

基于敏感度指导的训练后量化方法

王招元, 王铁军, 石梦茹

(成都信息工程大学 计算机学院, 四川 成都 610225)

摘要:随着大型语言模型(LLM)在问答、翻译、代码生成等自然语言处理任务中的广泛应用,其对计算资源和存储空间消耗迅速增长,严重制约了其在边缘设备或低资源环境中的部署。为此,本研究提出了一种新的敏感度指导的训练后量化方法(Sensitivity-Guided Post-Training Quantization, SGPTQ),旨在保持模型精度的同时,显著降低内存与计算开销。SGPTQ主要通过两项关键机制提升量化效果:首先,提出一种模块敏感度指导的自适应平滑策略,根据模型不同层对量化误差的敏感程度以及激活特征,动态调整权重平滑系数,从而有效减缓量化导致的精度退化;其次,引入非线性伪量化指导机制,先通过高位伪量化获取权重敏感度,再据此指导后续的低位量化决策,避免传统低位量化过程中出现的精度突降问题。SGPTQ采用渐进式两阶段量化策略,实现更稳定且精度更高的压缩过程。在多个主流模型和数据集上进行的推理实验表明,SGPTQ在保持模型推理性能方面优于现有的经典量化方法,尤其在3-bit权重压缩场景下表现更为显著,为边缘设备高效部署LLM提供了实用解决方案。

关键词:大语言模型;训练后量化;自适应平滑;敏感度指导;边缘设备

中图分类号:TP301.6

文献标识码:A

文章编号:1673-629X(2026)01-0105-09

doi:10.20165/j.cnki.ISSN1673-629X.2025.0223

Sensitivity-guided Post-training Quantization Method

WANG Zhao-yuan, WANG Tie-jun, SHI Meng-ru

(School of Computer Science, Chengdu University of Information Technology, Chengdu 610225, China)

Abstract: With the extensive adoption of large language models (LLMs) in natural language processing tasks, including question answering, translation, and code generation, the escalating demands on computational resources and storage capacity have significantly constrained their deployment in edge devices and resource-constrained environments. To address this challenge, we propose a novel post-training quantization approach, termed Sensitivity-Guided Post-Training Quantization (SGPTQ), designed to substantially reduce memory and computational costs while preserving model accuracy. SGPTQ enhances quantization efficacy through two primary mechanisms: Firstly, it introduces an adaptive smoothing strategy driven by module-specific sensitivity, dynamically adjusting weight smoothing coefficients based on the quantization error sensitivity of individual model layers and their activation characteristics, thereby effectively mitigating accuracy degradation induced by quantization. Secondly, it employs a nonlinear pseudo-quantization guidance mechanism that initially derives weight sensitivity via high-bit pseudo-quantization, subsequently informing low-bit quantization decisions to prevent abrupt accuracy declines commonly observed in conventional low-bit quantization. By adopting a progressive two-stage quantization framework, SGPTQ achieves a more stable and accurate compression process. Inference experiments conducted across multiple benchmark models and datasets demonstrate that SGPTQ outperforms existing quantization techniques, particularly in 3-bit weight compression scenarios, offering a robust and practical solution for efficient LLM deployment on edge devices.

Key words: large language models; post-training quantization; adaptive smoothing; sensitivity-guided; edge devices

0 引言

近年来,大语言模型(LLM)^[1]如GPT^[2]系列、BERT^[3]、LLaMA^[4]和DeepSeek^[5]等推动了生成式预训练技术发展。但其庞大的参数规模限制了实际部署。在模型压缩方面,杨扬等人^[6]通过分析卷积神经

网络训练过程中的参数统计特性,提出了一种基于统计分析的模型压缩方法,有效减少了参数数量和计算量。焦小奇^[7]针对预训练语言模型(PLMs)在自然语言处理(NLP)领域的应用中模型尺寸大、推理速度慢的问题,提出了多种模型压缩技术,降低模型的存储和

收稿日期:2025-05-17

修回日期:2025-09-18

基金项目:四川省科技计划重大科技专项项目(2024ZDZX0001)

作者简介:王招元(2001-),男,硕士研究生,研究方向为大语言模型量化;通信作者:王铁军(1979-),男,副教授,硕士,研究方向为高性能计算。

计算需求,提升实际应用能力。

除了上述的模型压缩方法,模型量化也是一种重要的模型压缩技术。量化技术通过将模型参数从高精度压缩至低精度(如 INT8、INT4),可显著降低存储和计算成本。其中,量化感知训练(QAT)^[8]是代表性量化方案之一,它需要对量化模型进行再训练以抵消量化后引入的性能下降,因此量化过程需要较高的计算开销。之前的仅编码器 Transformer^[9]的量化方案(包括 BERT)因模型参数量相对较少,大多直接基于 QAT 方法。例如,LLM-QAT^[10]直接在 LLM 上实现了标准 QAT 框架。多数应用场景,完全重新训练模型参数是一项挑战。因此,包含训练后量化(PTQ)在内的低成本量化方法成为了目前量化的主流。训练后量化的缺点是在低比特配置下会遭受显著的精度损失,从而影响模型性能。然而,现有训练后量化方法仍面临若干关键挑战,主要包括以下几点:

(1) 未能充分考虑不同模块对量化误差的敏感度差异,导致统一量化策略下的性能损失。

(2) 缺乏渐进式的量化机制,可能导致精度下降。

(3) 部分方法通过混合精度量化以维持模型性能,尽管可在一定程度上降低精度损失,但却引发推理效率下降,难以满足高效部署的需求。

为此,本研究提出了敏感度指导的渐进式量化(Sensitivity - Guided Post - Training Quantization, SGPTQ),针对 LLM 模块敏感度差异来优化权重量化的方法,通过自适应平滑和非线性伪量化指导策略提升训练后量化的准确性,同时保持高压缩比,为高效部署 LLM 提供新思路。通过对权重的自适应平滑操作能够有效地保留权重的关键信息,减少量化的误差,同时也不会导致混合精度造成的推理速度慢等问题。非线性伪量化指导策略采用了渐进式的量化机制,针对不同模块对量化误差的敏感度差异采取不同的量化策略(调整相应量化参数),避免了统一量化造成的精度损失。图 1 展示了使用 SGPTQ 等方法将 OPT 和 BLOOM 模型量化到 4-bit 精度的性能比较。

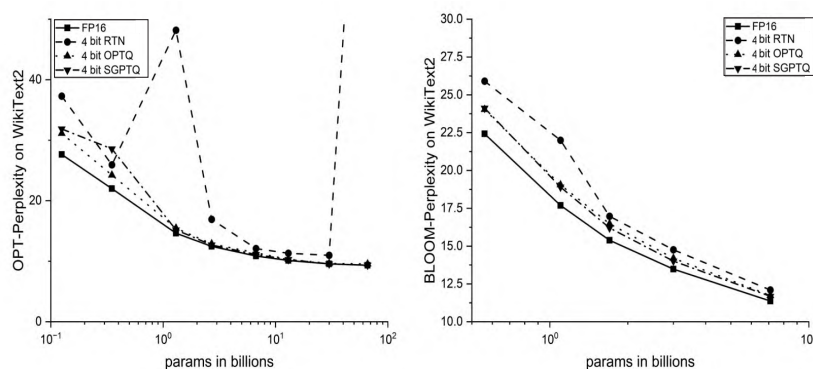


图 1 不同量化方法将 OPT 和 BLOOM 模型量化到 4-bit 的困惑度对比

SGPTQ 的目的是提高训练后量化 LLM 的准确性。SGPTQ 引入了自适应平滑机制和非线性量化指

导策略,在保持高压缩比的同时提供更准确的量化表示。SGPTQ 的整体量化架构如图 2 所示。

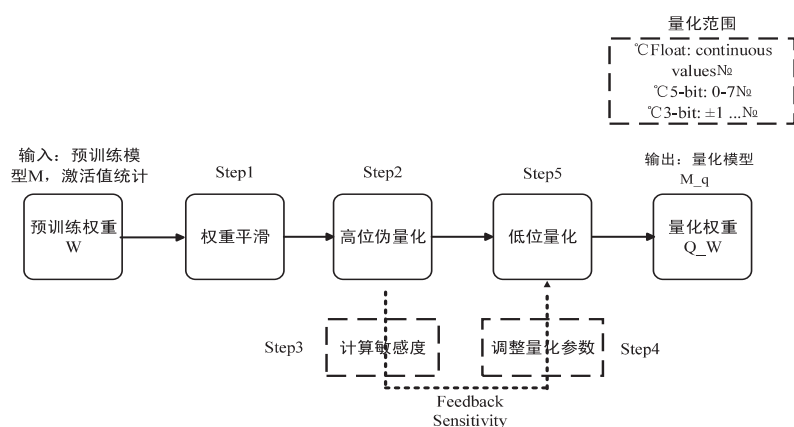


图 2 SGPTQ 方法的架构

SGPTQ 主要通过两个步骤实现低位量化。首先,通过收集到的激活值信息及误差来自适应调整平滑因子进行权重平滑操作,然后采用非线性伪量化计算敏感度,最后通过计算出的敏感度信息指导低位量化。

该文的主要贡献概括如下:

(1) 提出了一种基于模块敏感度的自适应平滑机制,通过该平滑机制动态优化不同层系的量化误差。

(2) 采用敏感度指导策略,通过非线性伪量化特

征迭代修正低位决策。

(3) 在 2 个公开数据集上进行实验,验证了 SGPTQ 方法的有效性。

1 相关工作

模型量化通过降低深度神经网络权重的比特精度来压缩模型规模并加速推理。当前 LLM 量化主要聚焦两种方案:全量化 (W8A8)^[11] 与低位权重量化 (W4A16)^[12]。前者将权重 (Weight) 与激活 (Activation) 统一量化为 8-bit 整数^[13],虽能降低计算单元开销,但受激活值动态范围的影响,易导致分布失配问题^[14]。后者则专注于权重的低位压缩 (如 3/4-bit),通过保留激活的 fp16 精度平衡效率与性能。该文聚焦于后一方向,因其在降低内存占用和加速推理方面更具实用性。

在低位权重量化中,OPTQ 基于最优脑量化 (OBQ) 框架^[15],提出逐列量化策略:利用 Hessian 矩阵 ($H_F = 2X_F X_F^T$) 补偿量化误差,其中 F 表示剩余的全精度权重集合。它首先按行确认量化参数,然后逐列量化权重,并实时更新未量化的权重。尽管该方法展示了在大语言模型 (LLMs) 中实现 4-bit 量化的潜力,但当模型尺寸减小或问题复杂度增加时,相较于 fp16 基线,模型精度会有所下降。GPTQT^[16] 通过两阶段量化框架进一步优化效率;OWQ^[17] 通过混合精度量化以维持模型性能。AWQ^[18] 通过激活统计保护关键权重通道,优先保留对模型性能影响大的权重。

SGPTQ 在多个关键点上相较现有方法展现出本质区别。首先,与 AWQ 基于激活统计实现通道加权的平滑机制不同,SGPTQ 从模型结构出发,提出了模块敏感度指导的自适应平滑策略。该策略通过分析模型中各模块对量化误差的敏感程度,动态调整权重的平滑系数,使得对精度影响较大的模块获得更强的保护,有效缓解量化带来的性能退化。

其次,SGPTQ 引入了非线性伪量化指导机制,区别于 OPTQ 等方法中采用逐层误差补偿和量化重建策略的做法。该机制先在高位伪量化阶段估计权重的量化敏感度,再据此指导后续的真实低位 (如 3-bit/4-bit) 量化决策,从根源上避免直接低位量化带来的突发性精度损失,从而提升压缩过程的稳定性与鲁棒性。

2 方法

本节详细阐述了 SGPTQ 量化方法的实现细节。SGPTQ 旨在通过动态平滑机制和非线性伪量化指导策略,优化 LLMs 在低位权重量化过程中的性能。本方法包括动态平滑机制和非线性伪量化指导两个核心模块,以下分别进行详细介绍。

2.1 权重平滑策略

Tang 等人^[19]的研究表明,较小的重构误差并不能保证模型性能更好。直接缩小量化范围会将大多数异常值截断到非常小,但这些异常值对于保持模型的性能至关重要,因此困惑度 (Perplexity, PPL) 会严重增加。相反,当保持这些异常值时,量化模型会随着重构误差的不断减小而获得更好的性能。该研究表明,异常值的权重比正常值更重要。但是,若直接保留权重的原始精度,将会带来推理速度降低这一问题。为了解决该问题,通过激活统计数据确定每个权重通道的缩放比例,并基于对模型结构中不同模块的量化敏感度的研究结果,提出了一种模块敏感度指导的平滑策略。

2.1.1 模块敏感度指导的平滑策略分析

针对 BLOOM^[20] 和 OPT^[21] 两个模型家族,该文设计了三个不同的平滑策略:(1) 全参数平滑 (All),对所有线性层进行激活感知缩放;(2) 仅前馈网络平滑 (MLP),仅对前馈网络实施平滑处理;(3) 仅自注意力平滑 (S-A),只针对自注意力模块实施平滑处理。

表 1 的实验结果表明,在不同的数据集和模型配置下,以上三种权重平滑策略对模型性能的影响存在显著差异。对 BLOOM 模型来说,自注意力模块的量化敏感度更高。在 WikiText2^[22] 数据集上,仅平滑自注意力层较全参数平滑 PPL 平均降低了 3.7%,而仅平滑前馈网络导致性能劣化 1.1%。而对 OPT 模型,前馈网络层的量化误差影响更显著。OPT-6.7B 采用全参数平滑时 PPL 为 13.65,而仅平滑前馈网络时 PPL 上升至 15.37。

表 1 BLOOM 和 OPT 模型在 WikiText2 和 PTB 数据集上不同平滑策略的平滑至 3-bit 的 PPL 结果

Model	Smooth	560M	1.1B	1.7B	3B	7.1B
BLOOM- WikiText2	All	36.87	27.75	21.24	17.23	13.65
	MLP	35.57	27.57	21.48	17.66	13.84
	S-A	34.06	24.49	20.63	16.97	13.58
BLOOM- PTB	All	73.96	77.74	43.00	30.55	24.12
	MLP	69.08	69.72	42.50	31.01	24.58
	S-A	66.74	69.72	42.86	29.97	23.89
Model	Smooth	125M	350M	1.3B	2.7B	6.7B
OPT- Wiki Text2	All	22.25	26.841	22.25	17.35	13.65
	MLP	21.50	65.648	21.50	16.80	15.37
	S-A	21.65	26.100	21.65	17.33	13.67
OPT- PTB	All	50.31	24.777	26.38	20.61	16.96
	MLP	52.54	68.425	26.50	20.29	18.59
	S-A	50.90	11.209	26.54	21.13	16.69

2.1.2 机制解释与策略选择

实验表明,Transformer 模型的量化敏感性因架构差异而异。BLOOM 模型的自注意力模块对量化误差影响显著,这个结果与该模块负责建模全局依赖关系、权重精确性直接影响语义连贯性的核心作用一致。相比之下,尽管其自注意力层具有较强的参数冗余,OPT 模型的前馈网络 (MLP) 表现出更高的精度需求。在后续实验中,采用的平滑方式均为仅平滑自注意力模块。OPT-350M 平滑后的困惑度 (PPL) 异常升高,可能因其小规模对量化误差和平滑的较高敏感性,导致权重分布被过度扰动。为缓解 OPT-350M 的量化异常,该文提出以下特定优化策略:

(1) 调整 act_scales 计算方式:采用 99% 分位数替代最大值,减少异常激活影响,并结合平滑前的权重分析动态调整缩放因子。对于离群值低于 0.5% 的层,缩放因子乘以 0.5,以减少权重扰动。

(2) 引入平滑前的权重分析:计算 act_scales 前,分析 OPT-350M 模型每层的离群值,为选择性平滑提供依据。

(3) 根据层重要性选择性平滑:基于层的综合重要性评分(计算方法详见 2.2 节),跳过对低重要性层的平滑操作。

表 2 展示了改进后的平滑方法 (OPT-Smooth) 对 OPT-350M 模型量化性能的修复效果。在 WikiText2 和 PTB 数据集上,PPL 分别由 26 841 和 24 777 降低至 46.01 和 54.94,验证了该方法在缓解量化异常方面的有效性。然而,优化后 PPL 仍偏高,表明 OPT-350M 对平滑操作较为敏感。该现象可能源于 OPT-350M 较小的参数规模及其权重分布的高度均匀性(离群值占比仅为 0.2% ~ 0.7%),使得平滑操作引发的微小扰动在量化过程中被放大,从而影响模型性能。

表 2 OPT-350M 在不同平滑策略下的困惑度比较

PPL ↓	Bits	Smooth	OPT-Smooth	RTN
WikiText2	3	26 841	46.01	64.57
PTB	3	24 777	54.94	88.04

为提升整体适应性,该文进一步提出了基于动态调整的自适应平滑策略(详见 2.2 节),通过结合激活方差与权重梯度动态优化平滑系数,以适配更广泛的模型结构。针对 OPT-350M 的异常表现,其优化策略优先聚焦于特定调整,以有效应对其高敏感性问题。后续自适应平滑方法将作为通用框架,进一步扩展至其他模型。

2.2 动态自适应平滑机制

动态平滑机制通过自适应调整平滑系数,根据模型各模块对量化误差的敏感度差异对权重进行平滑处

理,从而有效减少量化误差的影响。该机制的实现分为三个阶段。

(1) 收集统计信息,为量化过程中的敏感度分析提供基础。针对模型的第 l 层,计算激活值绝对值均值 $\mu_{\text{act},l}$ 。

$$\mu_{\text{act},l} = \frac{1}{N} \sum_{i=1}^N |a_{l,i}| \quad (1)$$

其中, $a_{l,i}$ 表示第 l 层的第 i 个激活值, N 为激活值样本数。

(2) 计算综合重要性评分。为准确评估每一层对量化误差的敏感度,引入多维度重要性评分机制,包括激活值重要性 $I_{a,l}$ 和量化误差影响 E_{imp} 两个指标。

$$I_{a,l} = \frac{\mu_{\text{act},l}}{\max_k \mu_{\text{act},k}} \quad (2)$$

$$E_{\text{imp}} = \frac{\|\nabla wL\|_{2,l}}{\max_k \|\nabla wL\|_k} \quad (3)$$

其中, $\|\nabla wL\|_{2,l}$ 表示第 l 层权重的梯度 L_2 范数,反映量化误差对模型损失的潜在影响。综合上述指标,计算第 l 层的综合重要性分数。

$$I_l = \alpha I_{a,l} + \beta E_{\text{imp}} \quad (4)$$

其中, α 、 β 为可调的权重系数,用于平衡不同指标的贡献。

(3) 生成动态平滑系数。基于第 l 层的综合重要性分数 I_l ,计算基础平滑系数 $\lambda_{l,\text{base}}$ 。

$$\lambda_{l,\text{base}} = \frac{1}{1 + e^{-I_l}} \quad (5)$$

该公式通过 Sigmoid 函数将重要性分数映射到 (0,1) 区间。为了进一步适应层的统计特性,引入自适应调整。

$$\lambda_l = \lambda_{l,\text{base}} \times \left(1 + \delta \cdot \frac{\sigma_{w,l}^2}{\varepsilon + \mu_{\text{act},l}}\right) \quad (6)$$

其中, δ 为调节参数, ε 为数值稳定性常数,避免除零问题。这种调整确保平滑系数能够动态反映权重的分布和激活值的特性。

2.3 敏感度指导的量化阶段

SGPTQ 中的非线性伪量化指导策略是实现高效低位权重量化的核心步骤。SGPTQ 通过高位伪量化所计算的敏感度指导来避免直接进行低位量化带来的精度损失。

2.3.1 非线性伪量化的敏感度计算

首先,对权重 w 进行高位量化(例如 5-bit),以获取初始量化权重 w_{init} 。与 OPTQT 不同,引入了一种基于激活值梯度的敏感度指标 $\text{Sens}_i(s_i)$,用于衡量权重对模型输出的影响。

$$s_i = \left| \frac{\partial L}{\partial w_i} \cdot \text{std}(A_i) \right| \quad (7)$$

其中, L 为损失, $\frac{\partial L}{\partial w_i}$ 为权重 w_i 的梯度, $\text{std}(A_i)$ 为对应激活值的标准差。这种敏感度计算结合了任务特异性和激活分布,能充分体现权重对模型输出的影响。

高位量化过程为:

$$w_{\text{init},i} = \text{round}\left(\frac{w_i - \mu A_i}{s_i \cdot s_{\text{init}}} \cdot (2^{b_{\text{init}}} - 1)\right) \quad (8)$$

其中, μA_i 为激活均值, s_{init} 为初始缩放因子, $b_{\text{init}} = 5$ 。通过敏感度调整缩放,这一步骤为后续低位量化提供了关键信息。

2.3.2 敏感度指导的低位量化

利用高位量化计算得到的敏感度 Sens_i ,设计了一种动态量化区间分配策略,将权重进一步降至低位(例如 3-bit 或 4-bit)。具体而言,根据敏感度大小动态调整量化步长 Δ_i 。

$$\Delta_i = \frac{s_i}{\sum_i s_i} \cdot (W_{\max} - W_{\min}) \cdot \frac{1}{2^{b_{\text{final}}}} \quad (9)$$

其中, b_{final} 为目标位数(例如 3-bit)。最终低位量化权重为:

$$w_{q,i} = \text{round}\left(\frac{w_{\text{init},i}}{\Delta_i}\right) \cdot \Delta_i \quad (10)$$

这一过程通过敏感度动态分配量化区间,优先保留高敏感度权重的信息,从而优化模型性能。具体的 SGPTQ 算法流程见算法 1。

算法 1:SGPTQ 算法流程

输入:预训练模型 M , 激活值统计信息 act_scales

输出:量化模型 M_q

(1) For layer_i in M // Smooth 阶段

(a) $\text{importance}_i = \text{ComputeImportance}(\text{layer}_i)$ // 计算层重要性分数

(b) $\text{act_score} = \text{GetActScore}(\text{layer}_i, \text{act_scales})$ // 获取激活值特征

(c) $\text{importance_score} = \text{ComputeScore}(\text{importance}_i, \text{act_score})$ // 计算综合评分

(d) $\alpha = \text{ComputeAlpha}(\text{importance_score})$ // 计算平滑系数

(e) $\text{Smooth}(\text{layer}_i, \alpha)$ // 平滑权重

(2) End for

(3) For layer_i in M // 量化阶段

(a) $\text{Sensitivity}_i = \text{QuantizeHigh}(\text{layer}_i.\text{weight}, s_{\text{init}}, \mu A_i)$

// 非线性伪量化计算敏感度

(b) $\text{AdjustQuantParams}(\text{layer}_i, \text{sensitivity})$ // 调整量化参数

(c) $\text{ApplyFinalQuantization}(\text{layer}_i)$ // 应用最终量化

(4) Return M_q

2.4 评估方法

实验部分主要使用两类评估指标:困惑度(Perplexity, PPL)和准确率(Accuracy)。

在语言生成任务中,该文采用困惑度(PPL)作为核心评估指标,衡量语言模型量化后的推理能力。数据集选用 WikiText2 和 PTB^[23]以验证泛化性。困惑度评估概率分布或语言模型预测样本的有效性,本质上是衡量模型在预测序列中的下一个单词时的意外程度。困惑度值越低,表示性能越优异,表明模型置信度越高,遇到的数据意外程度越低。其计算公式为:

$$P_{\text{PL}} = \exp\left(-\frac{1}{N} \sum_{i=1}^N \log P(w_i)\right) \quad (11)$$

其中, P_{PL} 为困惑度, N 为词数, $P(w_i)$ 为模型对词 w_i 的预测概率。困惑度值越低,表示模型性能越好,生成文本更符合语言规律。表 1~4 以及表 8 展示了不同模型与量化方法在上述两个数据集上的困惑度。

其次,为评估量化后的模型在理解类任务中的表现,对应指标为准确率(Accuracy),在 PIQA 和 ARC-Easy 数据集上对 OPT 和 BLOOM 模型进行了 Zero-shot 推理实验。在 Zero-shot 设置中,模型不使用任何针对 PIQA 或 ARC-Easy 的训练数据或微调,直接基于预训练权重进行推理。输入通常是一个问题及其候选答案,准确率计算公式为:

$$A = \frac{M}{N} \quad (12)$$

其中, A 表示准确率, M 表示正确预测的样本数量, N 表示总样本数量。表 5 和表 6 展示了不同模型与量化方法在上述两个数据集上的准确率。

3 实验讨论与结果分析

以困惑度和准确度为核心指标,实验主要评估 SGPTQ 在语言生成任务和语言理解两类任务上的量化效果。实验中,主要比较了 SGPTQ、RTN^[24]、OPTQ 三种大型语言模型(LLM)的量化方法。针对 OPT 与 BLOOM 模型家族,数据集选用 WikiText2 和 PTB 以验证泛化性。实验硬件配置方面,参数规模小于 13B 的模型在 NVIDIA 3090 GPU 上完成量化处理,而更大规模模型则在 NVIDIA A800 GPU(80 GB 显存)上进行处理以满足内存需求。

3.1 量化 OPT 模型结果

3.1.1 OPT 模型在 WikiText2 上的量化结果

如表 3 所示,在 4-bit 量化条件下,SGPTQ 在中大型模型(1.3B 至 66B)中表现优异,兼顾精度与鲁棒性,明显优于 RTN 的不稳定输出,也优于 OPTQ 在部分场景下的表现。特别是在 3-bit 量化下,SGPTQ 在全系列模型中均展现出良好效果。小模型上,其保留原始特征的能力更强,PPL 更低;随着模型规模扩大,改进幅度虽略减,但仍具优势。尤其在 OPT-66B 上,OPTQ 性能明显下降,而 SGPTQ 成功将 PPL 降低

4.01,接近全精度模型水平。

3.1.2 OPT 模型在 PTB 上的量化结果

在 PTB 上,4-bit 量化下 RTN 波动较大,OPT-66B 上 PPL 升至 225.66;OPTQ 较为稳定但仍与全精度有差距。除 OPT-350M 外,SGPTQ 在各规模下均保持更小精度损失。3-bit 量化进一步放大差异,RTN 性能剧烈退化,OPTQ 在大模型上也明显退步(如 OPT-66B 达 28.12),而 SGPTQ 依然稳定,OPT-30B 上 PPL 仅为 12.58。

对比表3中的结果可见,SGPTQ 在大多数 OPT 模型中实现了显著性能提升,尤其在 3-bit 量化下,几乎在所有模型规模上均优于 OPTQ,困惑度更低。

3.2 量化 BLOOM 模型结果

如表4所示,对于 BLOOM 模型,SGPTQ 同样在 WikiText2 数据集上优于早期的量化方法。虽然 OPTQ 在这类 LLM 上的表现相对更好,但整体上仍略逊于 SGPTQ。

表3 OPT 模型在 WikiText2 和 PTB 上的 PPL 结果

Method	Dataset	Bits	125M	350M	1.3B	2.7B	6.7B	13B	30B	66B
Full	WikiText2	16	27.65	22.00	14.63	12.47	10.86	10.13	9.56	9.34
RTN		4	37.28	25.94	48.17	16.92	12.10	11.32	10.98	110
OPTQ		4	31.12	24.24	15.47	12.87	11.39	10.31	9.63	9.55
SGPTQ		4	31.86	28.56	15.18	12.69	11.11	10.29	9.56	9.35
RTN		3	1.3e3	64.57	1.3e4	1.6e4	5.8e3	3.4e3	1.6e3	6.1e3
OPTQ		3	53.85	33.79	20.97	16.88	14.86	11.61	10.27	14.16
SGPTQ		3	49.86	44.55	19.72	15.73	12.33	11.69	10.22	9.89
Full		16	32.55	26.08	16.96	15.11	13.08	12.34	11.84	11.28
RTN	PTB	4	53.89	36.79	57.30	31.05	18.84	16.51	15.40	225.66
OPTQ		4	45.17	34.52	21.85	19.14	16.56	14.94	14.26	13.81
SGPTQ		4	38.97	36.66	17.71	15.60	13.39	12.53	11.99	11.45
RTN		3	1.4e3	88.04	1.3e4	1.4e4	5.7e3	2.8e3	1.2e3	5.0e3
OPTQ		3	73.19	47.08	32.10	24.81	21.88	16.68	15.36	28.12
SGPTQ		3	72.42	51.81	23.69	18.98	14.97	13.64	12.58	12.01

表4 BLOOM 模型在 WikiText2 和 PTB 上的 PPL 结果

Method	Dataset	Bits	560M	1.1B	1.7B	3B	7.1B
Full	WikiText2	16	22.42	17.69	15.39	13.48	11.37
RTN		4	25.90	22.00	16.97	14.76	12.10
OPTQ		4	24.03	19.05	16.48	14.20	11.73
SGPTQ		4	24.12	18.90	16.20	14.01	11.68
RTN		3	57.08	50.19	63.59	39.36	17.38
OPTQ		3	32.31	25.08	21.11	17.40	13.47
SGPTQ		3	32.58	23.68	19.99	16.74	13.31
Full		16	41.23	46.96	27.91	23.11	19.39
RTN	PTB	4	51.10	66.85	33.58	27.68	22.42
OPTQ		4	46.97	62.47	31.84	26.49	21.67
SGPTQ		4	44.29	51.09	29.63	24.36	20.13
RTN		3	126	185	106	66.78	35.04
OPTQ		3	70.35	87.04	46.11	34.02	26.14
SGPTQ		3	62.02	67.68	40.68	29.51	23.35

在 PTB 数据集上进行 4-bit 量化时,SGPTQ 优于 OPTQ 和 RTN;当量化精度降至 3-bit 时,这些方法的性能差距进一步拉大,SGPTQ 在 BLOOM-7.1B 上的

困惑度为 23.35,仍显著优于 OPTQ 和 RTN。

上述实验结果充分证明了 SGPTQ 方法在不同数据集、不同模型、不同参数量、不同量化位数下的稳定

性和优越性,特别是在低位量化场景下的表现尤为突出。

3.3 其他实验

3.3.1 Zero-shot 实验

为验证 SGPTQ 在 Zero-shot 任务中的效果,在 OPT 和 BLOOM 模型上进行了 PIQA 与 Arc_easy 测试,对比全精度模型(Full)与多种量化方法(RTN、OPTQ、SGPTQ)在 4-bit 与 3-bit 下的表现。表 5 展示

了 OPT 模型在上述任务中的准确率。

结果表明,SGPTQ 在大多数模型与任务中优于 OPTQ,尤其在 3-bit 量化下提升显著。例如,在 Arc_easy 任务中,OPT-13B 模型下 SGPTQ 准确率为 59.73%,较 OPTQ 提升 2.91 百分点,接近全精度表现,远高于 RTN 的 30.60%。这验证了 SGPTQ 在极低精度下的性能保持能力。

表 5 OPT 模型在 PIQA 和 Arc_easy 上的准确率

Method	Dataset	Bits	125M/%	350M/%	1.3B/%	2.7B/%	6.7B/%	13B/%
Full	PIQA	16	62.02	64.74	72.36	74.81	76.39	76.88
RTN		4	61.43	63.44	67.63	73.72	76.44	76.01
OPTQ		4	61.26	63.71	70.73	73.99	76.28	76.61
SGPTQ		4	61.53	64.00	70.78	73.61	76.27	76.44
RTN		3	56.09	60.61	52.77	51.90	50.49	52.99
OPTQ		3	59.25	61.32	68.34	71.38	73.29	75.24
SGPTQ		3	59.14	62.51	68.49	71.59	74.59	75.95
Full	Arc_easy	16	39.69	40.36	50.93	54.34	60.14	61.83
RTN		4	36.32	38.55	49.20	52.90	57.68	61.31
OPTQ		4	39.02	37.92	59.97	53.11	59.72	61.32
SGPTQ		4	38.17	38.72	49.70	54.29	59.88	61.70
RTN		3	30.43	36.07	27.97	26.05	25.04	30.60
OPTQ		3	36.15	36.91	56.17	48.19	53.41	56.82
SGPTQ		3	35.39	36.86	46.33	49.83	56.18	59.73

表 6 BLOOM 模型在 PIQA 和 Arc_easy 上的准确率

Method	Dataset	Bits	560M/%	1.1B/%	1.7B/%	3B/%	7.1B/%
Full	PIQA	16	65.07	67.14	69.97	70.51	73.72
RTN		4	53.11	65.29	67.74	69.86	72.69
OPTQ		4	64.31	66.05	68.77	69.42	72.96
SGPTQ		4	63.98	66.70	69.09	69.85	72.85
RTN		3	58.60	60.80	60.88	66.28	69.70
OPTQ		3	61.62	62.62	65.18	68.34	70.95
SGPTQ		3	61.75	63.16	65.34	68.11	71.10
Full	Arc_easy	16	41.71	45.41	48.11	53.24	57.37
RTN		4	39.40	42.51	44.70	51.35	56.14
OPTQ		4	40.24	44.49	44.49	52.82	56.14
SGPTQ		4	41.10	46.00	45.80	52.92	56.30
RTN		3	45.44	46.87	37.58	45.08	48.61
OPTQ		3	39.14	41.79	42.85	46.63	51.56
SGPTQ		3	40.50	41.20	43.50	46.80	52.90

表 6 展示了 BLOOM 模型的准确率。尽管提升幅度相对较小,SGPTQ 在各规模下依然保持稳定优势。以 BLOOM-7.1B 为例,3-bit 下 SGPTQ 准确率为 52.90%,高于 OPTQ 的 51.56% 与 RTN 的 48.61%。

说明 SGPTQ 具备更强的低比特精度保留能力。

实验表明,SGPTQ 在 Zero-shot 任务中,尤其在 3-bit 量化下,相较 OPTQ 展现出更优精度与更强泛化性。

3.3.2 推理速度对比

为评估 SGPTQ 的推理效率,参考 OPTQ 的 benchmark 流程,对不同规模模型在固定输入长度(128 tokens)下进行逐 token 推理测试,记录每步前向耗时(ms/token),并取中位数结果以减小噪声干扰。实验基于单张 NVIDIA RTX 3090,使用 PyTorch 实现。

表7 不同量化方法下的 OPT 模型推理时间

Method	Bits	125M	350M	1.3B	2.7B	6.7B
Full	16	7.27	14.57	14.96	20.26	24.03
OPTQ	4	7.44	13.90	15.10	24.46	24.00
SGPTQ	4	7.46	13.98	14.60	23.80	23.48
OPTQ	3	7.55	15.05	15.67	21.89	24.47
SGPTQ	3	7.67	15.80	15.05	21.85	23.83

从结果可以看出,在低吞吐量场景下,SGPTQ 在推理速度上的表现整体优于标准的 OPTQ,尤其在模型规模逐渐增大的情况下更为明显。例如,在 6.7B 模型上,SGPTQ-4bit 的推理时间为 23.48 ms/token,相比于 OPTQ 的 24.00 ms/token 有所下降。对于 3-bit 量化场景,SGPTQ 同样在多个模型尺寸上实现了

表7 显示 SGPTQ 整体推理速度优于 OPTQ,优势随模型规模增加更为明显。如在 6.7B 模型上,SGPTQ-4bit 推理时间为 23.48 ms/token,低于 OPTQ 的 24.00 ms/token。在 3-bit 量化下,SGPTQ 同样在多个模型上展现出更高效率。

更优的执行效率。

3.4 消融实验

为验证 SGPTQ 的关键组件的有效性,在不同规模的 BLOOM 模型上设计了消融对比实验,结果如表 8 所示。

表8 BLOOM 模型在 PTB 的 PPL 结果

Method	560M	1.1B	1.7B	3B	7.1B
OPTQ	70.35	87.04	46.11	34.02	26.14
Smooth+OPTQ	66.74	69.72	42.86	29.97	23.89
Sensitivity+OPTQ	62.34	78.28	41.33	30.11	24.72
SGPTQ	62.02	67.68	40.68	29.51	23.35

OPTQ 采用线性量化并通过网格搜索优化限幅区间,虽在 4-bit 下表现尚可,但在 3-bit 时精度下降显著。

Smooth+OPTQ 在 OPTQ 基础上引入权重平滑,有效优化权重分布,显著提升量化精度。在 BLOOM-3B 和 7.1B 模型上,PPL 分别下降约 4.05 和 2.25。

Sensitivity+OPTQ 引入层敏感度指导机制,进一步提升精度,在中大模型上效果尤为突出,验证了敏感度指导的有效性。

通过综合对比,SGPTQ 融合高位敏感度优化与低位伪量化引导策略,在全部 BLOOM 模型上均取得最优表现,展现了出色的泛化能力与鲁棒性。

4 结束语

该文提出了一种基于敏感度指导的训练后量化方法——SGPTQ,通过自适应权重平滑与非线性伪量化指导相结合,有效缓解了低位宽量化带来的精度损失。在 OPT 与 BLOOM 模型上的系统实验表明,SGPTQ 在多数量化场景下均优于现有方法,尤其在 3-bit 低精度配置下,展现出显著的性能优势,验证了其在大模型

压缩与高效部署中的实用价值。

尽管 SGPTQ 在保持精度与压缩比之间取得了良好平衡,但仍存在一定局限性;量化过程在大型模型上的计算开销较高,模块敏感度差异的处理在部分任务中存在不稳定性,且在特定规模模型(如 OPT-350M)上的通用性尚需进一步验证。

总体而言,SGPTQ 为大语言模型的低比特量化提供了有效的新思路,展现出良好的泛化能力和实际应用潜力。未来工作可围绕提升量化效率、拓展任务类型及模型结构,进一步完善和推广该方法。

参考文献:

- [1] THIRUNAVUKARASU A J, TING D S J, ELANGO VAN K, et al. Large language models in medicine [J]. Nature Medicine, 2023, 29(8): 1930-1940.
- [2] KALYAN K S. A survey of GPT-3 family large language models including ChatGPT and GPT-4 [J]. Natural Language Processing Journal, 2024, 6: 100048.
- [3] DEVLIN J, CHANG M W, LEE K, et al. BERT: pre-training of deep bidirectional transformers for language understanding [C]//Proceedings of the 2019 conference of the North A-

- merican chapter of the association for computational linguistics; human language technologies, volume 1 (long and short papers). Minneapolis: Association for Computational Linguistics, 2019: 4171–4186.
- [4] TOUVRON H, LAVRIL T, IZACARD G, et al. LLaMA: open and efficient foundation language models[J]. arXiv: 2302.13971, 2023.
- [5] DEEPSEEK-AI, GUO D, YANG D, et al. DeepSeek-R1: incentivizing reasoning capability in LLMs via reinforcement learning[J]. arXiv: 2502.05238, 2025.
- [6] 杨 扬, 蓝章礼, 陈 巍. 基于统计分析的卷积神经网络模型压缩方法[J]. 计算机系统应用, 2018, 27(8): 49–55.
- [7] 焦小奇. 预训练语言模型的压缩技术研究[D]. 武汉: 华中科技大学, 2024.
- [8] JACOB B, KLIGYS S, CHEN B, et al. Quantization and training of neural networks for efficient integer–arithmetic–only inference[C]//Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR). Salt Lake City: IEEE, 2018: 2704–2713.
- [9] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is all you need[J]. Advances in Neural Information Processing Systems, 2017, 30: 5998–6008.
- [10] LIU Z, OGUZ B, ZHAO C, et al. LLM–QAT: data–free quantization aware training for large language models[J]. arXiv: 2305.17888, 2023.
- [11] DETTMERS T, LEWIS M, BELKADA Y, et al. Gpt3. int8(): 8-bit matrix multiplication for transformers at scale[J]. Advances in Neural Information Processing Systems, 2022, 35: 30318–30332.
- [12] KIM S, HOOPER C, GHOLAMI A, et al. SqueezeLLM: dense– and –sparse quantization[J]. arXiv: 2306.07629, 2023.
- [13] XIAO G, LIN J, SEZNEC M, et al. SmoothQuant: accurate and efficient post–training quantization for large language models[C]//Proceedings of the 40th international conference on machine learning (ICML 2023). Honolulu: PMLR, 2023: 38087–38099.
- [14] YAO Z, YAZDANI AMINABADI R, ZHANG M, et al. ZeroQuant: efficient and affordable post–training quantization for large–scale transformers[J]. Advances in Neural Information Processing Systems, 2022, 35: 27168–27183.
- [15] FRANTAR E, ASHKBOOS S, HOEFLER T, et al. GPTQ: accurate post–training quantization for generative pre–trained transformers[J]. arXiv: 2210.17323, 2022.
- [16] GUO Y, LANG Y, REN Q. GPTQT: quantize large language models twice to push the efficiency[C]//Proceedings of the 2024 IEEE international conference on cybernetics and intelligent systems (CIS) and IEEE international conference on robotics, automation and mechatronics (RAM). Hangzhou: IEEE, 2024: 368–373.
- [17] LEE C, JIN J, KIM T, et al. OWQ: outlier–aware weight quantization for efficient fine–tuning and inference of large language models[C]//Proceedings of the AAAI conference on artificial intelligence. Vancouver: AAAI, 2024: 13355–13364.
- [18] LIN J, TANG J, TANG H, et al. AWQ: activation–aware weight quantization for on–device LLM compression and acceleration[J]. Proceedings of Machine Learning and Systems, 2024, 6: 87–100.
- [19] TANG H, SUN Y, WU D, et al. EasyQuant: an efficient data–free quantization algorithm for LLMs[J]. arXiv: 2403.02775, 2024.
- [20] LE SCAO T, FAN A, AKIKI C, et al. BLOOM: a 176B–parameter open–access multilingual language model[J]. arXiv: 2211.05100, 2022.
- [21] ZHANG S, ROLLER S, GOYAL N, et al. OPT: open pre–trained transformer language models[J]. arXiv: 2205.01068, 2022.
- [22] MELIS G, DYER C, BLUNSOM P. On the state of the art of evaluation in neural language models[J]. arXiv: 1707.05589, 2017.
- [23] MARCUS M, SANTORINI B, MARCINKIEWICZ M A. Building a large annotated corpus of English: The Penn Treebank[J]. Computational Linguistics, 1993, 19(2): 313–330.
- [24] KRISHNAMOORTHY R. Quantizing deep convolutional networks for efficient inference: a whitepaper[J]. arXiv: 1806.08342, 2018.