薛德军, 张庆国, 刘艳, 等. 大模型量化方法综述[J]. 智能计算机与应用,2026,16(5):61-72. DOI:10.20169/j.issn.2095-2163.24080701

大模型量化方法综述

薛德军, 张庆国, 刘 艳, 师庆辉, 赵一方
(同方知网数字科技有限公司, 北京 100192)

摘 要: 大模型(Large Language Models,简称 LLMs)结构复杂,参数众多,计算成本极高,特别是对内存的大量需求,使得大模型的实际部署面临着诸多挑战。因此,模型压缩与加速技术成为推动 LLMs 商业化应用的关键问题之一。在当前的各种 LLMs 压缩与加速方案中,量化技术是最主要的有效手段之一。量化是减少 LLMs 内存消耗和提升推理速度的最直接途径,尤其在支持低位数据类型(如 INT4)快速处理的硬件上效果显著。本文对大模型的量化技术进行了全面系统的总结和分析,介绍了量化类别,梳理了近年来常用的 LLMs 量化技术及其优缺点,重点介绍了量化感知训练和后训练量化两种量化方法,并指出 LLMs 量化未来可能的发展方向,为后续量化技术的发展提供依据与参考。

关键词: 大模型; 量化; 量化感知训练; 后训练量化
中图分类号: TP183 **文献标志码**: A **文章编号**: 2095-2163(2026)05-0061-12

A review of quantization methods for Large Language Models

XUE Dejun, ZHANG Qingguo, LIU Yan, SHI Qinghui, ZHAO Yifang
(Tongfang Knowledge Network Digital Technology Co., Ltd., Beijing 100192, China)

Abstract: Large language models (LLMs) have complex structures, many parameters and high computational cost, especially the large demand for memory, which makes their actual deployment face many challenges. Therefore, model compression and acceleration technology has become one of the key technologies to promote the commercial application of large language models. In various current LLMs compression and acceleration schemes, quantization technology is one of the most important and effective ways. Quantization is the most direct way to reduce LLMs memory consumption and improve inference speed, especially on hardware that supports fast processing of low-bit data types (such as INT4). This paper makes a comprehensive and systematic summary and analysis of the quantization technology of LLMs, introduces the quantization categories, combs the commonly used LLMs quantization technology in recent years and its advantages and disadvantages. The two quantization methods of quantization-aware training and post-training quantization are mainly introduced, and points out the possible future development direction of LLMs quantization, so as to support the future development of quantization technology.

Key words: LLMs; quantization; quantization-aware training; post-training quantization

0 引 言

近年来,大模型发展势头迅猛,在自然语言处理、知识问答、文本生成、机器翻译等领域取得显著成果,且在各种任务中表现出色[1-3],已成为人工智能领域的一个核心议题。相较于传统语言模型,ChatGPT、LLaMA、Claude 等代表性 LLMs 展现出了更强大的泛化性能及涌现能力[4]。此外,LLMs 本身所具备的语言理解能力已成功迁移至多模态模型中[5-9]。这些开创性的成果充分彰显了 LLMs 在未来应用领域的巨大发展潜力。

然而,LLMs 在推理过程中需要更多的显存和 GPU 计算资源,例如,GPT-3 模型需要加载 350 GB 显存,参数为 FP16 格式至少需要 5 张 A100-80 GB GPU 卡[10]才能支撑运行。随着模型规模的扩大和序列长度的增加,推理的内存成本也会相应地增长,

作者简介: 薛德军 (1970—),男,博士,高级工程师。主要研究方向:数据要素市场,数字出版,人工智能;张庆国(1976—),男,硕士,高级工程师。主要研究方向:人工智能技术应用,数据要素市场研究;师庆辉 (1978—),男,学士,高级工程师。主要研究方向:自然语言处理,深度学习,大模型;赵一方(1984—),男,博士,高级工程师。主要研究方向:数据要素市场,数字出版。
通信作者: 刘 艳 (1982—),女,博士,高级工程师。主要研究方向:人工智能技术应用。Email: ly10430@cnki.net。
收稿日期: 2024-08-07

这为 LLMs 在资源受限的设备(尤其是移动终端)上的部署[4]带来了巨大的挑战,阻碍了 LLMs 的实际应用部署。

为解决上述问题,研究人员和工程师们探索并实施了各种模型压缩与加速技术,如剪枝[11-13]、蒸馏[14]、量化[15-16]和低秩分解[17]等,对推动 LLMs 技术的普及和应用至关重要。与剪枝、蒸馏等方式相比,量化具有高压缩比、低成本、高灵活性的优势[18],成为当前首选的推理加速技术。本文详细梳理了 LLMs 量化的方法,对这些方法进行了深入的阐述和总结。通过对比分析不同的 LLMs 量化方法,指出各自的优缺点,并探讨了其在实际应用中可能遇到的问题和挑战,为 LLMs 量化技术的进一步发展提供依据和参考,推动 LLMs 量化技术的应用和发展。

1 模型量化技术概述

模型量化技术是人工智能领域的一个重要分支,是将多比特高精度浮点数(如 FP32、FP16 等)转化为少比特低精度定点数(如 INT8、INT4 等),以此来减少模型的存储空间和计算量,提高模型运行的速度。模型量化技术最早可以追溯到深度学习模型发展的初期。当前,8 位整数或 16 位半精度浮点量化技术在工业界已达到较为成熟的水平[19],并已经实现商业化应用。8 位整数量化方法的代表包括 Google 的 TensorFlowlite[20]、NVIDIA 的 TensorRT、Meta 的 PyTorch[21] 以及 Qualcomm 的 AI Model Efficiency Toolkit(AIMET)[22]等。开展模型量化技术商业化应用的公司包括 Qualcomm、NVIDIA 和 Google 等,均陆续发布了与产业相关的白皮书[23-24]和模型量化部署框架[20-22]。近年来 LLMs 的迅速发展,也推进了大模型量化技术的进展。

根据量化数据表示的原始数据范围是否均匀,可以将模型量化分为线性量化和非线性量化;根据激活值的量化实时性与否,可以将模型量化分为静态量化和动态量化;根据不同网络层量化位宽的不同,可以将量化分为统一精度量化和混合精度量化。

1.1 线性量化/非线性量化

1.1.1 线性量化

线性量化(又称均匀量化)是一种相邻量化值间距固定的量化方法。常见的线性量化过程可表示如下:

$$r = \mathrm{Round}(S(q - Z)) \tag{1}$$

其中,q 为原始的 float32 数值;Z 为 float32 数值的偏移量,又叫 Zero Point;S 为 float32 的缩放因子,又叫 Scale;Round(·)为四舍五入近似取整的数学函数,除了四舍五入,使用向上或者向下取整也可以;r 为量化后的整数值。

均匀量化的优势在于其简便易行和较低的计算负担。在均匀量化过程中,各量化区间的宽度保持一致,确保了信号在整个数值范围内的分辨率大致相等。然而,对于那些频率范围较大的信号,固定的区间宽度可能会引起较大的量化误差。

根据参数 Z 是否为零可以将线性量化分为对称量化和非对称量化。

对称量化在量化前和量化后的零点保持一致,即浮点零点与量化后整数零点直接对应。这种方法的计算量低,缺点是量化后的数据非饱和,有一部分区域不存在量化的数据。

非对称量化需要一个偏移量 Z 来完成零点的映射,即量化前的零点和量化后的零点不一致。因其额外引入了一个偏移量来修正零点,因此需要的计算量大。优点是其量化后的数据是饱和的,即量化前的最小值对应量化范围的最小值,量化前的最大值对应量化范围的最大值。

1.1.2 非线性量化

非线性量化又叫非均匀量化,量化值之间的间隔不固定。网络中值的分布往往是不均匀的,更多的是类似高斯分布的形式。非均匀量化依据信号的幅值范围及其重要性来调整各量化区间的宽度。在位宽固定的条件下,非均匀量化往往能够实现比均匀量化更高的精确度和更低的误差,这得益于神经网络权重的非均匀分布。然而,非均匀量化方法的计算效率可能较低,因为其通常涉及复杂的查找操作,这些操作与 GPU 等并行计算硬件并不直接兼容。

1.2 静态/动态量化

量化神经网络中权重与激活的核心差异在于,权重在推理过程中大多是固定的,因此可以静态计算量化所需的缩放与零点参数,或在校准中使用的统计数据。激活并非如此,因其范围和统计信息在运行之前是未知的。因此,根据激活值的量化实时性与否,可以分为静态量化和动态量化两类。

静态量化中离线计算好模型权重和激活的量化参数,推理的时候不再调整直接使用。对激活值量化需要获取激活值的分布信息,因此,静态量化中需要提供一定的数据来推理网络,收集网络的激活值信息,确定相关的量化参数。

在动态量化中,激活值相关的量化参数在推理

阶段实时计算。虽然效果更好，但是给推理带来额外的开销。

1.3　统一精度量化/混合精度量化

统一精度量化的核心是：所有量化网络层均采用相同的位宽（相同的整型类型）。这种方法简化了精度选择的过程，无需细致考量各个层面对量化误差的敏感度。然而，当网络整体量化或量化到较低精度时，容易出现模型准确率显著下滑的问题。

混合精度量化可以把不同的网络层量化至不同的位宽。其核心理念在于，对量化误差敏感、精度降低影响较大的层，保持较高的精度；而对那些可以较为激进地量化的层，则采用更低的精度。通过这种方式，可以在整体上降低网络的位宽，同时尽可能地减少对网络准确度的影响。混合精度量化还需要特别关注的一个关键问题是如何确定不同网络层的最佳量化位宽。

2　大模型量化方法

LLMs 在推理过程中需要更多的显存和计算资源，这使得资源受限的设备（如移动设备）的部署面临一定的限制。为了减少模型的复杂性和计算资源需求，并使模型在资源受限的环境中更高效地运行，可以对 LLMs 进行量化处理。

对 LLMs 进行量化，不可避免地会使得模型损失部分预训练阶段学习到的信息，这可能影响到模型在特定任务上的表现，如何在精度和效率之间找到平衡是一个挑战。因此，在选定特定的量化方法并确定了量化模型后，对量化函数中的量化参数或原始网络模型中的参数进行优化是很有必要的，这有助于在一定程度上恢复网络模型的性能。目前，根据是否对量化模型进行重新训练，量化方法主要分为量化感知训练和后训练量化两种，具体方法及优缺点见表 1。

2.1　量化感知训练

量化感知训练（Quantization Aware Training，QAT）是在训练或微调过程中对模型进行量化，使 LLMs 在训练过程中适配低精度计算，减少由量化引起的精度损失，保持量化之后 LLMs 的高性能表现。LLMs 上的 QAT 方法分为全参数再训练（Full-Parameter Re-Training）[18] 和量化感知微调（Quantization-Aware Fine-tuning，QAF）。

表 1　LLMs 量化方法分类及优缺点

Table 1　Classification of quantization methods for LLMs

量化类型	优势	不足	方法
量化感知训练	能减少由量化引起的精度损失，保持量化之后 LLM 的高性能，可以做到在对应任务的性能上与全精度模型相差无几	计算成本较高，消耗大量资源，高度依赖训练数据集	LLM-QAT、AlphaTuning、PEQA、QLORA、QA-LoRA、LoftQ、LACos-Bloom、INT2.1
仅权重量化	大幅减小模型大小，使模型可以在资源受限的环境中高效运行，提高模型的推理速度	引起一定的精度损失，影响模型性能	LLM.int8（）、BiLLM、PB-LLM、SqueezeLLM、nuQmm、GPTQ、Quip、OWQ、SignRound、SpQR、FlexRound、EasyQuant、AWQ、LoRC、FineQuant
权重和激活量化	更全面地优化模型的存储和计算需求，使模型能更好地适应不同的数据分布和任务需求	对模型性能的影响更大	SmoothQuant、Outlier Suppression+、Omniquant、RPTQ、QLLM、Outlier Suppression、FPTQ、MoFQ、ZeroQuant-FP、ZeroQuant、Rex、OliVe

2.1.1　全参数再训练

全参数再训练（Full-Parameter Re-Training）是指在完成模型压缩后，对模型所有参数进行完整微调，以恢复或接近原始模型的性能。其中，LLM-QAT[25] 首次在 LLMs 上应用 QAT 的方法，实现了第一个准确的 4 bit 量化 LLMs。LLM-QAT 是利用预训练模型生成的数据来实现无数据的蒸馏方法，除了量化权重和激活外，还量化了 KV 缓存，这对于提高吞吐量和支持当前模型大小的长序列依赖性至关重要。还有些研究将 QAT 与蒸馏相结合，以保留原始模型的涌现能力[26-27]。

2.1.2　量化感知微调

量化感知微调（Quantization-Aware Fine-tuning，QAF）是在微调过程中对 LLMs 进行量化，其核心宗旨在于确保模型在经过微调后，即便其参数量化至较低位宽，也能维持原有的性能水平（表 2）。

AlphaTuning[28] 与 PEQA[29] 通过冻结量化权重并对模型进行微调的方式，来压缩和加速 LLMs。AlphaTuning[28] 将模型参数分解为二进制权重和缩放因子，仅在下游任务中微调少量的缩放因子，同时冻结其他二进制权重。这种策略不仅能在多种下游

任务中实现与全量微调相当的性能,还能大幅减少参数量和内存占用。PEQA[29]则仅更新量化比例因子,同时冻结整数权重矩阵,可显著降低优化器状态所需的内存量,并实现快速的任务切换。其中,AlphaTuning 采用二进制编码量化(BCQ)[30]。

表 2 量化感知微调方法的效果

Table 2 Performance of Quantization-Aware Fine-tuning methods

方法	效果
AlphaTuning	在 4 bit 量化下实现了> 10 ×的压缩比,并减少了> 1 000 ×的可训练参数。
PEQA	即使 LLMs 被量化到 4 bit 精度以下,PEQA 能在减小模型大小的同时,通过进一步指令微调,恢复甚至提升量化模型的性能。
LORA	采用低秩自适应方法,参数量较全参数微调显著降低,参数量和现有高效参数微调方法持平或更低;性能优于其它参数高效微调方法,和全参数微调基本持平。
QLORA	在单个 48 GB GPU 上微调 65 B 参数模型,保持完整的 16 bit 调优任务性能。
QA-LoRA	在微调阶段显著减少了时间和内存消耗,与 QLoRA 相比,QA-LoRA 的在推理阶段更快、更准确。
LoftQ	为 LoRA 微调提供更佳的初始化,提升了量化模型在下游任务的表现,尤其在低比特量化场景下展现出卓越的性能,同时保持了高效性和兼容性。
INT2.1	可将 7 B 参数的 LLMs 微调所需的内存降低 5.6 倍,并成功生成了可以生成连贯英语文本的 INT2 量化 LLM。

此外,还有一些研究采用低秩自适应(LoRA)方法[31],在相对合理的计算资源下,对量化的 LLMs 进行高效再训练[26-31]。QLoRA[32]采用新的数据类型 4 bit NormalFloat(NF4)将 LLMs 权重降低至 4 bit,然后在量化后的模型中插入少量可学习的低秩适配器对模型进行微调,再用双量化技术对量化参数再次进行量化,并使用分页优化器(Paged Optimizers)管理内存峰值,避免在长序列情况下出现内存不足错误,实现了在单个 GPU 上微调 LLMs,同时在 Vicuna 基准上实现最先进的结果[31]。Li 等[33]发现 QLoRA 中 LoRA 矩阵的零初始化会降低下游任务效率,因此,提出一个新颖的量化框架 LoftQ。LoftQ 通过交替应用量化技术和低秩近似,以便更接近原始高精度权重,显著改进了下游任务的性能,尤其在低比特量化下展现出优越性。

QA-LoRA[34]在低秩适应框架下,通过分组量化来增加量化方参数的数量,同时减少适应参数的数量,实现了量化感知的低秩适应,在适应效率和部署效率上均优于 QLoRA。LACos-Bloom[35]用 8 bit 块状均匀量化模型、再用低秩自适应(LoRA)和 8 bit Adam 优化器对量化后的模型进行微调,实现了参数高效微调,可以端到端地在单个 GPU 上微调数十亿参数的 LLMs。INT2.1[36]通过在量化模型的每个层注入低秩分解矩阵,并在端到端训练中学习最小化量化模型和全精度模型的输出分布之间的距离,可将 7 B 参数的 LLMs 的微调内存占用降低 5.6 倍,并成功生成了可以生成连贯英语文本的 INT2 量化 LLMs。

这些研究表明,通过量化感知微调技术,可以在保持模型性能的同时,显著降低训练和部署 LLMs 的成本。

2.2 后训练量化

后训练量化(Post Training Quantization, PTQ)是在模型训练完成后进行量化,无需修改模型结构或重新训练,仅使用少量校验数据就可以得到一个性能较好的量化模型,可以显著降低模型的存储和计算复杂度。但这种方法在极低比特(小于 4 bit)条件下的表现并不理想。PTQ 的主要优势是操作的简便高效。但由于神经网络对量化误差敏感,往往会导致精度的显著下降。LLMs 的 PTQ 工作大致分为两类:仅权重量化和权重+激活量化。

2.2.1 仅权重量化

仅权重量化(Weight-Only Quantization)是仅对模型的权重进行量化操作,以整型形式存储模型权重,可以压缩模型的大小。通过将高精度的浮点数权重转换为低位数的定点数或二进制数,从而大幅减小模型的大小。这种优化不仅使得模型可以在资源受限的环境中高效运行,还能提高模型的推理速度,使得人工智能应用更加实用和便捷(见表 3)。

有些研究使用均匀的最近临舍入(Rounding-to-Nearest,RTN)方法进行量化[18]。RTN 是权重量化中最常见的方法之一[37],将连续实数值映射到最近的整数。

表 3　部分仅权重方法的效果

Table 3　Performance of partial Weight-Only Quantization Methods

方法	量化位宽/bit	内存占用情况	性能提升
LLM. int8 ( )	8	8 bit 量化将模型参数从 16/32 位转换为 8 bit，将模型的内存占用减少了一半，可在配备消费级 GPU 的单台服务器上运行 OPT-175B/BLOOM 模型。	可以在 175 B 参数的 LLM 中执行推理，而不会显著降低性能
SqueezeLLM	3	在 LLaMA-7B 模型的 3 bit 量化中，显著降低了与 FP16 基线的困惑度差距，最大可达 2.1 倍	在 3 bit 量化下实现了接近无损的压缩，显著降低了量化误差。此外，当部署在 A6000 GPU 上时，与基准模型相比获得了高达 2.3 倍的加速比。
nuQmm	2	有效降低了内存占用	通过 2 bit 量化，实现了在 8 个 GPU 上生成每个 token 的延迟降低了 47.3%，在 2 个 GPU 上降低了 23.2%，，并显著减少了 GPU 的使用数量和能耗。
GPTQ	3	量化到 3 bit，OPT-175 B 模型从原来的 326 GB（FP16 格式）减少到约 63 GB ，包括嵌入层和输出层，并且保持全 FP16 精度。	能够在大约 4 个 GPU 小时内压缩具有 1 750 亿个参数的大型网络，在 NVIDIA A100 等高端 GPU 上提供 3.25 倍的加速，在 NVIDIA A6000 等具有成本效益的 GPU 上提供 4.5 倍的速度提升。
Quip	2	能显著降低模型的内存占用。	不相干预处理可以显著改进现有的量化算法，并在仅使用 2 bit 权重的情况下首次实现了对 LLMs 的可行量化
OWQ	3.1	OWQ 与 WCT 结合量化后的 LoRA 模型在推理过程中仅使用 24.4% 的整体内存。	使用 OWQ 的 3.1 bit 模型在性能上可与使用 4 bit OPTQ 优化的模型相媲美，在灵活性、效率和实用性方面取得了显著的进步
SignRound	3	可有效地降低模型的内存占用	能够在 400 步内，仅仅改变大约 5 %的舍入值就可以显著提高一些量化模型的性能，且在推理过程中没有引入任何额外的开销。
SpQR	3	SpQR 将 LLM 的内存占用减少了约 3.4 倍以上	在 LLaMA 和 Falcon 模型上，SpQR 方法将模型压缩到 3-4 bit/参数，并保持了几乎无损的压缩效果，相对 16 bit 基线的困惑度损失小于 1%，速度提高了 20%~30 %。
FlexRound	3	可显著降低模型内存占用	第一个对图像分类和自然语言理解及自然语言生成进行全面实验的工作，可以有效量化大规模语言模型，仅对性能造成轻微影响。
EasyQuant	4	可显著降低模型内存占用	显著改善量化后模型在多个基准测试上的性能，且优于需要训练数据的方法，同时运行速度提高 10 倍以上。
AWQ	3	实现了在具有 64 GB 内存的移动 GPU 上部署 70 B Llama-2 模型，表明 AWQ 量化可以显著降低 LLM 的内存需求，使其能够在资源受限的设备上进行部署。	在桌面和移动 GPU 上实现了 3 倍以上的速度提升。
FineQuant	4	可显著降低模型内存占用	能够将 1 750 亿参数的模型部署在 2 个 GPU 上，将开销降低了 64%，并在相同数量的 GPU 上实现了 3.65 倍的吞吐量提升。模型并行推理能力提升，允许在单个具有 8 个 GPU 的 A100 节点上服务 4 个 OPT-175B 实例。

注：大部分文章将模型量化到不同精度进行比较，表格中列出的是文中所采用的最低量化精度值。未明确标注内存占用情况的，是因为文章中并未具体阐述模型内存占用的减少量，下同。

1) GPTQ(又称 OPTQ)[38]是典型的仅权重量化技术，基于自适应舍入原理，通过将 GPT 模型与后训练量化(PTQ)结合，为量化提供了创新的解决方案。其采用非对称逐行量化的策略，每次量化一

层,允许权重以任意顺序进行量化。在量化每个权重时,GPTQ 使用量化误差来更新未量化权重的值,以补偿量化带来的误差。另外 GPTQ 还通过批量量化更新优化计算流程,仅更新当前批次中相关的权重,避免全局更新带来的低效计算,可以在 4 h 内将 175 B 参数的模型量化到 3~4 bit 权重,实现单个 GPU 上运行 175 B 参数的生成模型。

2) 许多研究选择 GPTQ 作为基线方法:

(1) Quip[39] 可视为 GPTQ 等自适应舍入方法的增强版,包括自适应舍入和不相关处理步骤,通过不相关处理实现了更佳的量化效果。自适应舍入通过最小化二次误差代理目标对权重进行量化,以获得更准确的量化结果。不相关预处理步骤通过随机正交矩阵的预处理和后处理,确保权重矩阵和海森矩阵的不相关,从而降低量化误差,首次证明了 2 bit 量化 LLM 的可行性。

(2) SignRound[40] 使用带符号的梯度下降对权重四舍五入进行轻量级的块级调优,仅改变约 5% 的舍入值,就可以显著提高某些量化模型的性能,且不引入额外的推理开销。相比简单使用最近舍入方法的 GPTQ,该方法考虑了权重之间的关系,采用轻量级的块级调优,仅需少量未标注数据和 400 步优化,比 GPTQ 更加高效。

(3) FlexRound[37] 利用元素逐项除法导出的倒数导数规则,灵活量化预训练权重,尤其是对于绝对值较大的权重。相较仅进行四舍五入或上舍/下舍,FlexRound 可以根据权重大小灵活进行量化,从而提高量化性能。

(4) EasyQuant[41] 通过保留权重中的异常值和优化量化范围,有效降低了最临近舍入中的量化误差,提高了量化后模型的性能。

3) 其它使用均匀量化方法的还包括 ZeroQuant-V2[42] 与 FineQuant[43]:

(1) ZeroQuant-V2[42] 采用逐行对称量化将权重量化为 8 bit,模型性能基本保持不变,但在 4-bit 权重量化时,模型性能显著降低。为了解决这一问题,ZeroQuant-V2[42] 引入了低秩补偿(LoRC)的方法,利用低秩矩阵分解来优化量化误差矩阵,以改善量化模型的性能,是一种有效的量化误差校正方法。

(2) FineQuant[43] 采用均匀量化方法,引入自适应选择量化粒度的细粒度量化算法,可以根据不同的模型权重矩阵自动确定合适的量化粒度,在量化范围和精度之间取得平衡,有效地解决了量化带来的精度损失问题。

4) 一些模型也使用均匀量化,例如,GLM-130 B 模型[44] 采用 Absmax 对称量化,以权重值的最大绝对值作为缩放因子,在不进行微调的情况下直接从 FP16 量化到 INT4,使其在 4 个 RTX 3090 Ti 或 8 个 RTX 2080 Ti 上进行推理,且保持模型性能几乎不下降,降低了百亿级参数级别模型的使用门槛。

还有一些研究采用非均匀量化方法,训练后 LLMs 的权重分布是不均匀的,因此量化权重使用的间隔也是不均匀的[18]。SqueezeLLM[45] 是基于敏感度的非均匀量化,该方法通过分析权重值对最终损失的影响,采用 k-means 聚类确定量化箱的分配,将量化箱分配在敏感值附近,实现了在极低精度(如 3 bit)下对 LLM 进行无损压缩,并在不牺牲模型性能的情况下显著减少了模型大小和推理时间。nuQmm[46] 扩展了非均匀量化方法,即二进制编码量化(BCQ)[30],允许根据目标压缩比探索不同的量化位宽和分组大小,避免对激活进行量化。使用 BCQ 方法对权重矩阵进行非均匀量化,每个权重矩阵的每行共享一个缩放因子,并通过迭代求解找到最佳的缩放因子和二进制向量。Lin 等[47] 发现权重对模型性能的重要性并不相同,只有 1% 左右的显著权重对模型性能影响很大,因此提出激活感知权重量化方法 AWQ。该方法基于激活分布,采用每个通道的缩放因子,在量化过程中自动搜索最优的缩放因子,以保护重要权重,并最小化量化误差,首次实现了多模态语言模型的低比特量化。

此外,Li 等[48] 发现校正量化的激活分布以匹配其浮点对应形式可轻松恢复 LLMs 的准确性。为了实现这一目标,提出一种称为范数调整技术(Norm Tweaking[48]),可作为当前 PTQ 方法中的一个插件。其调整策略包括校准数据生成和通道间距离约束,用于更新归一化层的权重,使模型获得更好的泛化能力。

上述方法在量化时,所有的网络层均采用相同位宽的整型数据(即相同的量化级别),然而,对网络进行较低精度的量化时,这种方法可能会引起网络准确率的显著下降。为解决该问题,部分研究采取混合精度量化策略,即允许不同的网络层可以量化到不同的位宽。SpQR[49] 通过识别和隔离异常值,将其存储在更高精度空间,同时将其他权重压缩至 3~4 bit,实现压缩比和准确率之间的平衡。LLM. int8 ()[50] 采用混合精度分解,将少数具有大值特征维度分离至 16 bit 矩阵乘法中,而其他 99.9% 的值仍使用 8 bit 乘法,这种分解只增加了约

0.1%的额外内存。LLM.int8 ( )还采用向量量化，通过为矩阵乘法中的每个内积分配单独的量化归一化常数，提升量化精度。OWQ[51]通过分析激活函数的异常值，识别对量化敏感的权重列，并使用高精度(fp16)表示这些“弱列”，以减少量化误差，而其他权重则使用低精度量化，以节省存储空间，从而在维持低比特率的同时提高了量化大模型的性能。

2.2.2　权重+激活量化

权重和激活同时进行量化(Weight + Activation Quantization)，可以进一步压缩模型大小，提高存储和计算效率(见表 4)，使模型可能更好地适应不同的数据分布和任务需求。

表 4　部分权重+激活量化方法的效果

Table 4　Effect of partial Weight + Activation Quantization

方法	* bit 权重+激活	内存占用情况	性能提升
SmoothQuant	W8A8	在不损失精度的情况下，SmoothQuant 可以实现 2 倍的内存减少，能够在单个节点内服务 530 B LLM	将 SmoothQuant 集成到 PyTorch 与 FasterTransformer 中，获得了高达 1.56 倍的推理加速
Outlier Suppression+	W8A8，W6A6，W4A4	能够降低模型的内存占用。	在 8 bit 和 6 bit 量化下，实现了接近浮点数精度的性能，并在 4 bit BERT 量化中取得了 15.5%的性能提升，达到新的最佳水平。
Omniquant	W4A4，W6A6，W4A16，W3A16，W2A16	与 16 bit 全精度模型相比，量化模型显著减少了内存使用。	在各种量化场景中，特别是在低比特设置中，都取得了最先进的性能，同时保持了 PTQ 的时间和数据效率。可在单卡 A100-40GPU 上，在 1 ~ 16 小时内完成对 LLaMA-7B ~ LLaMA70B 模型量化参数的更新。
RPTQ	W3A3	将 OPT-175B 模型的关键/值缓存量化为 4 bit 位，可以将内存占用降低 73%(困惑度损失小于 0.5)或 80%(困惑度损失小于 1.5)	首次将激活量化到 3 bit，并保持较高准确率。
QLLM	W4A4	显著降低优化成本和 GPU 内存使用量	在单个 GPU 上仅需 10 h 即可完成 LLaMA-2-70B 的 4 bit 量化，并在 5 个零样本任务上平均精度提高了 7.89%。
FPTQ	W4A8	FPTQ 优化了内存与计算效能之间的平衡，实现了在 INT8 推理过程中采用 4 bit 格式来储存权重数据，进而加速了内存读取速度并提升了运算速率。	实现了对大型语言模型的高效 W4A8 量化压缩，同时在推理速度和模型精度方面取得了接近或略好于原始 FP16 模型的性能。
MoFQ	W8A8，W4	能够降低模型的内存占用。	MoFQ 在 4 bit 权重量化中优于 GPTQ，量化速度快一个数量级；并在 8 bit 权重激活量化中接近全精度模型的性能。
ZeroQuant-FP	W4A8	能够降低模型的内存占用。	FP8 激活量化在大模型中表现更佳，在参数超过 1 B 的模型中性能优势更加明显。同时，FP4 权重量化在小模型中明显优于 INT4。低秩补偿(LoRC)等策略进一步减轻了由于尺度因子限制对权重量化造成的退化。
ZeroQuant	W4A8	对权重的 INT4 量化，与 FP16 模型相比减少了 3 倍的内存占用，而准确率只降低轻微。	ZeroQuant 的 INT8 模型在 BERT 和 GPT-3 类型的模型上实现了高达 5.19 倍的加速比，同时保持了足够的精度。
OliVe	W4A4	通过牺牲正常值来处理异常值，并利用内存访问的连续性，有效解决了大型语言模型量化中的内存占用问题。	基于 OliVe 的加速器比现有的离群点感知加速器 GOBO 分别提高了 4.5 倍的加速比和 4.0 倍的能耗降低比，并且具有更高的模型精度。
Outlier Suppression	W4A4	能够降低模型的内存占用。	提升了 Transformer 模型在低比特量化下的性能，达到了与全精度相近的水平。

低位浮点格式(例如 FP8、FP4)提供了一种可以替代传统低比特量化(例如 INT8、INT4)的方案，且 FP8 已获得 NVIDIA 等领先硬件供应商的支持[52]。MoFQ[52]和 ZeroQuant-FP[53]的研究都表明，

在激活量化方面,FP8 量化明显优于 INT8。MoFQ 采用低比特整数(INT)格式和浮点(FP)格式的混合格式量化,逐层选择最优量化格式。此外,MoFQ[52]还将标准 FP 格式中的特殊 NaN(非数字)与 Inf(无穷大)值重新映射为归一化数值,以增强并让 FP 格式表示更多值,这一点对 FP4 等低位宽格式尤为重要。ZeroQuant-FP[53]使用 FP8 和 FP4 格式进行权重和激活量化,并引入了 LoRC 方法以减少量化误差。

Rex[54]方法采用残差误差扩展和基于组稀疏扩展的方式,使得原始值和去量化值之间的量化误差较小,从而以效率换取更高的模型性能。OliVe[55]采用异常值-受害者对(Outlier-Victim Pair, OVP)机制,修剪部分量化的低精度正态值,为高精度异常值腾出额外空间。该方法可以有效地利用现有的硬件加速器(如脉冲阵列和张量核)进行内存对齐的编码和解码,与现有的异常值感知加速器相比,OliVe 可以在 GPU 上实现 4.5 倍的性能提升和 4 倍的能效降低。ZeroQuant[56]集成了细粒度硬件友好的量化方案、逐层的知识蒸馏和优化的量化支持,显著提高了模型在低精度(如 INT8)下的推理性能,同时保留接近全精度(如 FP16)的准确率。QuantEase[57]是一个逐层量化框架,对各个层进行单独量化,将每层量化问题建模为一个离散结构的非凸优化问题,通过应用坐标下降(Coordinate Descent,CD)技术获得有效的解决方案。

激活通常包含较多异常值,这些异常值对 LLMs 性能至关重要,因此需要特别处理。一些研究通过管理激活异常值实现 LLMs 量化,包括通道缩放[10,58-59]、通道聚类[60]等。

(1)通道缩放通过为每个通道分配不同的缩放因子来减小量化误差。SmoothQuant[58]通过在线下对每个通道的激活值进行缩放,来平滑激活值的分布,从而减少激活值的量化误差,实现对大规模语言模型(LLMs)进行精确和高效的 8 位权值和激活量化。

(2)Outlier Suppression+[59]除使用通道缩放,还增加了信道方向移位来驱动 W6A6 量化,以纠正异常的不对称呈现,减轻问题通道的影响,同时定量分析了平移和缩放的最佳值,兼顾了异常的不对称性以及下一层权重引起的量化误差。Omniquant[10]包含两个创新组件:可学习权重剪裁(LWC)和可学习等价变换(LET)。LWC 通过优化剪裁阈值来调制权重中的极端值。同时,LET 通过在 Transformer 编码器中学习数学上等价的变换来处理激活中的异常值,从而将量化的难点从激活值转移到权重上。

(3)通道聚类是将激活中的通道按数值范围相似性分组,然后对每个组使用不同的量化参数进行量化。RPTQ[60]通过将具有相似值范围的通道聚类,对每个聚类内的激活值进行量化,从而减小了量化误差。此外,RPTQ 还将重排序操作与层归一化操作和线性层的权重相结合,以最小化计算开销。RPTQ 显著降低了 LLMs 的内存占用,同时在保持较高准确率的情况下实现了 LLMs 的 3 bit 量化。

(4)还有一些研究聚焦于抑制出现在激活中的异常值[61-63]。Outlier Suppression[61]通过抑制而非回避异常值来提高量化性能,为 Transformer 量化研究提供了新的视角。作者提出两种解决 Transformer 模型量化中异常值问题的方法:Gamma Migration 和 Token-Wise Clipping。前者通过等效转换,将 γ 的影响迁移到后续模块,获得一个对量化更友好的模型;后者利用词元范围大的方差,从词元角度设计一个从粗粒度到细粒度的管道,快速找到最小的最终量化损失的剪裁范围。FPTQ[62]提出一种新颖的对数激活均衡技术,用于处理难以量化的网络层,通过非线性映射压缩激活值分布范围;同时采用细粒度权重量化技术,在所有层中以组为单位计算量化级数,以获得更优性能,实现了高精度的 W4A8 量化且无需微调。QLLM[63]提出一种抑制 LLMs 中激活异常值的独特方法——自适应通道重组。其核心思想是通过通道分解和聚合,重新分配异常通道中的大激活值。通道分解可降低异常值大小,使激活更适配量化;而通道聚合则可以保持原始通道数,提高效率。自适应重组可以自适应地确定每个层的最优重组比例。

本文对比了仅权重与权重+激活量化方法:仅权重量化可将量化极限推到更低的位宽(甚至到 1 bit),显著降低 LLMs 所需设备的内存,但会引入精度损失,降低模型性能。而权重+激活量化能更全面地优化模型的存储和计算需求,使模型能更好地适应不同的数据分布和任务需求,但需要更多的位宽(8 bit)来存储权重和激活。因此,单一权重和权重+激活量化方法各有利弊,仍需进一步优化与改进,以持续满足实际应用需求。

3 大模型量化的其他方法

3.1 极低比特量化

上面提到的量化方法大多数成功地将模型量化

到 8 bit 或 4 bit，4 bit 量化参数提供了最佳的零样本准确性和模型大小之间的权衡，适用于所有模型家族和规模[64]。然而，当量化位宽 ≤ 3 bit[25] 时，LLMs 的质量显著下降。为了将 LLMs 量化推进到更低的比特范围，需要采用更激进的压缩方法。网络二值化作为一种强大的压缩技术脱颖而出，可以极大地将模型权重降低到 1 bit[47-50]，从而降低昂贵的计算和内存需求。

Onebit[65] 是一个 1 bit 量化感知训练框架，提出了新的 1 bit 线性层架构，将权重矩阵分解为符号矩阵和两个值向量，以保持高秩并弥补精度损失。同时引入了符号-值独立分解（SVID）方法，通过矩阵分解将高比特权重矩阵分解为低比特矩阵，为 1 bit 模型提供了有效的初始化。BitNet b1.58[66] 是一种新的 1.58 bit 大模型，该模型中每个参数取值为 {-1,0,1}，直接在训练过程中将权重和激活分别量化为 1.58 bit 和 8 bit。PB-LLM[67] 选择性地对语言模型中的权重矩阵进行部分二值化，将大部分权重二值化为 1 bit，只保留少部分权重以更高的比特数存储，但只保持了 800 左右的困惑度，平均权重为 1.7 bit。为了应对量化技术无法在超低位宽下保持 LLMs 的性能这一挑战，Huang 等[68] 提出了 BiLLM，一个开创性的 1 bit 后训练量化方案，专为预训练 LLMs 量身定制。基于 LLMs 的权重分布，BiLLM 首先对显著权重进行识别和结构选择，并通过有效的二进制残差近似策略最小化压缩损失。此外，考虑到非显著权重的钟形分布，提出了一种最优分裂搜索，以准确地对其进行分组和二值化。BiLLM 首次在 1.08 bit 权重下实现了高精度推理（例如 LLaMA2-70 B 的 8.41 困惑度），在多个 LLM 家族和评估指标上显著优于 SOTA 量化方法。此外，BiLLM 能够在单个 GPU 上在 0.5 h 内实现 70 B 权重的 LLMs 的二值化过程，显示出令人满意的时间效率。

3.2　高效内核和量化算法

高效内核是指专门为量化 LLMs 推理生成的低级矩阵运算内核，是对量化 LLMs 的底层支持。

上文提到的 LLM.int8()[50]、GPTQ[38] 和 SmoothQuant[59] 等方法，针对其特定的目标硬件（如 GPU）手动开发定制内核。手动方法可能非常耗时且容易出错，因此，Shen[69] 和 Pegolotti[70] 开发了基于 CPU 的自动高效内核。Shen[69] 基于 Intel Deep Learning Boost 开发了高效的稀疏矩阵-密集矩阵乘法（Sparse Matrix - Dense Matrix Multiplication，SpMM）内核，其性能比现有稀疏库（如 oneMKL、TVM、LIBXSMM）提高了 1 个数量级，并比密集 GEMM 核快 5 倍。QIGen[70] 是一种自动代码生成方法，用于生成量化 LLMs 生成推理的高效内核。该方法考虑了目标架构和性能模型，包括硬件特性和量化方法的准确性约束，显著提升了吞吐量，且精度只有轻微损失。

3.3　资源受限的边缘设备量化方法

一些研究探讨在资源受限的边缘设备上部署压缩和量化的 LLMs[71-73]，实现硬件友好的量化方法。例如，Wang 等[71] 提出硬件感知自动量化（HAQ）框架，利用强化学习自动搜索最优的量化策略，相较于基于规则的基线方法，该框架可以降低 1.4-1.95 倍的延迟，降低 1.9 倍的能耗，并在模型大小相似前提下获得更高的精度。Rahman[72] 针对 MobileBERT 模型，将其从 TensorFlow 格式转换为 TensorFlow-Lite 格式，并采用动态范围量化技术，根据数据的范围调整权重和激活的量化范围，并将浮点数转换为 8-bit 整数，优化其在资源受限的边缘设备上的部署。Lin 等[73] 将 LLMs 部署到 6 G 移动边缘计算（MEC）系统，采用后训练量化、量化低秩适配器、混合专家量化等量化技术，以适应资源受限的边缘环境，降低通信和计算需求。

4　结束语

LLM 量化的趋势主要体现在以下几个方面：

（1）极低比特量化。BiLLM[68] 首次在 1.08 bit 的权重精度下成功实现了高精度推理，这一突破使得 1 bit 量化变得可行。BitNet b1.58[66] 模型通过 1.58 bit 的权重和计算方式，实现了低能耗和高效计算，为大模型的部署和应用提供了重要的优化方向。而 OneBit[65] 把 LLM 压缩到 1.007 3 bit 时，仍然能使其保持约 83% 的性能。这些成果为未来 LLMs 实现更小巧的模型尺寸和更低的能耗提供了可能，同时还能保持良好的性能。这些进展使得极低比特量化成为了研究的热点之一。

（2）自适应量化技术。现有的大模型在后训练量化过程中普遍采用手工设定的量化参数，但由于缺乏有效的优化，这些方法在面临低比特量化时往往会导致性能显著下降。相比之下，自适应量化技术可根据模型的各个部分自动调整量化级别，这样既能保证性能，又可优化资源消耗。该技术可能需要对模型的结构和权重进行深入分析，以及对量化误差机制的深刻理解。自适应量化技术有望显著提升模型部署的效率，实现计算性能和能效的双重提

升,预计将成为未来 LLMs 量化研究的关键方向。

(3)量化与其他压缩技术的融合。Kim 等[26]和 Neill 等[27]将量化技术(QAT)与模型蒸馏相结合,成功保留了原模型的涌现性特征,证明量化与其它压缩技术的结合使用能够取得显著成效。进一步地,量化还能与剪枝、稀疏性等方法相结合,以减少模型的参数和计算负担,从而提高模型的运行效率和扩展性。剪枝可以先减少模型复杂性,然后通过量化来降低权重表示的精度,从而实现双重压缩效果。在模型蒸馏的过程中,量化的使用能够减少教师模型和学生模型中的参数数量,进而提升压缩后模型的性能。此外,量化与稀疏性的结合也能有效减少非零元素的数量和精度,从而降低存储和计算成本。在 LLM 量化方面,这些结合使用的方法是一种值得关注的重要研究方向。

本文全面深入地梳理了大模型的量化技术,详细地梳理了近年来在 LLMs 量化方面常用的技术,并对其优缺点进行了全面而深入的剖析。在此基础上,本文进一步探讨了大型模型量化技术未来的发展方向,可为后续研究提供有益参考。尽管大模型量化技术已取得一定的进展,但多数算法仍依赖于为压缩小型模型而设计的框架,且压缩和量化大型模型所面临的挑战依旧存在[18]。未来需在保障大型模型通用性与泛化能力的基础上,继续研发更高效、更实用的量化策略。

参考文献

[1] CHANG Yupeng, WANG Xu, WANG Jindong, et al. A survey on evaluation of large language models[J]. arXiv,2307.03109, 2023.

[2] ZHAO W X, ZHOU Kun, LI Junyi, et al. A survey of large language models[J]. arXiv,2303.18223, 2023.

[3] HUANG J, CHANG K C. Towards reasoning in large language models: A survey [C]//Proceedings of the Findings of the Association for Computational Linguistics. ACL, 2023: 1049-1065.

[4] WEI J, TAY Y, BOMMASANI R, et al. Emergent abilites of Large Language Models[J]. arXiv, 2206.07682v2,2022.

[5] MU Yao, ZHANG Qinglong, HU Mengkang, et al. Embodiedgpt: Vision-language pre-training via embodied chain of thought[J]. arXiv ,2305.15021,2023.

[6] XU Peng, SHAO Wenqi, ZHANG Kaipeng, et al. Lvlm-ehub: A comprehensive evaluation benchmark for large vision-language models[J]. arXiv preprint arXiv,2306.09265, 2023.

[7] ZHANG Yiyuan, GONG Kaixiong, ZHANG Kaipeng, et al. Meta-transformer: A unified framework for multimodal learning [J]. arXiv preprint arXiv,2307.10802, 2023.

[8] HUANG Linyan, WANG Huijie, ZENG Jia, et al. Geometric-aware pretraining for vision-centric 3D object detection[J]. arXiv preprint arXiv,2304.03105, 2023.

[9] HUANG Linyan, LI Zhiqi, SIMA Chonghao, et al. Leveraging vision-centric multi-modal expertise for 3D object detection [C]//Proceedings of the Advances in Neural Information Processing Systems. NeurIPS, 2024:1-15.

[10] SHAO Wenqi, CHEN Mengzhao, ZHANG Zhaoyang, et al. Omniquant: Omnidirectionally calibrated quantization for large language models[J]. arXiv preprint arXiv,2308.13137, 2024.

[11] MA Xinyin, FANG Gongfan, WANG Xinchao. LLM-pruner: On the structural pruning of large language models[J]. arXiv, 2305.11627v3, 2023.

[12] FRANTAR E, ALISTARH D. Sparsegpt: Massive language models can be accurately pruned in one-shot[J]. arXiv, 2301.00774v3, 2023.

[13] SUN Mingjie, LIU Zhuang, BAIR A, et al. A simple and effective pruning approach for large language models[J]. arXiv preprint arXiv,2306.11695, 2023.

[14] ZHANG Chen, YANG Yang, LIU Jiahao, et al. Lifting the curse of capacity gap in distilling language models[J]. arXiv, 2305.12129v1, 2023.

[15] FRANTAR E, ASHKBOOS S, HOEFLER T, et al. Gptq: Accurate post-training quantization for generative pre-trained transformers[J]. arXiv preprint arXiv,2210.17323, 2022.

[16] ZHU Xunyu, LI Jian, LIU Yong, et al. A survey on model compression for Large Language Model [J]. arXiv, 2308.07633v3, 2023.

[17] YAO Zhewei, WU Xiaoxia, LI Cheng, et al. Zeroquant-v2: Exploring post-training quantization in llms from comprehensive study to low rank compensation[J]. arXiv,2303.08302v3, 2023.

[18] WANG Wenxiao, CHEN Wei , LUO Yicong, et al. Model compression and efficient inference for large language models: A survey[J]. arXiv,2402.09748v1. 2024.

[19] 杨春, 张睿尧, 黄泷, 等. 深度神经网络模型量化方法综述[J]. 工程科学学报, 2023, 45(10): 1613-1629.

[20] JACOB B, KLIGYS S, CHEN B, et al. Quantization and training of neural networks for efficient integer-arithmetic-only inference [C]//Proceedings of 2018 IEEE/CVF Conference on Computer Vision and Pattern Recoonition Piscataway, NJ: IEEE, 2018: 2704-2713.

[21] REED J K, DEVITO Z, HE H, et al. Torch. fx: Practical program capture and transformation for deep learning in Python [J]. arXiv, 2112.08429, 2022. https://arxiv. org/abs/ 2112.08429.

[22] SIDDGOWDA S, FOURNARAKIS M, NAGEL M, et al. Neural network quantization with AI model efficiency toolkit (AIMET) [J]. arXiv, 2201.08442, 2022. https://arxiv. org/ abs/2201.08442.

[23] KRISHNAMOORTHI R. Quantizing deep convolutional networks for efficient inference: A whitepaper[J]. arXiv, 2201.08442, 2022. https://arxiv. org/abs/1806.08342.

[24] NAGEL M, FOURNARAKIS M, AMJAD R A, et al. A white paper on neural network quantization[J]. arXiv, 2106.08295. 2021. https://arxiv. org/abs/2106.08295.

[25] LIU Zechun , OGUZ B, ZHAO Changsheng, et al. LLm-qat: Data-free quantization aware training for large language models [J]. arXiv preprint arXiv,2305.17888, 2023.

[26] KIM M, LEE S, HONG S J, et al. Understanding and improving knowledge distillation for quantization-aware training of large

transformer encoders[J]. arXiv preprint arXiv, 2211. 11014, 2022.
[27] NEILL J O, DUTTA S. Self-distilled quantization: Achieving high compression rates in transformer-based language models[J]. arXiv preprint arXiv,2307. 05972, 2023.
[28] KWON S J, KIM J, BAE J, et al. Alphatuning: Quantization-aware parameter-efficient adaptation of large-scale pre-trained language models[J]. arXiv preprint arXiv,2210. 03858, 2022.
[29] KIM J, LEE J H, KIM S, et al. Memory-efficient fine-tuning of compressed large language models via sub-4-bit integer quantization[J]. arXiv preprint arXiv,2305. 14152, 2023.
[30] RASTEGARI M, ORDONEZ V, REDMON J, et al. Xnor-net: Imagenet classification using binary convolutional neural networks[C]//Proceedings of European Conference on Computer Vision. Cham:Springer, 2016: 525-542.
[31] KAUSHAL A, VAIDHYA T, RISH I. Lord: Low rank decomposition of monolingual code llms for one-shot compression[J]. arXiv preprint arXiv,2309. 14021, 2023.
[32] DETTMERS T, PAGNONI A, HOLTAMAN A, et al. Qlora: Efficient finetuning of quantized llms[J]. arXiv preprint arXiv, 2305. 14314, 2023.
[33] LI Yixiao, YU Yifan, LIANG Chen, et al. Loftq: Lora-fine-tuning-aware quantization for large language models[J]. arXiv preprint arXiv,2310. 08659, 2023.
[34] XU Yuhui, XIE Lingxi, GU Xiaotao, et al. Qa-lora: Quantization-aware low-rank adaptation of large language models[J]. arXiv preprint arXiv,2309. 14717, 2023.
[35] HUA W Y, WILLIAMS B, SHAMSI D. Lacos-bloom: Low-rank adaptation with contrastive objective on 8 bits siamese-bloom[J]. arXiv preprint arXiv,2305. 06404, 2023.
[36] CHAI Y J, GKOUNTOURAS J, KO G G, et al. INT2. 1: Towards fine-tunable quantized large language models with error correction through low-rank adaptation[J]. arXiv, 2306. 08162v1. 2023.
[37] LEE J H, KIM J, KWON S J, et al. Flexround: Learnable rounding based on element-wise division for post-training quantization[J]. arXiv preprint arXiv,2306. 00317, 2023.
[38] FRANTAR E, ASHKBOOS S, HOEFLER T, et al. GPTQ: Accurate post-training quantization for generative pre-trained transformers[J]. arXiv,2210. 17323v2. 2023.
[39] CHEE J, CAI Y H, KULESHOV V, et al. Quip: 2-bit quantization of large language models with guarantees[J]. arXiv preprint arXiv,2307. 13304, 2023.
[40] CHENG Wenhua, ZHANG Weiwei, SHEN Haihao, et al. Optimize weight rounding via signed gradient descent for the quantization of llms[J]. arXiv preprint arXiv,2309. 05516, 2023.
[41] TANG Hanlin, SUN Yifu, WU Decheng, et al. EasyQuant: An efficient data-free quantization algorithm for LLMs[J]. arXiv, 2403. 02775v1, 2024.
[42] YAO Zhewei , WU Xiaoxia, LI Cheng, et al. Zeroquant-v2: Exploring post-training quantization in llms from comprehensive study to low rank compensation[J]. arXiv,2303. 08302v3, 2023.
[43] KIM Y J, HENRY R, FAHIM R, et al. FineQuant unlocking efficiency with fine-grained weight-only quantization for LLMs[J]. arXiv,2308. 09723v1. 2023.
[44] ZENG Aohan, LIU Xiao, DU Zhengxiao, et al. Glm-130b: An open bilingual pre-trained model[J]. arXiv preprint arXiv,2210. 02414, 2022.
[45] KIM S, HOOPER C, GHOLAMI A, et al. Squeezellm: Dense-and-sparse quantization[J]. arXiv preprint arXiv,2306. 07629, 2023.
[46] PARK G, PARK B, KWON S J, et al. Nuqmm: Quantized matmul for efficient inference of large-scale generative language models[J]. arXiv preprint arXiv, 2206. 09557, 2022.
[47] LIN Ji, TANG Jiaming, TANG Haotian, et al. AWQ: Activation-aware weight quantization for LLM compression and acceleration[J]. arXiv,2306. 00978v2. 2023.
[48] LI Liang, LI Qingyuan, ZHANG Bo, et al. Norm tweaking: Igh-performance low-bit quantization of large language models[J]. arXiv,2309. 02784, 2023.
[49] DETTMERS T, SVIRSCHEVSKI R, EGIAZARIAN V, et al. SpQR: A sparse-quantized representation for near-lossless LLM weight compression[J]. arXiv,2306. 03078v1, 2023.
[50] DETTMERS T, LEWIS M, BELKADA Y, et al. LLM. int8 ( ): 8 bit matrix multiplication for transformers at scale[J]. arXiv preprint arXiv,2208. 07339, 2022.
[51] LEE C, JIN J, KIM T, et al. Owq: Lessons learned from activation outliers for weight quantization in large language models[J]. arXiv preprint arXiv,2306. 02272, 2023.
[52] ZHANG Yijia, ZHAO Lingran, CAO Shijie, et al. Integer or floating point? new outlooks for low-bit quantization on large language models[J]. arXiv preprint arXiv,2305. 12356, 2023.
[53] WU Xiaoxia, AO Zhewei, HE Yuxiong. Zeroquant-fp, A leap forward in llms post-training w4a8 quantization using floating-point formats[J]. arXiv preprint arXiv,2307. 09782, 2023.
[54] YVINEC E, DAPGONY A, CORD M, et al. Rex: Data-free residual quantization error expansion[J]. arXiv preprint arXiv, 2203. 14645, 2022.
[55] GUO Cong, TANG Jiaming, HU Weiming, et al. Olive: Accelerating large language models via hardware-friendly outlier-victim pair quantization[C]//Proceedings of the 50th Annual International Symposium on Computer Architecture. 2023: 1-15.
[56] YAO Zhewei, AMINABADI R, ZHANG Minjia, et al. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers[J]. arXiv preprint arXiv,2206. 01861, 2022.
[57] BEHDIN K, ACHARYA A, GUPTA A, et al. Quantease: Optimization-based quantization for language models-an efficient and intuitive algorithm[J]. arXiv,2309. 01885v2, 2023.
[58] XIAO G X, LIN J, SEANEC M, et al. Smoothquant: Accurate and efficient post-training quantization for large language models[C]// Proceedings of International Conference on Machine Learning. IMLS, 2023: 38087-38099.
[59] WEI Xiuying, ZHANG Yunchen, LI Yuhang, et al. Outlier suppression+: Accurate quantization of large language models by equivalent and optimal shifting and scaling[J]. arXiv preprint arXiv,2304. 09145, 2023.
[60] YUAN Zhihang, NIU Lin, LIU Jiawei, et al. Rptq: Reorder-based post-training quantization for large language models[J]. arXiv preprint arXiv,2304. 01089, 2023.
[61] WEI Xiuying, ZHANG Yunchen, ZHANG Xiangguo, et al. Outlier suppression: Pushing the limit of low-bit transformer language models[C]//Proceedings of Advances in Neural Information Processing Systems. NeurIPS, 2022:17402-17414.
[62] LI Qingyuan, ZHANG Yifan, LI Liang, et al. Fptq: Fine-grained post-training quantization for large language models[J].

arXiv preprint arXiv,2308. 15987, 2023.
[63]LIU Jing, GONG Ruihao, WEI Xiuying, et al. Qllm: Accurate and efficient low-bitwidth quantization for large language models [J]. arXiv preprint arXiv,2310. 08041, 2023.
[64] DETTMERS T , ZRTTLEMOYER L. The case for 4 - bit precision: K - bit inference scaling laws [C]// Proceedings of International Conference on Machine Learning. IMLS, 2023: 7750-7774.
[65] XU Yuzhuang, HAN Xu, YANG Zonghan, et al. OneBit: towards extremely low - bit large language models [J]. arXiv, 2402. 11295v1, 2024.
[66]MA Shuming, WANG Hongyu, MA Lingxiao, et al. The Era of 1-bit LLMs: All large language models are in 1. 58 bits[J]. arXiv,2402. 17764v1, 2024.
[67] SHANG Yuzhang, YUAN Zhihang, WU Qiang, et al. PB - LLM: Partially binarized large language models [J]. arXiv preprint arXiv,2310. 00034, 2023.
[68] HUANG Wei, LIU Yangdong, QIN Haotong, et al. BiLLM: Pushing the limit of post - training quantization for LLMs [J]. arXiv,2402. 04291v1, 2024.
[69] SHEN Haihao, MENG Hengyu, DONG Bo, et al. An efficient sparse inference software accelerator for transformer - based language models on cpus[J]. arXiv preprint arXiv,2306. 16601, 2023.
[70] PEGOLOTTI T, FRANTAR E, ALISTARH D, et al. QIGen: generating efficient kernels for quantized inference on Large Language Models[J]. arXiv,2307. 03738v1. 2023.
[71] WANG Kuan, LIU Zhijian, LIN Yujun, et al. Haq: Hardware-aware automated quantization with mixed precision [C]// Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Piscataway, NJ: IEEE, 2019: 8612-8620.
[72] RAHMAN M W U, ABRAR M M, COPENING H G, et al. Quantized transformer language model implementations on edge devices[J]. arXiv preprint arXiv,2310. 03971, 2023.
[73] LIN Zheng, QU Guanqiao, CHEN Qiyuan, et al. Pushing large language models to the 6g edge: Vision, challenges, and opportunities[J]. arXiv preprint arXiv,2309. 16739, 2023.