一种基于内存对齐的大模型混合精度量化方法

李章明[1]，关伟凡[1]，常政威[2]，张凌浩[2]，胡庆浩[1]

(1. 中国科学院自动化研究所复杂系统认知与决策重点实验室，北京 100190；
2. 国网四川省电力公司，四川 成都 610041)

摘　　要：随着大模型规模的不断增长，模型推理的内存占用和计算开销成为重要挑战。模型量化是降低模型资源消耗的有效方法，但现有方法在权重量化过程中存在离群点处理不足、量化精度损失显著以及内存访问效率低下等问题。为此，提出一种内存对齐的大模型混合精度量化方法，通过将模型参数表示成不同位宽的量化参数实现混合精度量化方法，在降低模型存储的同时缓解量化带来的精度损失问题。具体来说，基于小组显著性分析划分权重离群点，将模型参数按单指令多数据流(SIMD)单元对齐分组，并依据显著性对不同小组采用 8 bit 或 2 bit 量化；针对 2 bit 量化可能导致的精度损失，引入分块量化补偿策略。此外，设计了一种高效的混合精度权重打包与存储方案，通过位图(Bitmap)记录数据块位宽类型，支持随机访问。实验结果表明，该方法在保证模型精度的同时，显著降低了内存占用并提升了计算效率。通过在 Llama2-7 B，13 B 和 70 B 上进行验证，相比最先进的方法，在 WikiText2 和 C4 数据集上的困惑度(PPL)分别下降 8.13，2.84，1.37 及 5.80，并且量化后的 70 B 模型相对 BF16 权重存储约减 87%。此外在 7 个 QA 数据集上平均准确率提升 6.24%。其结果表明，基于内存对齐的大模型混合精度量化方法能够同时提升压缩率、访存效率与模型性能。

关　键　词：大模型压缩；训练后量化；低比特量化；混合精度量化；离群点划分

中图分类号：TP 391　　**DOI**：10.11996/JG.j.2095-302X.2026010039

文献标识码：A　　**文 章 编 号**：2095-302X(2026)01-0039-09

A mixed-precision quantization method for large language models via memory alignment

LI Zhangming[1], GUAN Weifan[1], CHANG Zhengwei[2], ZHANG Linghao[2], HU Qinghao[1]

(1. The Key Laboratory of Cognition and Decision Intelligence for Complex Systems,
Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China;
2. State Grid Sichuan Electric Power Company, Chengdu Sichuan 610041, China)

Abstract: As large models continue to grow in scale, the memory footprint and computational overhead of model inference have become critical challenges. Mixed-precision quantization is an effective approach to reduce resource consumption, but existing methods suffer from insufficient outlier handling, significant quantization accuracy loss, and inefficient memory access. To address these issues, a memory-aligned mixed-precision quantization method for large models was proposed. First, weights were divided into SIMD-aligned groups, and outlier groups were identified via group-wise significance analysis, with high-significance groups quantized to 8 bit and others to 2 bit. A block-wise compensation strategy was introduced to mitigate accuracy degradation caused by 2 bit quantization. Furthermore, an efficient packing and storage scheme was designed for mixed-precision weights, where a bitmap was used to record the bit width of each data block, enabling random access. Experimental results demonstrated that the proposed method significantly reduced memory usage and improved computational efficiency while maintaining model accuracy.

收稿日期：2025-06-10；定稿日期：2025-10-11；通信作者：胡庆浩，E-mail：huqinghao2014@ia.ac.cn
Received：10 June，2025；**Finalized**：11 October，2025；**Corresponding author**：HU Qinghao，E-mail：huqinghao2014@ia.ac.cn
基金项目：国家电网有限公司科技项目(5700-202426249A-1-1-ZN)
Foundation items：Science and Technology Project of State Grid Corporation of China (5700-202426249A-1-1-ZN)

Specifically, on Llama2-7 B/13 B/70 B, the approach achieved perplexity reductions of 8.13/2.84/1.37 on WikiText-2 and 5.80 on C4 relative to state-of-the-art baselines. The quantized 70 B model reduced weight storage by approximately 87% compared with BF16. Across seven QA benchmarks, an average accuracy gain of 6.24% was achieved. Last, these results indicated that a mixed-precision quantization method for large language models via memory alignment could simultaneously improve compression ratio, memory-access efficiency, and overall model performance.

Keywords: large language model compression; post-training quantization; low-bit quantization; mixed-precision quantization; outlier extraction

近年来，随着深度学习和大规模预训练模型的迅速发展，其在自然语言处理、计算机视觉等领域取得了显著成果。以 GPT (Generative Pre-trained Transformer)，Llama (Large Language Model Meta AI)和DeepSeek等为代表的大模型凭借其强大的表征能力，在各类任务上刷新了性能记录。然而，模型参数数量的激增也带来了巨大的存储和计算压力，成为实际应用中制约大模型部署与推理效率的重要瓶颈。例如，一个 1 750 亿参数的 GPT-3 模型若以 FP32 格式存储，仅权重部分就需要近 700 GB 内存，远超大多数设备的承载能力。同时，大矩阵乘法运算的密集计算需求也对硬件算力提出了极高要求，导致推理延迟、能耗上升，严重限制了在边缘设备上的实际应用。

为了解决这一问题，量化技术通过把高精度浮点数表示压缩为低比特数值表示，不仅能够大幅减少存储占用，还能借助低比特计算单元提升计算吞吐率。但在面向大模型时仍面临 2 个关键挑战：其一，模型权重中普遍存在的离群值(Outliers)，会导致均匀量化的精度急剧下降；其二，极低比特(如 2 bit)量化虽能进一步压缩模型，但会引入显著的量化误差，尤其在权重分布不均匀时更为突出。此外，传统量化方案通常采用全局统一比特位宽，难以兼顾模型不同部分的敏感度差异，导致资源分配低效。

针对上述问题，近年来研究者提出了多种改进方案。如 GOU 等[1]提出了硬件友好的离群值-受害者对(Outlier-Victim Pair, OVP)量化方法，将离群值与其相邻的受害者值配对，通过内存对齐的方式进行编码，提升了推理速度和能效。SHANG 等[2]采用了一种部分二值化的极端低比特量化方法，将大部分模型参数压缩为 1 bit，同时筛选一小部分关键权重保留更高精度。HUANG 等[3]则是采用了结构化显著权重选择与二值残差逼近，对显著权重进行二值化；同时针对非显著权重，引入最优分割搜索，将其按照钟形分布进行分组后二值化。

尽管上述方法在一定程度上缓解了量化带来的性能损失，但在面对极低比特情况时仍存在一定的挑战。为此，本文提出了一种内存对齐的大模型混合精度量化方法，其核心创新包括：

1) 基于小组显著性分析的离群点划分。将权重按 SIMD 单元(如 16 个数值/组)对齐分组，通过显著性度量动态识别高敏感区域，避免离群值干扰；

2) 分层混合精度量化策略。对高显著性组采用 8-bit 组内均值量化，对低显著性组实施 2 bit 分块量化并引入补偿机制，平衡压缩率与精度；

3) 硬件友好的存储设计。通过位图(Bitmap)编码实现混合位宽权重的紧凑打包，确保内存访问。

1 相关工作

1.1 大模型压缩

大模型压缩通常包括剪枝、蒸馏和量化，目标是在尽量保持通用性与任务性能的同时显著降低显存占用、计算开销与推理延迟。剪枝通过删除不重要的参数或结构来引入稀疏性与结构简化：①在无需重训的前提下进行一次性不规则权重剪枝(如SparseGPT[4]和Wanda[5])；②发展出结构化/半结构化剪枝以便获得端到端速度收益。蒸馏采用“教师-学生”范式，将教师模型的概率分布或排序信号迁移至小模型，近年来在生成式LLM上常用反向KL或排序损失以缓解“模式平均”，如MiniLLM[6]展示了在白盒LLM上有效的蒸馏策略。在本节中主要讨论量化方法。量化是指将权重或激活量化成定点数的形式，大模型原始的权值和激活表示通常采用BF16或FP16格式，这种半精度浮点操作的运算很依赖于浮点运算器，而定点运算则有功耗低、运算速度快和存储小等优点，通过量化的方式将权重或激活表示成定点运算并实现大模型的加速和压缩。早期的量化方法[7-9]主要是对卷积神经网络进行压缩与加速，由于大模型的权重分布差异和参数量规模的不同，很多研究工作无法直接使用，进而诞生

了大模型量化方法研究。根据是否需要使用原始训练数据对大模型进行量化微调可以分为量化感知训练(Quantization-Aware Training, QAT)方法和训练后量化(Post-Training Quantization, PTQ) 方法。在QAT方面，LLM-QAT[10]提出了一种无需数据的大模型QAT方法，通过预训练模型产生的输出构建微调数据，此外，还同时对键-值缓存进行了压缩。由于大模型微调代价较高，基于QAT的大模型量化算法较少，多数大模型量化算法都是PTQ方法。

1.2　训练后量化方法

PTQ方法根据是否量化激活可以分为权重量化(Weight-only)和激活-权重(Activation-weight)联合量化2种方式，前者可以做到较低的量化位宽(4 bit以下)，在推理时以降低内存消耗为主，也有少量加速推理的作用。后者可以低比特计算指令集进行加速，目前以8 bit量化为主，低位宽(4 bit)下的激活-权重联合量化会造成不可忽视的精度损失。在大模型权重量化方面，GPTQ[11] (Gradient-based Post-Training Quantization)方法发现权重量化到低位宽的时候会出现精度下降严重的问题，提出了一种基于近似二阶信息的逐层量化技术，并针对大模型庞大的参数量问题进行了改进和优化，每个权重的位宽可以几乎精度无损地减少到3位或4位。LRQ[12](Low-Rank Quantization)方法通过使用低秩的权重缩放矩阵(Weight-scaling matrices)来替代传统的全参数缩放方式，实现参数共享，并允许个别权重进行调整，从而提升量化后模型的泛化与性能。GuidedQuant[13]方法发现现有的后训练量化技术在权重量化到低位宽时往往忽视了不同隐藏特征对最终任务损失的影响，且常用的方法在引入末端损失指导时又忽略了权重间的重要关联。为此，提出了一种基于末端损失梯度指导的逐层量化技术，并在输出通道内保留了权重间的交互依赖，从而提升权重量化的效果。在激活-权重联合量化方面，LLM.int8()[14]方法采用向量量化(Vector-wise)将模型激活和权重同时进行8比特量化，有效地将推理内存占用减半，并采用了混合精度量化来处理大模型中的离群点问题，并将离群点表示为高位量化精度，控制了激活与权重同时量化带来的精度损失。ZeroQuant[15]提出了硬件友好的量化方案，采用逐层的知识蒸馏对激活和权重同时量化到8 bit，缓解了激活低比特量化困难的问题。SmoothQuant[16]观察到不同的标记在其通道中表现出相似的变化，并引入了一种跨通道缩放变换，可以有效地平滑激活的幅值，使模型更易于量化。OmniQuant[17]则提出了可学习权重裁剪(Learnable Weight Clipping, LWC)和可学习等效变换(Learnable Equivalent Transformation, LET)，整体采用可微的计算范式，基于块级的输出误差最小化目标函数，在一细列的量化实验中都取得了良好效果。ZeroQuant-V2[18]引入了一种称为低秩补偿(Low-Rank Compensation, LoRC)的技术，由于低比特量化误差较大，该技术对误差矩阵进行低秩分解，得到低秩的浮点因子。由于在定点矩阵乘法的基础上增加了低秩的浮点计算，模型参数量和计算量都有所增加。

1.3　混合精度量化

为了进一步提升大模型在低比特环境下效果，研究者们考虑混合精度量化，即对模型最终效果影响较大的参数进行高比特量化或保留原始参数值，对于不重要的参数进行低比特量化。例如，AWQ[19](Activation-aware Weight Quantization)发现不同权重的量化对大模型性能的影响是不同的，仅保护 1%的显著权重就可以大量降低量化误差。因此，提出了激活感知的权重量化方法，将大幅值激活对应的权重通道的重要性考虑在内，并结合了通道缩放技术来降低离群点对量化精度的影响。而SpQR[20](Sparse-Quantized Representation)对权重离群点进行高位宽存储，其余权重量化到低位宽(如 3 bit)，同时采用了稀疏的存储格式对离群点进行存储，并对低位宽量化采用细粒度的分组量化。QuIP[21](Quantization with Incoherence Processing)则是采用了自适应舍入过程，对一个二次的代理目标函数进行最小化，同时提出了高效的预处理和后处理方法，通过随机正交矩阵的乘法来确保权值和 Hessian 矩阵不相干，最后实现了大模型权重的 2 bit 量化。Norm Tweaking[22]主要是通过校正量化激活的分布以匹配其浮点的对应分布，这有助于恢复大模型的准确性，该方法还提出了校准数据生成方式和通道距离约束来更新归一化层的权重，以便更好地提升量化方法的泛化性能。Olive[1]方法采用异常值-受害者对的量化方法，以较低的硬件开销处理异常值，同时发现了离群点很重要但其附近的正常值却不重要的现象。QuantEase[23]将量化问题重表示为离散的非凸优化问题，并通过坐标下降法来进行求解。RPTQ[24](Reorder-based Post-training Quantization)发现现激活的离群点主要发生在不同的通道上，因此提出了一种基于通道重排序的方

法，将通道按照数值范围排序并聚类，有效地减轻了通道间的数值范围差异。此外，该方法还将重排序操作集成到层归一化运算(LayerNorm)和线性层权值中，以降低通道重排序的开销。PBLLM[2](Partially-binarized LLM)方法发现将大规模语言模型权重极度二值化时(1 bit)会严重损害模型的语言推理能力，因此提出了一种部分二值化(Partially-binarized)策略，通过筛选少量核心权重保留更高位宽，并采用近似二阶信息指导逐层重构与补偿，使模型在极端量化下仍能恢复推理能力。BILLM[3](1-bit PTQ Framework for LLMs)方法观察到在超低比特(1 bit)权重量化下，现有方法普遍性能下降严重，因此提出一种基于后训练量化的二值框架，通过结构化选择显著权重与二值残差逼近策略，并对非显著权重采用最优分组法精确二值化，实现了极低位宽下的推理。

2 方法

内存对齐的混合精度量化算法包含3个关键模块：基于二阶信息的离群参数分组算法、分层混合精度量化和混合精度权重的打包与存储。这些模块共同作用，在保证模型精度的同时显著提升推理效率。

2.1 基于二阶信息的离群参数分组算法

由于量化将连续数值压缩到有限的离散级别，所采用的比特宽度直接决定了模型压缩倍率和模型最终精度。量化位宽越高，模型精度保持越高，其压缩倍率越低，反之亦然。

为了平衡大模型量化过程中存在的精度与效率之间的矛盾，本文通过显著性分析，将权重分为2类：高显著性权重和低显著性权重。本文提出了一种基于二阶信息的离群点分组算法，用于更准确地识别和处理关键权重。不同于传统算法对单个参数或整个通道参数进行显著性分析，并提出 SIMD 单元友好的权重分组策略，以 SIMD 处理单元一次性处理的数据个数为分组大小。

具体来说，首先根据每一层的激活 $\boldsymbol{X}$ 求 Hessian 矩阵，即

$$\boldsymbol{H}^{-1}=\left(2\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}+\lambda\boldsymbol{I}\right)^{-1} \tag{1}$$

式中：$\boldsymbol{X}$ 表示激活；$\boldsymbol{I}$ 表示单位矩阵；λ 表示超参数。然后将每层的权重参数 $\boldsymbol{W}\in\boldsymbol{R}^{T\times P}$ 按照与 SIMD 单元大小对齐的小组进行划分(如每组包含 m 个数值，典型的 SIMD 单元大小为 16)，如图 1 所示，将 W 的输入维度按照每 m 行为一块进行划分，一共有 n 块($\boldsymbol{W}_1,\cdots,\boldsymbol{W}_n$)，其中 $\boldsymbol{W}_i\in\boldsymbol{R}^{m\times P}$，$\boldsymbol{W}_i$ 中的每一列为一个小组，然后计算每个小组的显著性

$$\boldsymbol{S}_i=\sum_j\frac{\boldsymbol{w}_{ji}^2}{\left[\boldsymbol{H}^{-1}\right]_{ii}^2} \tag{2}$$

式中：i 表示第 i 组；j 表示第 i 组元素的位置。然后根据显著性高低排序，选出前 K 个显著性高的小组在标志位图(Bitmap)上标注为显著性权重，其中标志位图是一个布尔矩阵，用于标注每组权重是否为显著性权重。

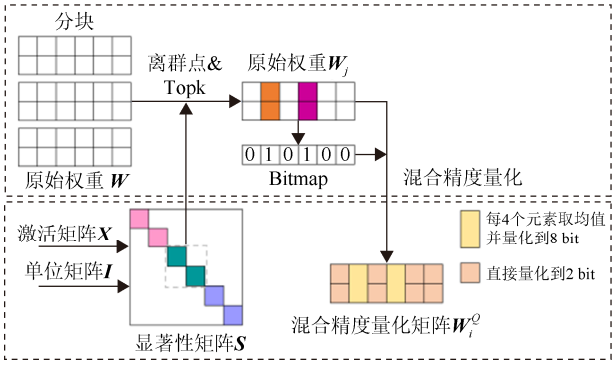

图 1 基于二阶信息的离群参数分组算法示意图

Fig. 1 Schematic diagram of outlier parameter grouping algorithm based on second-order information

为了支持后续的混合权重打包，需要对每 4 个异常值为一组求均值，并量化到 8 bit。

模型量化通常采用仿射变换的方式将原始模型的参数 $\boldsymbol{X}$ (或激活)映射到较低比特宽度的离散数值空间，即

$$\boldsymbol{X}_q=clamp\left(round\left(\frac{\boldsymbol{X}}{\boldsymbol{s}}\right)+\boldsymbol{z},range\right) \tag{3}$$

式中：$\boldsymbol{s}$ 表示缩放因子；$\boldsymbol{z}$ 表示零点；$round$ 表示取整函数；$clamp$ 表示限制函数取值范围在低比特表示空间内；$range$ 表示低比特空间范围，反量化过程则为

$$\boldsymbol{X}'=(\boldsymbol{X}_q-\boldsymbol{z})\times\boldsymbol{s} \tag{4}$$

2.2 分层混合精度量化

根据显著性离群点分组算法，本文首先确定了权重参数 end for 中的离群点。对于剩余的非显著性权重，采用 2 bit 量化。由于 2 bit 量化精度远低于 8 bit，可能会导致较大的模型精度损失。为此，本文引入分块量化策略，利用未量化权重对已量化权重的精度损失进行补偿。具体如下：

步骤 1. 初始化量化结果 $\boldsymbol{Q}$ 为一个零矩阵，初始化误差矩阵 $\boldsymbol{E}$ 为一个零矩阵；

步骤 2. 对 $\boldsymbol{H}^{-1}$ 进行 Cholesky 分解，得到 $\boldsymbol{H}^{-1}$ 的逆矩阵信息；

步骤 3. 迭代处理权重列的量化；

步骤 4. 计算量化误差，更新误差矩阵 $\boldsymbol{E}$；

步骤 5. 更新权重矩阵 $\boldsymbol{W}$ 中的权重，以减小误差。

以上的非显著性权重低比特量化的具体实现如下：

上述已经求出显著性的权重 $\boldsymbol{W}_1$，那么剩余的是非显著性权重，记为 $\boldsymbol{Y}$。首先对 $\boldsymbol{Y}$ 分成每 B 个一个 Block，如图 2 所示。

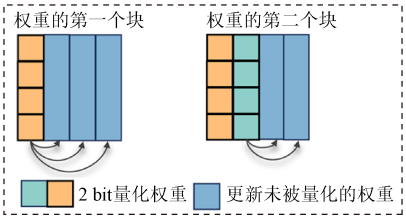

图 2　分块量化策略示意图

Fig. 2　Illustration of block-wise quantization strategy

然后对 $\boldsymbol{H}^{-1}$ 进行 Cholesky 分解，得到

$$\boldsymbol{H}^{-1} = Cholesky\left(\boldsymbol{H}^{-1}\right)^{\mathrm{T}} \tag{5}$$

对于每个 Block 中每列数据进行 2 bit 量化，即

$$\boldsymbol{Q}_{:,j} = quant\left(\boldsymbol{Y}_{:,j}\right) \tag{6}$$

式中：*quant* 表示 2 bit 量化函数。对每列进行量化后再进行量化误差的计算，即

$$\boldsymbol{E}_{:,j\text{-}i} = \frac{\left(\boldsymbol{Y}_{:,j} - \boldsymbol{Q}_{:,j}\right)}{\left[\boldsymbol{H}^{-1}\right]_{j,j}} \tag{7}$$

使用该 Block 中其他未量化的权重去补偿该误差，并更新权重

$$\boldsymbol{Y}_{:,j:(i+B)} = \boldsymbol{Y}_{:,j:(i+B)} - \boldsymbol{E}_{:,j\text{-}i}\boldsymbol{H}^{-1^2}_{j,j:(j+B)} \tag{8}$$

式中：$\boldsymbol{Y}_{:,j:(i+B)}$ 表示该 Block 中未开始量化的权重。最后将 Block 中所有列都进行 2 bit 量化，并使用所有剩余的未开始量化的 Block 中权重进行量化误差补偿，即

$$\boldsymbol{Y}_{:,:(i+B)} = \boldsymbol{Y}_{:,:(i+B)} - \boldsymbol{E}\boldsymbol{H}^{-1^2}_{i:(i+B),(i+B):} \tag{9}$$

这里以权重 $\boldsymbol{W}_1$ 为例对非显著性权重进行量化，其余剩余 $n-1$ 块权重类似。本算法整体流程如下：

算法 1. 基于内存对齐的大模型混合精度量化算法。

输入： 原始权重 $\boldsymbol{W}$，激活 $\boldsymbol{X}$，Block B。

输出： $\boldsymbol{W}_{\text{high}}$, $\boldsymbol{Y}$, Bitmap。

1. $\boldsymbol{H}^{-1} \leftarrow \left(2\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}} + \lambda I\right)^{-1}$.　// Hessian 矩阵逆
2. $(\boldsymbol{W}_1, \cdots, \boldsymbol{W}_n) \leftarrow \boldsymbol{W}$　　//分块
3. $\boldsymbol{S}_i \leftarrow \sum_j \frac{\boldsymbol{w}_{ji}^2}{\left[\boldsymbol{H}^{-1}\right]_{ii}^2}$　　//显著性计算
4. $\text{bitmap} \leftarrow \text{Top}K(\boldsymbol{S}_i)$　　//获取 bitmap
5. $\boldsymbol{W}_{\text{high}} \leftarrow \text{high\_quanti}(\text{mean}(\text{high}(\boldsymbol{S}_i)))$　//高比特量化
6. $\boldsymbol{H}^{-1} = Cholesky\left(\boldsymbol{H}^{-1}\right)^{\mathrm{T}}$　　//针对显著性低的权重列 $\boldsymbol{Y}$
7. for $i = 0, B, 2B \cdots$ do
8. 　for $j = i, i+1, \cdots i+B-1 \cdots$ do
9. 　　$\boldsymbol{Q}_{:,j} \leftarrow quant\left(\boldsymbol{Y}_{:,j}\right)$　　//量化列
10. 　　$\boldsymbol{E}_{:,j\text{-}i} \leftarrow \frac{\left(\boldsymbol{Y}_{:,j} - \boldsymbol{Q}_{:,j}\right)}{\left[\boldsymbol{H}^{-1}\right]_{j,j}}$　　//量化误差
11. 　　$\boldsymbol{Y}_{:,j:(i+B)} = \boldsymbol{Y}_{:,j:(i+B)} - \boldsymbol{E}_{:,j\text{-}i}\boldsymbol{H}^{-1^2}_{j,j:(j+B)}$　// 更新权重
12. 　end for
13. $\boldsymbol{Y}_{:,:(i+B)} = \boldsymbol{Y}_{:,:(i+B)} - \boldsymbol{E}\boldsymbol{H}^{-1^2}_{i:(i+B),(i+B):}$　　//更新剩余权重
14. end for
15. return $\boldsymbol{W}_{\text{high}}$, $\boldsymbol{Y}$, Bitmap

2.3　混合精度权重的打包与存储

在存储整个权重矩阵时，对于低位宽量化权重(2 bit)，将 16 个 2 位的值拼接成 4 个字节进行存储；而对于高位宽权重(8 bit)，则将 4 个 8 位的数值拼接成 4 个字节，其中每个 8 位数值实际上代表了 4 个 int8 数值的平均值。该设计的本质是为了能够将高位宽和低位宽的数据存放在同一个权重矩阵中。

对于一个大小为 $T\times P$ 的权重矩阵 $\boldsymbol{W}$，每次读取权重时，将一列中的 16 个权重作为一组，供 SIMD128 位寄存器进行计算，因此每列需要访问 $T/16$ 个组。打包后的权重矩阵形状为$[T/16, P]$，每个元素均为 4 字节的 INT32，即该矩阵共有 P 列，每列 $T/16$ 个元素，每个元素占 4 字节。

此外，还额外存储了一个 Bitmap，用于指示每个 32 位数据是由 2 位数据，还是由 8 位数据打包而成。该 Bitmap 为大小为$[T/16, P]$的二值矩阵，其

存储空间为[T/16/8, P]字节。

3 实验

3.1 实验设置

在数据集方面，为了验证方法的有效性，本文在多个语言任务上进行了实验验证，实验数据集包括了 WikiText2[25]，C4[26]和 7 个 zero-shot QA 数据集(包括 PIQA[27], BoolQ[28], OBQA[29], Winogrande[30], Hellaswag[31], ARC-e[32], ARC-c[32])。

在测试指标方面，本文参考相关工作采用了困惑度(Perplexity, PPL)指标。该指标对于量化较为敏感，可以较好地刻画量化带来的精度损失，且越低越好。此外还包括在 7 个 zero-shot QA 数据集的 Accuracy，该指标越高越好。

本文还对比了 RTN[33], GPTQ[11], PBLLM[2], BILLM[3], ARB-LLM[34], SliM-LLM[35]和 QuIP[21]等大模型量化方法，由于本文主要针对内存对齐的模型量化方法，则不对于一些基于稀疏的量化方法如 SpQR 等进行对比，因为稀疏表示导致内存访问速度受到影响，无法公平对比。

为了验证方法的普适性，本文在 Llama2 上 7 B, 13 B 和 70 B 等不同规模的大模型上进行了验证，以证明本方法在不同规模大模型上的普适性。

在校准数据集方面，本文的校准集与 GPTQ 等相同，选择了来自 C4 数据集的 128 个随机的长度为 2 048 的序列，主要是从网站中爬取的通用文本数据。在本文实验中选取 K = 50 和 100。分别对应 Ours (K = 50)和 Ours (K = 100)，K 指的是将权重中显著性高的前 K 列作为 8 bit 量化。

3.2 实验结果

本文首先在Wikitext2数据集上验证了量化方法的困惑度，表1给出了不同量化方法在LLama2 7 B, 13 B, 70 B等不同规模的大模型上进行低比特量化的模型PPL。从表1可以看出，Ours (K=100)方法在7 B，13 B和70 B等不同规模的大模型上进行2.14 bit量化后均实现了比当前量化方法更低的PPL指标。在700亿参数规模的大模型上，本方法的PPL上升1.47个点，但模型参数内存下降约87%(2.14 bit vs 16.00 bit)。另外一个现象是，随着模型规模的上升，困惑度指标的上升越少。由于PPL越高，模型表现越差，说明模型在参数更大的规模上，量化损失更低，而在7 B和13 B等规模的模型上，其量化损失相对更高一些。

表 1 不同量化方法在 WikiText2 数据集上的困惑度

Table 1 The perplexity of different quantization methods on WikiText2

方法	位宽	7 B	13 B	70 B
BF16	16.00	5.47	4.88	3.32
RTN	2.00	17 788.00	51 145.00	26 066.00
GPTQ	2.00	60.45	19.70	9.12
QuIP	2.00	39.73	13.48	6.64
PBLLM	1.70	69.20	151.09	28.37
BILLM	2.08	32.48	16.77	8.41
ARB-LLM	2.08	16.44	11.85	6.16
SliM-LLM	2.16	16.01	9.41	6.28
Ours (K=50)	2.14	25.44	13.82	7.56
Ours (K=100)	**2.14**	**7.88**	**6.57**	**4.79**

注：加粗数据表示最优值。

为了验证量化方法在不同测试集上的量化损失问题，本文在Llama2-7B上进行实验，在C4数据集上进行测试，其结果见表2。表2中对比了BILLM方法，其权重除了1 bit参数外，还需保存1 bit的索引，用来区别不同的权重，因此其位宽不是标准的1 bit，而是2.08 bit。从表2中可以看出，本方法在PPL方面低于其他量化方法。且相比最先进的SliM-LLM，Ours (K=100)取得了最优的效果为10.2，降低了5.8 PPL。

表 2 不同量化方法在 C4 数据集上的困惑度对比

Table 2 The perplexity of different quantization methods on C4

方法	分块	位宽	困惑(PPL↓)
BF16	-	16.00	6.97
GPTQ	128	2.00	43.24
QuIP	128	2.00	31.94
PBLLM	128	1.70	80.15
BILLM	128	2.08	40.52
ARB-LLM	128	2.08	20.12
SliM-LLM	128	2.16	16.00
Ours (K=50)	128	2.14	21.10
Ours (K=100)	**128**	**2.14**	**10.20**

注：↓表示数值越低越好；加粗数据表示最优值。

此外，为了进一步验证本方法的有效性，本文在7个zero-shot QA数据集上对比了现在主流的BILLM，ARM-LLM和SliM-LLM。见表3，Ours (K=100)相比最先进的方法，平均准确度取得了6.24%的提升。充分证明了本方法的有效性。

3.3 内存定性分析

为了进一步验证本方法的优势，还对内存访问效率与速度进行定性分析。本方法在“访问量”和“访问路径”上更具优势：与 BiLLM，ARB-LLM 和 PBLLM等需要同时维护Salient Weight，Non-Salient Weight 与 Bitmap 共 3 个矩阵，并在推理时先读 Bitmap 索引再跳转到对应权重的“两段式、非连续”访问不同，这种间接寻址/指针追踪会打散局部性、

表3　不同量化方法在7个zero-shot QA数据集上的准确性

Table 3　Accuracy of different quantization methods on the seven zero-shot QA datasets

方法	位宽	PIQA↑	BoolQ↑	OBQA↑	Winogrande↑	ARC-e↑	ARC-c↑	Hellaswag↑	Average↑
BILLM	2.08	60.39	59.42	**29.80**	51.93	39.98	23.72	35.90	43.02
ARB-LLM	2.08	66.59	66.33	29.60	57.85	51.01	27.56	48.33	49.61
SliM-LLM	2.16	53.64	52.59	15.00	47.98	25.08	21.50	26.29	34.58
Ours (K=50)	2.14	68.66	63.27	21.00	58.09	50.00	26.19	40.01	46.75
Ours (K=100)	2.14	**75.41**	**69.20**	26.40	**66.77**	**67.42**	**35.92**	**49.80**	**55.85**

注：↑表示数值越高越好；加粗数据表示最优值。

增加额外内存事务，因而更易出现缓存未命中并拉低带宽利用率、抬高访存延迟；本方法将 Salient 与 Non-Salient 权重通过内存对齐合并为单一权重矩阵，使读写顺序更线性、数据更连续，这完美契合了现代 CPU/GPU 内存子系统的设计：

1) 缓存友好。连续的内存访问模式可以最大限度地利用缓存行(Cache Line)。当 CPU 从内存中加载一个数据时，可将相邻的一大块数据一起加载到缓存中。访问下一个数据时，其已经在缓存里，速度极快。避免了缓存行的浪费。

2) 预取友好。内存控制器和硬件预取器(Prefetcher)可以非常准确地预测下一次需要访问的内存地址，并提前将其加载到缓存中，进一步隐藏内存访问延迟。此外访问任何一个权重只需要一个基地址(权重的起始地址)加上一个偏移量即可。计算简单，指令效率高。

3.4　消融研究

为了进一步验证本方法的有效性，在LLama2-7 B进行消融研究，本方法主要由 2 个关键部分组成分别是：基于二阶信息的离群参数分组算法(简称分组)、分层混合精度量化(简称混合精度)。将去除分组算法简称本文-G，将去除混合精度算法简称本文-M。从表 4 可以看出，去掉任何一个组成部分都会造成本方法在 Wikitext2 数据集上的 PPL 上升。具体而言，当去除分组算法时，本方法的 PPL 会上升 17.94。而当去除混合精度量化即都使用 2 bit 量化时，本方法的 PPL 则会上升 117.71。从而验证本方法各个部分的有效性。

表 4　本方法在 WikiText2 数据集上的消融研究

Table 4　Ablation study of our method on WikiText2

方法	分组	混合精度	PPL (↓)
本文-G		✓	43.38
本文-M	✓		143.15
本文	✓	✓	**25.44**

注：✓ 表示使用该方法；↓表示数值越低越好；加粗数据表示最优值。

3.5　可视化分析

为了进一步理解或解释本方法，将使用本文方法与 BILLM 方法量化后的参数进行了可视化对比，如图 3 所示。本文方法在参数表示上更加平滑(红色值更少)，更有利于减少模型量化误差，在保持内存对齐的同时，实现更好的大模型压缩。

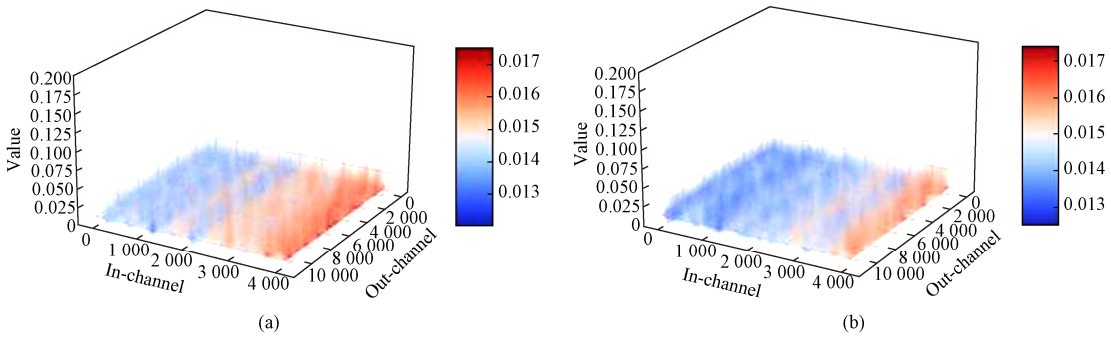

图 3　量化后参数的可视化对比结果

Fig. 3　Visual comparison results of quantized parameters ((a) BiLLM; (b) Ours)

4　结论

本文提出了一种内存对齐的大模型混合精度量化方法，通过小组显著性分析、分块量化补偿策略以及高效的权重打包与存储方案，显著降低了模型的内存占用和计算开销，同时保持了较高的推理精度。实验结果表明，该方法在多种大模型任务上均取得了良好的效果：相较于全精度模型，内存占用减少约 87%，70 B 模型的困惑度上升控制在 1.47。本研究虽然目前只关注文本领域，但为大模

型的高效推理提供了一种可行的量化策略，未来可进一步探索动态混合精度量化与硬件适配优化，以进一步提升性能。

参考文献 (References)

[1] GUO C, TANG J M, HU W M, et al. OliVe: accelerating large language models via hardware-friendly outlier-victim pair quantization[C]//The 50th Annual International Symposium on Computer Architecture. New York: ACM, 2023: 3.

[2] SHANG Y Z, YUAN Z H, WU Q, et al. PB-LLM: partially binarized large language models[EB/OL]. [2025-04-10]. https://arxiv.org/abs/2310.00034.

[3] HUANG W, LIU Y D, QIN H T, et al. BiLLM: pushing the limit of post-training quantization for LLMs[EB/OL]. [2025-04-10]. https://arxiv.org/abs/2402.04291.

[4] FRANTAR E, ALISTARH D. SparseGPT: massive language models can be accurately pruned in one-shot[EB/OL]. [2025-04-10]. https://arxiv.org/abs/2301.00774.

[5] SUN M J, LIU Z, BAIR A, et al. A simple and effective pruning approach for large language models[EB/OL]. [2025-04-10]. https://arxiv.org/abs/2306.11695.

[6] GU Y X, DONG L, WEI F R, et al. MiniLLM: knowledge distillation of large language models[EB/OL]. [2025-04-10]. https://arxiv.org/abs/2306.08543.

[7] QIU J T, WANG J, YAO S, et al. Going deeper with embedded FPGA platform for convolutional neural network[C]//2016 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. New York: ACM, 2016: 26-35.

[8] LIN D D, TALATHI S S, ANNAPUREDDY V S. Fixed point quantization of deep convolutional networks[EB/OL]. [2025-04-10]. https://arxiv.org/abs/1511.06393.

[9] COURBARIAUX M, BENGIO Y, DAVID J P. BinaryConnect: training deep neural networks with binary weights during propagations[EB/OL]. [2025-04-10]. https://arxiv.org/abs/1511.00363.

[10] LIU Z C, OGUZ B, ZHAO C S, et al. LLM-QAT: data-free quantization aware training for large language models[C]// Findings of the Association for Computational Linguistics. New York: ACL, 2024: 467-484.

[11] FRANTAR E, ASHKBOOS S, HOEFLER T, et al. GPTQ: accurate post-training quantization for generative pre-trained transformers[EB/OL]. [2025-04-10]. https://arxiv.org/abs/2210.17323.

[12] LEE J H, KIM J, YANG J Y, et al. LRQ: optimizing post-training quantization for large language models by learning low-rank weight-scaling matrices[C]//2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies. New York: ACL, 2025: 7708-7743.

[13] KIM J, EL HALABI M, PARK W, et al. GuidedQuant: large language model quantization via exploiting end loss guidance[EB/OL]. [2025-04-10]. https://arxiv.org/abs/2505.07004.

[14] DETTMERS T, LEWIS M, BELKADA Y, et al. LLM.int8(): 8-bit matrix multiplication for transformers at scale[EB/OL]. [2025-04-10]. https://arxiv.org/abs/2208.07339.

[15] YAO Z W, AMINABADI R Y, ZHANG M J, et al. ZeroQuant: efficient and affordable post-training quantization for large-scale transformers[EB/OL]. [2025-04-10]. https://arxiv.org/abs/2206.01861.

[16] XIAO G X, LIN J, SEZNEC M, et al. SmoothQuant: accurate and efficient post-training quantization for large language models[EB/OL]. [2025-04-10]. https://arxiv.org/abs/2211.10438.

[17] SHAO W Q, CHEN M Z, ZHANG Z Y, et al. OmniQuant: omnidirectionally calibrated quantization for large language models[EB/OL]. [2025-04-10]. https://arxiv.org/abs/2308.13137.

[18] YAO Z W, WU X X, LI C, et al. ZeroQuant-V2: exploring post-training quantization in LLMs from comprehensive study to low rank compensation[EB/OL]. [2025-04-10]. https://arxiv.org/abs/2303.08302.

[19] LIN J, TANG J M, TANG H T, et al. AWQ: activation-aware weight quantization for LLM compression and acceleration[EB/OL]. [2025-04-10]. https://arxiv.org/abs/2306.00978.

[20] DETTMERS T, SVIRSCHEVSKI R, EGIAZARIAN V, et al. SpQR: a sparse-quantized representation for near-lossless LLM weight compression[EB/OL]. [2025-04-10]. https://arxiv.org/abs/2306.03078.

[21] CHEE J, CAI Y H, KULESHOV V, et al. QuIP: 2-bit quantization of large language models with guarantees[EB/OL]. [2025-04-10]. https://arxiv.org/abs/2307.13304.

[22] LI L, LI Q Y, ZHANG B, et al. Norm tweaking: High-performance low-bit quantization of large language models[C]//The 38th AAAI Conference on Artificial Intelligence. Philadelphia: AAAI, 2024: 18536-18544.

[23] BEHDIN K, ACHARYA A, GUPTA A, et al. QuantEase: optimization-based quantization for language models[EB/OL]. [2025-04-10]. https://arxiv.org/abs/2309.01885.

[24] YUAN Z H, NIU L, LIU J W, et al. RPTQ: reorder-based post-training quantization for large language models[EB/OL]. [2025-04-10]. https://arxiv.org/abs/2304.01089.

[25] MERITY S, XIONG C M, BRADBURY J, et al. Pointer sentinel mixture models[EB/OL]. [2025-04-10]. https://arxiv.org/abs/1609.07843.

[26] RAFFEL C, SHAZEER N, ROBERTS A, et al. Exploring the limits of transfer learning with a unified text-to-text transformer[J]. The Journal of Machine Learning Research, 2020, 21(1): 140.

[27] BISK Y, ZELLERS R, LE BRAS R, et al. PIQA: reasoning about physical commonsense in natural language[C]//The 34th AAAI Conference on Artificial Intelligence. Philadelphia: AAAI, 2020: 7432-7439.

[28] CLARK C, LEE K, CHANG M W, et al. BoolQ: exploring the surprising difficulty of natural yes/no questions[C]//2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. New York: ACL, 2019: 2924-2936.

[29] MIHAYLOV T, CLARK P, KHOT T, et al. Can a suit of armor conduct electricity? A new dataset for open book question answering[C]//2018 Conference on Empirical Methods in Natural Language Processing. New York: ACL, 2018: 2381-2391.

[30] SAKAGUCHI K, LE BRAS R, BHAGAVATULA C, et al. WinoGrande: an adversarial winograd schema challenge at scale[J]. Communications of the ACM, 2021, 64(9): 99-106.

[31] ZELLERS R, HOLTZMAN A, BISK Y, et al. HellaSwag: can a machine really finish your sentence?[C]//The 57th Annual Meeting of the Association for Computational Linguistics. New York: ACL, 2019: 4791-4800.

[32] CLARK P, COWHEY I, ETZIONI O, et al. Think you have solved question answering? try ARC, the AI2 reasoning challenge[EB/OL]. [2025-04-10]. https://arxiv.org/abs/1803.05457.

[33] KRISHNAMOORTHI R. Quantizing deep convolutional networks for efficient inference: a whitepaper[EB/OL]. [2025-04-10]. https://arxiv.org/pdf/1806.08342.

[34] LI Z T, YAN X L, ZHANG T N, et al. ARB-LLM: alternating refined binarizations for large language models[EB/OL]. [2025-04-10]. https://arxiv.org/abs/2410.03129.

[35] HUANG W, QIN H T, LIU Y D, et al. SliM-LLM: salience-driven mixed-precision quantization for large language models[EB/OL]. [2025-04-10]. https://arxiv.org/ abs/2405.14917.