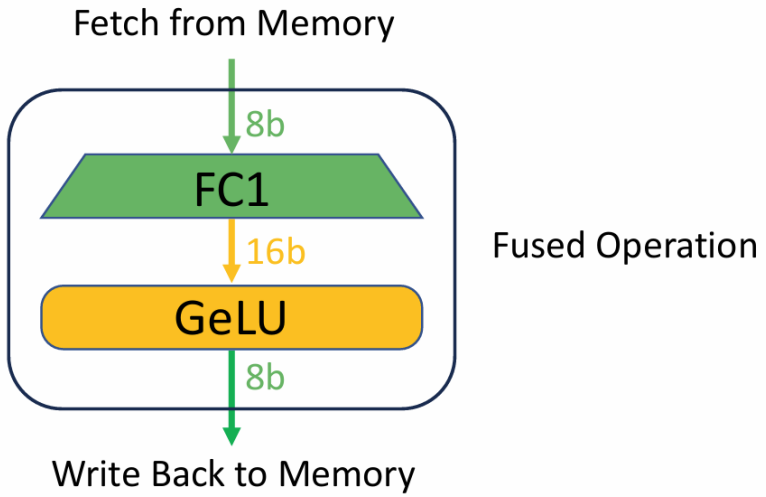



**Figure 5.** A typical Transformer block. GEMM operations are colored green while non-GEMM operations are yellow.



# 8-bit Transformer推理

- **实验：**将不同算子量化为Posit8，对SQuAD v1.1问答数据集F1分数的影响
- **实验结果：**
  - Attn Scaling 对模型精度的影响最大
  - BERT（大模型）对量化更鲁棒，而MobileBERT（小模型）更敏感

对精度影响从小到大

| Operations | MobileBERT | BERT |
|---|---|---|
| BF16 | 89.9 | 88.2 |
| GEMM | 89.4 | 88.1 |
| GEMM + Residual | 89.0 | 88.1 |
| GEMM + LayerNorm | 88.7 | 88.1 |
| GEMM + Activation | 86.7 | 88.1 |
| GEMM + Attn Scaling | 70.4 | 87.4 |

**Table 1.** Accuracy impact of quantizing different Transformer operations to Posit8 in MobileBERT and BERT. The table shows F1 scores on the SQuAD v1.1 dataset.

# Posit8的近似计算

- 对于输入向量 $z = [z_1, z_2, \cdots, z_K]$，Softmax的第$i$个输出为

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

- 分解为**指数运算**（$e$^$z$）和**倒数运算**（$1/x$）
- 通过位运算分别对指数运算和倒数运算进行近似
  - 倒数：保留符号位，其他取反
  - 指数：$e^x \approx \frac{1}{Sigmoid(-x)} - 1$ （Sigmoid通过位取反+右移实现近似）
- 加速器芯片上，无需专用的除法器和指数运算单元

# 8-bit Transformer推理

- **实验：**不同级别的算子融合，对SQuAD v1.1问答数据集F1分数的影响
- **实验结果：**
  - MobileBERT（小模型）需要更多的算子融合来保持精度
  - Posit8并非一定更优于FP8，要根据具体的模型进行选择

数值越高越好

| Model | Size | BF16 | No Fusion | | GEMM + Attn Scaling Fusion | | + Activation Fusion | | + LayerNorm Fusion | | + Residual Fusion | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Posit8 | E4M3 | Posit8 | E4M3 | Posit8 | E4M3 | Posit8 | E4M3 | Posit8 | E4M3 |
| $\text{MobileBERT}_{\text{tiny}}$ | 16M | 88.8 | 86.3 | 87.0 | 87.4 | 87.1 | 87.7 | 87.5 | **87.9** | 87.8 | 88.4 | **88.1** |
| MobileBERT | 25M | 89.9 | 65.1 | 82.7 | 85.0 | 84.9 | 88.3 | 86.7 | **89.0** | 87.9 | 89.4 | 88.6 |
| $\text{DistilBERT}_{\text{base}}$ | 66M | 86.9 | **86.2** | **86.1** | 86.4 | 86.1 | 86.7 | 86.4 | 86.7 | 86.5 | 86.7 | 86.5 |
| $\text{BERT}_{\text{base}}$ | 109M | 88.2 | 87.1 | **87.7** | **88.1** | 88.0 | 88.1 | 88.0 | 88.1 | 88.0 | 88.1 | 88.0 |
| $\text{BERT}_{\text{large}}$ | 334M | 93.2 | 92.3 | **93.0** | **92.8** | 93.1 | 93.0 | 93.1 | 93.0 | 93.2 | 93.1 | 93.1 |

**Table 2.** Transformers' F1 scores on SQuAD v1.1 using Posit8 and FP8 with varying levels of operation fusion. Figures in bold indicate the minimum fusion level needed to achieve within 1% accuracy drop. For MobileBERT models, we need to fuse all operations to achieve within 1% drop. For BERT models, we can easily achieve the same goal even without any fusion.

# 8-bit Transformer推理

- 实验：Whisper模型

数值越低越好

| Model | BF16 | Data Type | No Fusion | Fuse GEMM + Attn Scaling | + Activation Fusion | + LayerNorm Fusion | + Residual Fusion |
|---|---|---|---|---|---|---|---|
| $\text{Whisper}_{\text{tiny}}$ (39M) | 7.54 | Posit (8, 1) | 10.42 | 9.67 | 9.50 | 9.65 | 9.97 |
| | | Posit (8, 2) | 9.39 | 9.26 | 9.87 | 8.87 | 8.22 |
| | | E4M3 | 10.64 | 9.88 | 11.20 | 9.70 | 8.48 |
| $\text{Whisper}_{\text{small}}$ (244M) | 3.41 | Posit (8, 1) | 3.52 | 3.67 | 3.50 | 3.62 | 3.49 |
| | | Posit (8, 2) | 3.71 | 3.69 | 3.68 | 3.63 | 3.53 |
| | | E4M3 | 3.62 | 3.58 | 4.01 | 3.48 | 3.41 |
| $\text{Whisper}_{\text{large}}$ (1550M) | 2.17 | Posit (8, 1) | 2.26 | 2.35 | 2.30 | 2.66 | 2.15 |
| | | Posit (8, 2) | 2.34 | 2.38 | 2.48 | 2.37 | 2.13 |
| | | E4M3 | 3.06 | 2.41 | 2.95 | 2.39 | 2.14 |

**Table 5.** Whisper models' word error rate (WER) on LibriSpeech, with different levels of operation fusion and data types.

# 8-bit Transformer推理

- **实验：LLM模型**

数值越低越好

| Model | BF16 | Data Type | No Fusion | Fuse GEMM + Attn Scaling | + Activation Fusion | + LayerNorm Fusion | + Residual Fusion |
|---|---|---|---|---|---|---|---|
| GPT-2 Large (762M) | 16.38 | Posit (8, 1) | 18.00 | 17.75 | 17.50 | 17.50 | 16.63 |
| | | Posit (8, 2) | 17.50 | 17.50 | 17.50 | 17.50 | 16.63 |
| | | E4M3 | 17.13 | 17.13 | 17.13 | 17.13 | 16.63 |
| GPT-2 XL (1.5B) | 14.69 | Posit (8, 1) | 18.00 | 17.75 | 17.75 | 17.50 | 14.94 |
| | | Posit (8, 2) | 17.75 | 17.75 | 17.75 | 17.75 | 14.94 |
| | | E4M3 | 15.63 | 15.63 | 15.63 | 15.63 | 14.94 |
| LLaMA 2 (7B) | 5.19 | Posit (8, 1) | 5.56 | 5.53 | 5.53 | 5.52 | 5.30 |
| | | Posit (8, 2) | 5.44 | 5.40 | 5.38 | 5.37 | 5.29 |
| | | E4M3 | 5.80 | 5.80 | 5.77 | 5.75 | 5.36 |
| LLaMA 2 (13B) | 4.63 | Posit (8, 1) | 4.85 | 4.78 | 4.78 | 4.77 | 4.72 |
| | | Posit (8, 2) | 4.86 | 4.82 | 4.81 | 4.80 | 4.72 |
| | | E4M3 | 5.10 | 5.09 | 5.07 | 5.06 | 4.73 |

**Table 6.** Perplexity of LLMs on WikiText-103 using Posit (8, 1), Posit (8, 2), and FP8 with incremental levels of operator fusion.

# 小结

- **通过算子融合，Posit8和FP8都能获得与BF16相近的精度**
- **小模型量化难度更大，需要更多的算子融合**
- **数据类型的选择（Posit8/FP8）取决于具体的模型和任务**
  - 例如，Posit8的高动态范围更适合LLM

# 8-bit Transformer微调

- **相关工作：**
  - 只对GEMM输入进行量化，中间结果以16bit存储
- **本文的贡献：**
  - 量化全部算子
  - 中间结果压缩至8bit存储

| | QLoRA | Nvidia FP8 | 本文方法 |
|---|---|---|---|
| 数据类型 | 4-bit NormFloat | E4M3 E5M2 | Posit8, FP8 |
| 推理/训练支持 | 推理/训练 | 推理/训练 | 推理/训练 |
| 量化的算子 | 无 | GEMM | 全部 |

# Per-Tensor Scaling

- **推理阶段（传统Int8）：**
  - 在模型部署前，使用少量校准数据，对**缩放系数(scale)**进行校准
  - 校准完成后，scale为**静态**，与输入数据无关 (Post-Training Quantization)
- **推理阶段（本文）：**
  - 选取合适的数据类型，多数情况下**无需额外的缩放**
- **训练阶段:**
  - 梯度的分布范围更广，超出表示范围
  - **需要进行缩放**

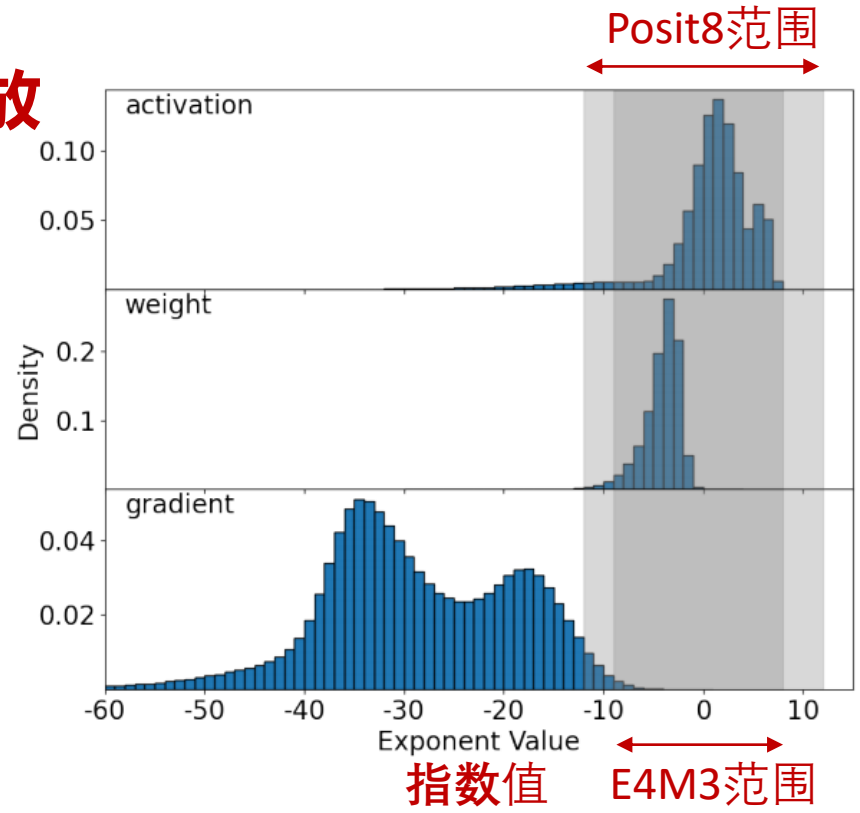
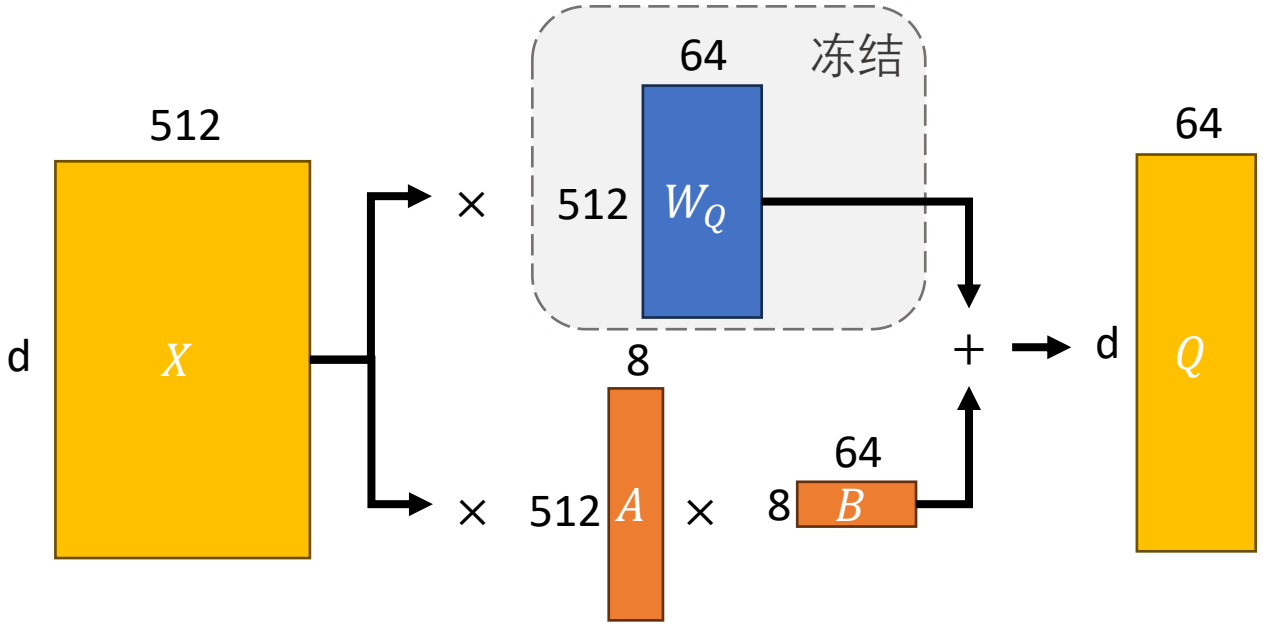
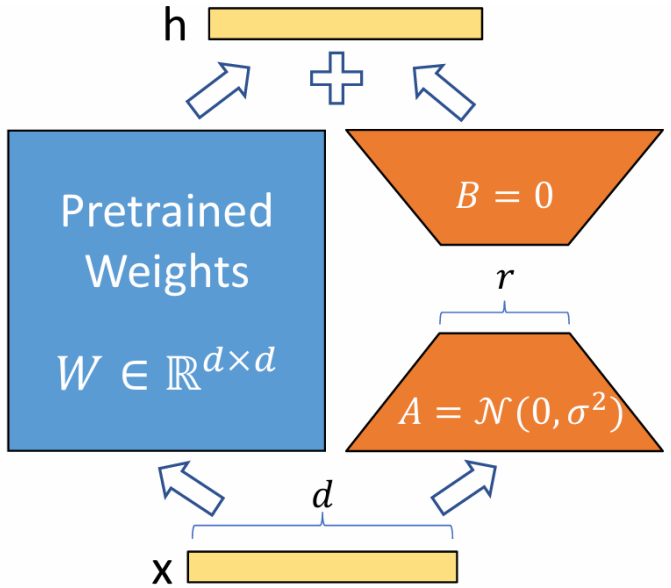



**Figure 10.** Tensor value distributions during fine-tuning of MobileBERT on SQuAD. The darker gray region indicates the span of E4M3, whereas the area encompassed by both light and dark gray represents the range achievable by Posit8. While both E4M3 and Posit8 cover the range of activations and weights, they fail to cover the activation gradients.

# LoRA

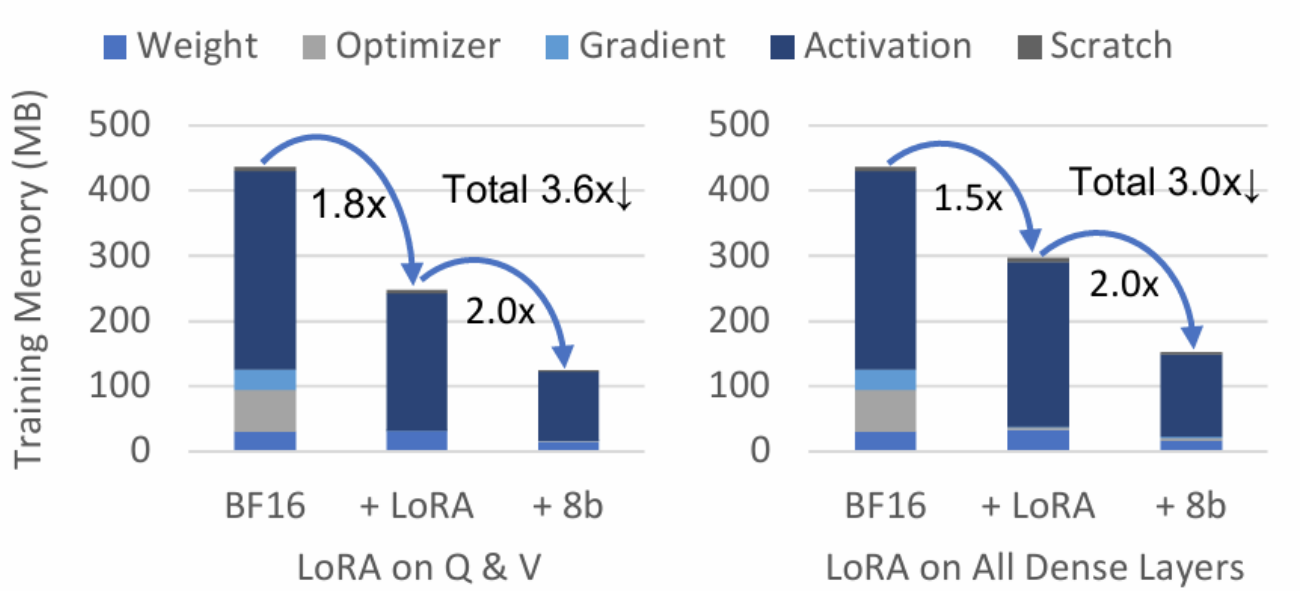
- LoRA（Low-Rank Adaptation）是一种参数高效的微调方法
  - 基本假设：预训练权重$W$可能满秩，但微调更新$\Delta W$具有低内在秩
  - 通过对$\Delta W$进行低秩分解，减少需要更新的参数的数量



# 8-bit Transformer微调

- LoRA：减小训练内存1.8倍
- 8bit量化：减小训练内存2倍
- 两种方法结合：减小训练内存3倍



# 8-bit Transformer微调

- 使用FP8和Posit8可以获得和BF16相似的性能
- Softmax的近似对训练无影响

| Model | Method | # Trainable Parameters | Accuracy | | | | |
|---|---|---|---|---|---|---|---|
| | | | MNLI | QNLI | MRPC | SST-2 | SQuAD |
| $\text{MobileBERT}_{\text{tiny}}$ (16.5M) | Full Training FP32 [28] | 15.1M | 82.0 | 89.9 | 86.7 | 91.6 | 88.6 |
| | LoRA BF16 | 0.3M | 82.9 | 90.7 | 88.0 | 91.4 | 88.1 |
| | LoRA Posit8 | 0.3M | 82.2 | 90.6 | 86.8 | 91.4 | 86.4 |
| | LoRA Posit8 Approximation | 0.3M | 82.9 | 90.8 | 87.5 | 91.1 | 86.5 |
| | LoRA FP8 | 0.3M | 81.9 | 90.7 | 87.8 | 90.8 | 87.5 |
| MobileBERT (25.3M) | Full Training FP32 [28] | 25.3M | 83.9 | 91.0 | 87.5 | 92.1 | 90.0 |
| | LoRA BF16 | 0.3M | 83.9 | 91.5 | 87.5 | 92.4 | 89.0 |
| | LoRA Posit8 | 0.3M | 83.3 | 91.5 | 87.8 | 91.7 | 87.4 |
| | LoRA Posit8 Approximation | 0.3M | 83.1 | 91.5 | 87.5 | 92.2 | 88.1 |
| | LoRA FP8 | 0.3M | 83.0 | 91.1 | 87.8 | 91.7 | 87.8 |
| $\text{RoBERTa}_{\text{base}}$ (125.0M) | Full Training FP32 [15] | 125.0M | 87.6 | 92.8 | 90.2 | 94.8 | - |
| | LoRA BF16 | 0.3M | 87.3 | 92.9 | 89.2 | 94.7 | 91.5 |
| | LoRA Posit8 | 0.3M | 87.1 | 92.5 | 89.5 | 94.6 | 91.2 |
| | LoRA Posit8 Approximation | 0.3M | 86.9 | 92.5 | 89.0 | 94.4 | 91.1 |
| | LoRA FP8 | 0.3M | 86.8 | 92.9 | 89.5 | 95.0 | 91.2 |
| $\text{RoBERTa}_{\text{large}}$ (355.0M) | Full Training FP32 [15] | 355.0M | 90.2 | 94.7 | 90.9 | 96.4 | 94.6 |
| | LoRA BF16 | 0.8M | 90.3 | 94.5 | 91.7 | 96.2 | 94.6 |
| | LoRA Posit8 | 0.8M | 90.0 | 94.3 | 91.9 | 96.0 | 94.0 |
| | LoRA Posit8 Approximation | 0.8M | 90.0 | 94.3 | 91.2 | 96.0 | 93.7 |
| | LoRA FP8 | 0.8M | 89.9 | 94.6 | 90.2 | 96.1 | 94.1 |

# Idea

- **与此类似：**
  - 将本文的高效8bit微调方法用于其他模型
  - 将其他微调技术用于本文的8-bit情形
- **由此需要：**
  - 文中提到，不同的模型，不同的任务，适配的数据类型不同
  - Posit(8, 1) Posit(8, 2) E4M3 E5M2
  - 开发一种自适应量化机制，能够根据每层或每个操作时的激活分布自动调整数据类型

# 8-bit Transformer Inference and Fine-tuning for Edge Accelerators


Jeffrey Yu
Stanford University
Palo Alto, United States
jeffreyy@stanford.edu

Kartik Prabhu
Stanford University
Palo Alto, United States
kprabhu7@stanford.edu

Yonatan Urman
Stanford University
Palo Alto, United States
yurman@stanford.edu

Robert M. Radway
Stanford University
Palo Alto, United States
radway@stanford.edu

Eric Han
Stanford University
Palo Alto, United States
erichan2@stanford.edu

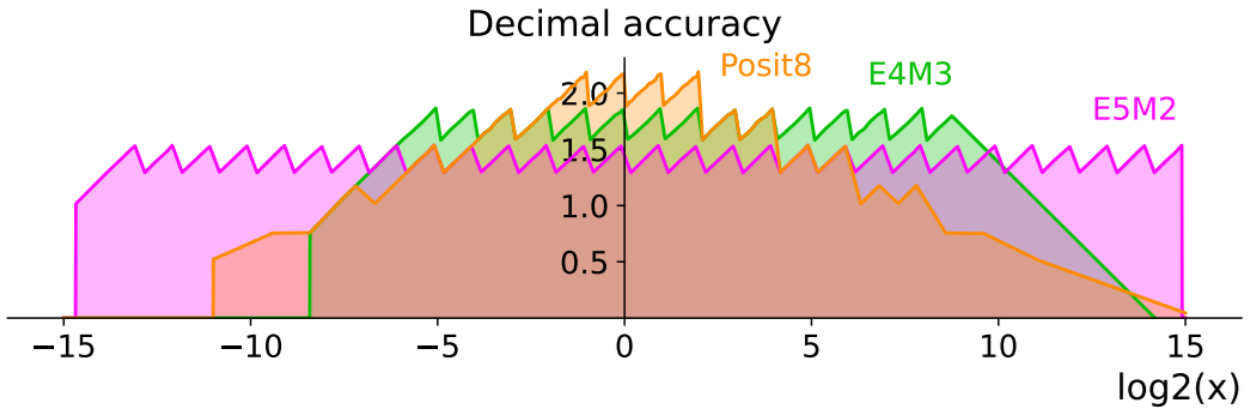
Priyanka Raina
Stanford University
Palo Alto, United States
praina@stanford.edu


ASPLOS 2024

汇报人：姚昌硕
April 11, 2025

# Backup Slides

# FP8 vs Posit8

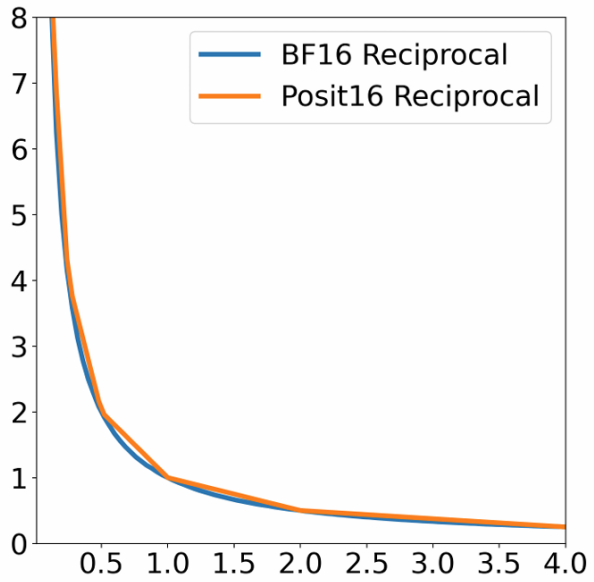


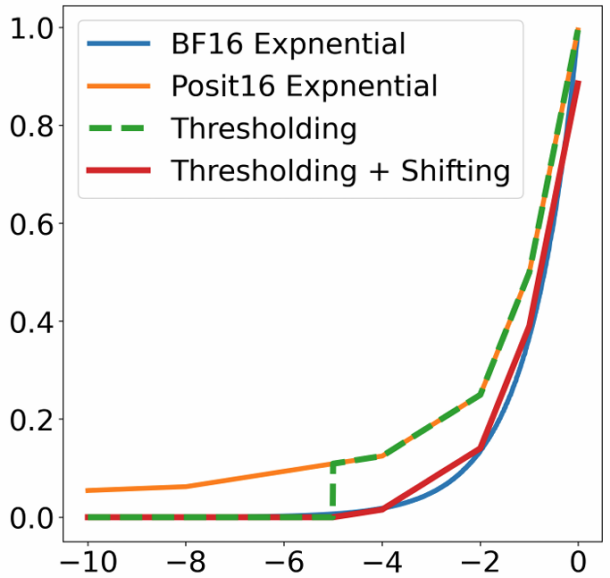

| | Posit8 | E4M3 | E5M2 |
|---|---|---|---|
| Max | 0 1 1 1 1 1 1 1 | 0 1 1 1 1 1 1 0 | 0 1 1 1 1 0 1 1 |
| | $2^{12} = 4096$ | $1.75 * 2^{8} = 448$ | $1.75 * 2^{15} = 57344$ |
| Min | 0 0 0 0 0 0 0 1 | 0 0 0 0 0 0 0 1 | 0 0 0 0 0 0 0 1 |
| | $2^{-12} = 2.441\times10^{-4}$ | $2^{-9} = 1.953\times10^{-3}$ | $2^{-16} = 1.526\times10^{-5}$ |
| Near 1 | 0 1 0 0 1 0 1 1 | 0 0 1 1 1 1 0 1 | 0 0 1 1 1 1 1 0 |
| | $(1.1011_2) = 1.6875$ | $(1.101_2) = 1.625$ | $(1.10_2) = 1.5$ |

# Posit8的近似计算

- 近似倒数：保留符号位，其他取反
- 近似Sigmoid：符号位取反，其余部分右移2bit，补0
- 近似指数：

$$e^x \approx \frac{1}{Sigmoid(-x)} - 1$$



# 加速器上的专用处理单元

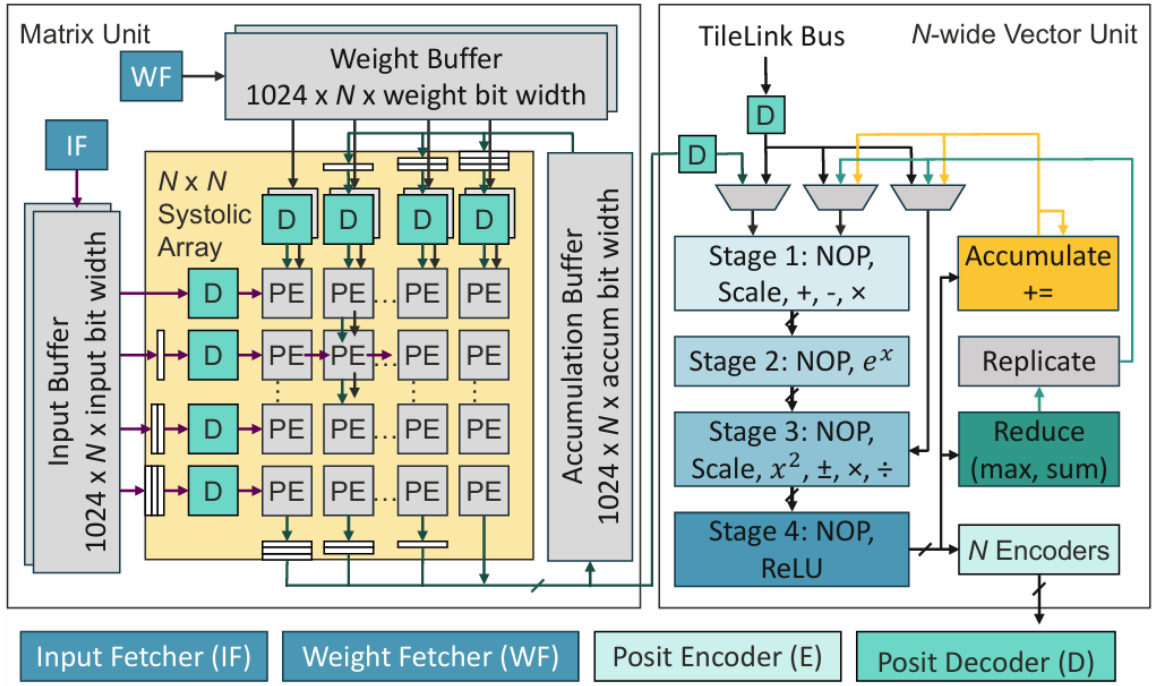



**Figure 11.** A standard neural network accelerator featuring a systolic array for matrix multiplication and a vector unit dedicated to executing element-wise operations and vector reductions. The encoders and decoders are only needed for posit-based accelerators. NOP means no operation.

# LoRA

- **相关工作** LoRA & QLoRA
- GEMM使用16bit浮点数计算
- 合并权重后需要重新量化
  - 导致精度损失

$$h = \text{dequant}(W_0)x + \alpha \cdot BAx$$

- **本文方法**
- GEMM使用8bit计算
- 不需要重新量化
  - 无精度损失

$$h = \text{quant}\left(W_0 + \alpha \cdot \text{quant}(B)\text{quant}(A)\right)x$$