SqueezeLLM:密集和稀疏量化

Sehoon Kim*
sehoonkim@berkeley.edu 加州大学伯克利分校

科尔曼·胡珀*
chooper@berkeley.edu 加州大学伯克利分校

Amir Gholami **
amirgh@berkeley.edu ICSI,加州大学伯克利分校

董震
zhendong@berkeley.edu 加州大学伯克利分校

李秀宇
xiuyu@berkeley.edu 加州大学伯克利分校

沉盛
sheng.s@berkeley.edu 加州大学伯克利分校

Michael W. Mahoney
mmahoney@stat.berkeley.edu ICSI,劳伦斯伯克利国家实验室,加州大学伯克利分校

Kurt Keutzer
keutzer@berkeley.edu 加州大学伯克利分校

抽象的

生成式大型语言模型 (LLM) 已在广泛的任务中展示了显着的结果。然而,由于其前所未有的资源需求,部署这些模型进行推理一直是一项重大挑战。这迫使现有的部署框架使用多 GPU 推理管道（这通常很复杂且成本高昂）,或者使用更小且性能较差的模型。在这项工作中,我们证明了 LLM 生成推理的主要瓶颈是内存带宽,而不是计算,特别是对于单批推理。虽然量化通过降低精度表示权重而成为一种有前景的解决方案,但之前的努力往往会导致性能显着下降。为了解决这个问题,我们引入了 SqueezeLLM,一种训练后量化框架,不仅可以无损压缩至高达 3 位的超低精度,而且可以在相同的内存约束下实现更高的量化性能。我们的框架融合了两个新颖的想法：(i)基于灵敏度的非均匀量化,它基于二阶信息搜索最佳的位精度分配； (ii) 密集和稀疏分解,以有效的稀疏格式存储异常值和敏感权重值。

当应用于 LLaMA 模型时,与具有相同内存要求的最先进方法相比,我们的 3 位量化可将与 FP16 基线的困惑度差距显着减少高达 2.1 倍。此外,当部署在 A6000 GPU 上时,我们的量化模型与基线相比可实现高达 2.3 倍的加速。我们的代码是开源的并且可以在线获取1。

1 简介

大型语言模型 (LLM) 的最新进展拥有多达数千亿个参数,并在海量文本语料库上进行训练,展示了其在各个领域卓越的问题解决能力 [4, 10, 16, 30, 51, 53, 57、62、64、84、84]。然而,由于资源要求很高,部署这些模型进行推理一直是一项重大挑战。例如,LLaMA-65B [63] 模型需要至少 130GB RAM 才能在 FP16 中部署,这超出了当前的 GPU 容量。即使存储如此大尺寸的模型也变得昂贵且复杂。

正如将在第 2 节中讨论的那样。如图3所示,生成任务的LLM推理的主要性能瓶颈是内存带宽而不是计算。这意味着我们加载和存储参数的速度成为内存限制问题的主要延迟瓶颈,而不是算术计算。

然而,与计算机的进步相比,最近内存带宽技术的进步明显缓慢,导致了称为内存墙的现象[50]。

因此,研究人员将注意力转向探索算法方法来克服这一挑战。

一种有前途的方法是量化,其中模型参数以较低的精度存储,而不是用于训练的典型 16 或 32 位精度。例如,已经证明LLM模型可以以8位精度存储而不会导致性能下降[75],其中8位量化不仅将存储需求减少一半,而且还有可能改善推理延迟和吞吐量。因此,人们对将模型量化到更低的精度产生了浓厚的研究兴趣。一种开创性的方法是 GPTQ [19],它使用免训练量化技术,为具有超过数百亿参数的大型 LLM 模型实现近乎无损的 4 位量化。然而,实现高量化性能仍然具有挑战性,特别是对于较低位精度和相对较小的模型（例如,< 50B 参数）,例如最近的 LLaMA [64] 或其指令跟踪变体 [8,22,61]。

在本文中,我们对低位精度量化进行了广泛的研究,并确定了现有方法的局限性。基于这些见解,我们提出了一种新颖的解决方案,即使在精度低至 3 位的情况下,也能实现无损压缩并提高相同大小模型的量化性能。

贡献。我们首先展示性能建模结果,证明内存（而不是计算）是生成任务的 LLM 推理的主要瓶颈。

基于这一见解,我们引入了 SqueezeLLM,这是一种训练后量化框架,它结合了一种新颖的基于灵敏度的非均匀量化技术和密集和稀疏分解位置方法。这些技术可实现超低位精度量化,同时保持有竞争力的模型性能,显着减小模型大小和推理时间成本。更详细地说,我们的主要贡献可以总结如下：

~~*同等贡献 †通讯~~
作者1https://github.com/SqueezeAILab/SqueezeLLM

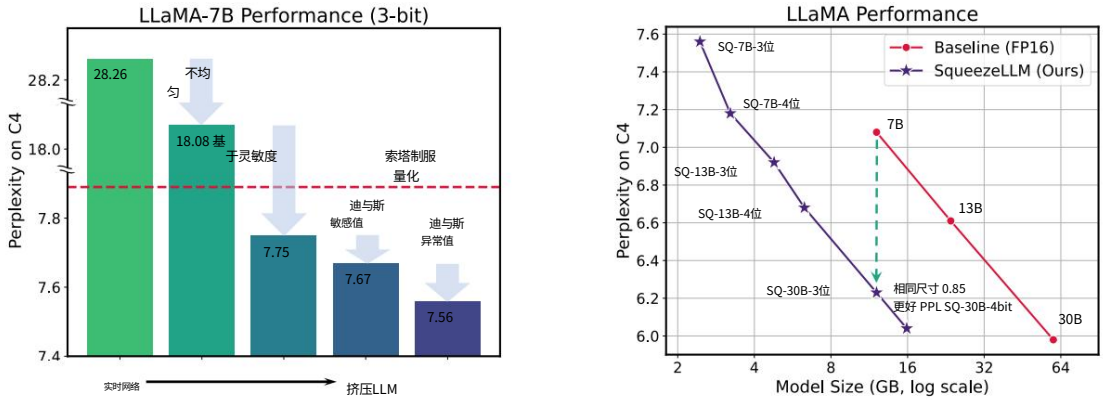

图 1：（左）我们引入了 SqueezeLLM，这是一种 LLM 训练后量化方案，其性能优于现有的最先进方法 [19, 45]。 SqueezeLLM 包含两种关键方法：(i) 基于灵敏度的非均匀量化（第 4.1 节），其中量化仓的分配更接近敏感值，以及 (ii) 密集和稀疏分解（第 4.2 节），其中将敏感值和异常值保留为全精度稀疏格式，有效限制量化范围并增强量化性能。当应用于具有 3 位量化的 LLaMA-7B 时，我们的方法在 C4 基准上优于最先进的方法 [19, 45]，其困惑度裕度超过 0.3。

（右）通过将我们的方法应用于不同大小的 LLaMA [64] 模型，我们可以实现困惑度和模型大小之间的改进权衡。
值得注意的是，我们的 3 位量化 LLaMA-30B 模型在 C4 基准上的困惑度显着提高了 0.85，同时保持相同的模型大小，优于 7B 模型。

· MemoryWall：我们对生成式LLM 推理进行了全面的性能基准测试，以分析不同组件对整体运行时的影响，发现内存带宽（而不是计算）是主要瓶颈。

例如，我们发现，通过简单地降低权重的位精度，同时保持激活并以 FP16 精度进行计算，我们可以在运行时实现与所使用的位精度成正比的线性加速（参见第 3 节和图 2））。 · 基于灵敏度的非均匀量化：我们证明，先前工作中普遍采用的均匀量化对于LLM 推理来说不是最佳的，原因有两个。首先，LLM 中不同层的权重分布表现出明显的不均匀模式，如图 3 所示。其次，先前工作中的推理计算并没有受益于均匀量化，因为算法是以 FP16 精度执行的，并且量化仅用于减少内存需求。

为了解决这两个限制，我们提出了一种新颖的基于灵敏度的非均匀量化方法，为 LLM 实现更优化的量化方案。我们的方法显着改善了 LLaMA-7B 模型在 3 位精度下的困惑度，使 C4 数据集上的困惑度从均匀量化的 28.26 提高到 7.75（参见第 4.1 节）。 · 密集和稀疏量化：我们观察到许多LLM 中的权重矩阵包含显着的异常值，使得低位精度量化极具挑战性。这些异常值也会影响我们的非均匀量化方案，因为它们使比特分配偏向这些极值。为了解决这个问题

sue，我们提出了一个简单的解决方案，将模型权重分解为密集组件和稀疏组件，其中后者从原始权重中提取异常值。通过分离异常值，密集部分表现出高达 10 倍的更紧凑范围，从而提高量化精度。

稀疏部分使用高效稀疏以全精度存储

压缩稀疏行（CSR）等存储方法。这种方法引入的开销最小，因为我们可以利用稀疏部分的高效稀疏内核，并与密集部分并行计算。通过仅提取 0.45% 的权重值作为稀疏分量，我们进一步将 C4 数据集上的 LLaMA-7B 模型的困惑度从 7.75 提高到 7.58（参见第 4.2 节）。 · 性能评估：我们使用 C4 和 WikiText2 基准测试在 LLaMA-7B、13B 和 30B 上的语言建模任务上广泛测试 SqueezeLLM，我们发现 SqueezeLLM 在不同位精度上始终大幅优于现有量化方法（参见表 1）。 1和图5）。 · 指令跟随模型的量化：我们还通过将其应用于 Vicuna-7B 和13B 模型来展示SqueezeLLM 在量化指令跟随模型方面的潜力[8]。我们使用两种评估方法。首先，我们评估 MMLU 数据集 [29] 的生
成质量，这是一个评估模型知识和解决问题能力的多任务基准。此外，按照 Vicuna [8] 中引入的评估方法，我们采用 GPT-4 与 FP16 基线相比，对量化模型的生成质量进行排名。在这两项评估中，SqueezeLLM 始终优于当前最先进的方法，包括 GPTQ 和 AWQ。值得注意的是，我们的 4 位量化模型在两项评估中都实现了与基线相当的性能（参见表 2 和图 6）。 · 模型部署和分析：我们在 A6000 GPU 上部署的模型不仅展示了改进的量化性能，而且还展示了延迟的显着改善。对于 LLaMA-7B 和 13B，我们观察到与基线 FP16 推理相比加速高达 2.3 倍。此外，与 GPTQ 相比，我们的方法实现了高达 4 倍的延迟，展示了我们的方法在量化性能和推理效率方面的有效性（见表 3）。

2 相关工作

2.1 高效 Transformer 推理

人们提出了各种方法来减少 Transformer 推理的延迟和内存占用。虽然一些方法专注于提高解码过程的效率 [6,26,37,54,70],但另一条研究重点是提高 Transformer 架构本身的效率。这可以大致分为高效架构设计 [32, 38, 42, 60, 67, 73],修剪 [18, 20, 40, 41, 48, 52, 65, 79],神经架构搜索 [7, 58, 59,66,74,78]和量化[35,55,81,82]。其中,后者已被证明可以在减少内存占用和改善延迟和吞吐量方面产生非常有希望的结果,我们将在下面简要讨论。

2.2 基于 Transformer 的模型的量化 量化方法可以根据两个因素大致分类。第一个因素是是否需要再培训[23]。

量化感知训练 (QAT) 需要重新训练模型以调整其权重,以帮助在量化后恢复准确性 [2, 35, 55, 82, 85, 86],而训练后量化 (PTQ) 无需任何重新训练即可执行量化 [5、44、49、56、87]。虽然 QAT 通常可以带来更高的准确性,但由于重新训练模型的成本高昂和/或无法访问训练数据和基础设施,对于法学硕士来说通常不可行。因此,大多数关于 LLM 量化的工作都集中在 PTQ 方法上 [12,19,45,75,80]。我们的工作还侧重于 PTQ 方法。

对量化方法进行分类的另一个重要因素是均匀量化与非均匀量化[23]。在均匀量化[13,19,31,35,45,46,55,82]中,权重范围被均匀分成大小相等的2个容器,其中 是位精度。统一形式量化越来越受欢迎,因为它可以通过以量化精度而不是全精度执行算术来实现更快的计算。然而,最近的硬件趋势表明,计算速度的提高并不一定会转化为端到端延迟或吞吐量的改善[24],特别是在生成 LLM 推理（第 3 节)等内存受限任务中。

此外,当权重分布不均匀时,均匀量化可能不是最佳的,就像一般神经网络和 LLM 的情况一样（图 3）。

因此,在这项工作中,我们关注非均匀量化,它以非均匀方式分配量化仓,没有任何限制。高级思想是将更多的箱分配给权重集中的区域,从而在给定的位精度下以更小的量化误差更精确地表示权重。虽然非均匀量化不支持用于计算加速的定点或整数算术,但这个缺点对于我们关注的内存限制问题并不重要,其中主要瓶颈在于内存带宽而不是计算操作。在非均匀量化方法[11,33,81]中,与我们最相似的工作是GOBO[81],它引入了一种用于非均匀量化的基于k均值聚类的查找表方法。与 GOBO 相比,我们的工作引入了两种新颖的方法:(i) 灵敏度感知和 (ii) 密集和稀疏量化方法,这些方法在基于 k 均值的非均匀量化框架内产生了实质性改进。

2.3 异常值感知量化 低位 Transformer 量化的挑

战之一是异常值的存在 [39],它会不必要地增加量化范围。为了解决这个问题,人们研究了异常值感知量化方法[3,12,68,69]。值得注意的是,[12]建议将离群值激活保留为浮点表示,而[69]建议将离群值因子迁移到后续层而不改变功能。所有这些方法都集中于处理激活中的异常值。在我们的工作中,这不是一个问题,因为所有激活都保持为浮点数。相反,我们的密集和稀疏量化方案是解决低位 LLM 量化权重值异常值的通用方法。

与我们的工作同时,SpQR [13] 最近的研究也探索了一种在量化背景下提取异常值的方法。然而,SpQR 采用了基于最佳脑外科医生 (OBS) 框架 [27, 28] 的不同灵敏度度量,其中权重以不干扰每层输出激活的方式进行量化。另一方面,我们的方法基于最佳脑损伤（OBD)框架[14,15,43,55,76],其中权重被量化以保持模型的最终输出而不受到扰动。虽然这两种方法都显示出前景,但我们观察到 OBD 方法可以产生更好的量化性能,因为它是量化后端到端性能下降的直接测量（第 8.2.4 节）。

更重要的是,SpQR 需要可能引入高开销和复杂性的方法来实现无损量化。相比之下,SqueezeLLM 通过两种关键方式解决这个问题。首先,SqueezeLLM 不包含分组。我们的密集和稀疏方案提供了一种直接的解决方案,可以防止异常值和敏感值对量化性能产生负面影响,从而消除了作为间接和次优解决方案进行分组的需要（第 8.2.3 节）。相反,SpQR 需要细粒度分组（例如,组大小 16）,这会增加模型大小,并由于需要双层量化方案而使量化管道复杂化。其次,SqueezeLLM 中基于灵敏度的非均匀量化允许更小的（例如 0.05%)甚至零稀疏级别来实现精确量化。这对于减小模型大小以及提高推理速度至关重要,因为较高的稀疏度（例如 1%)会降低推理延迟（第 5.4 节）。总而言之,通过避免分组和利用较小的稀疏性级别,SqueezeLLM 实现了准确和快速的量化,同时将平均位精度降低至 3 位,同时采用更简单的量化管道和实现。

另一项并行工作是 AWQ [45],它通过引入缩放因子来减少一些重要权重的量化误差,从而改进了 LLM 的仅权重量化方案。

然而,他们的方法也基于 OBS 框架,其中灵敏度由激活的幅度决定。在秒。如图 5 所示,我们证明我们的方法在各种模型和应用场景的量化性能方面始终优于 AWQ。

3 记忆墙

在目标硬件平台上分析神经网络推理时,考虑网络是否会被使用是至关重要的。

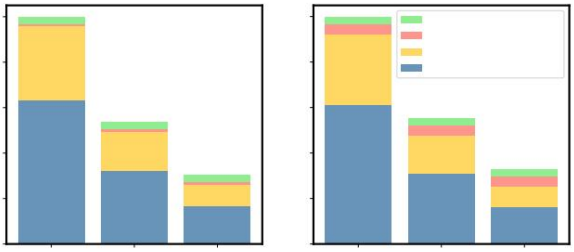

图 2:当降低序列长度为 128（左)和 2048（右)的权重的位精度时，LLaMA-7B 网络的归一化运行时间。结果是使用 A5000 GPU 的基于车顶线的性能模型获得的。对于短代和长代,主要瓶颈在于全连接层。

此外,仅降低权重（而不是激活)的精度就足以显着降低延迟。

主要受计算限制或内存限制。推理中的计算限制受到硬件峰值计算吞吐量的限制,而内存限制推理则受到数据从内存输入处理核心的速率的瓶颈。

用于评估内核是否受计算限制或受内存限制的典型指标是算术强度[71],它是该内核中计算操作与内存操作的比率。高算术强度意味着内存操作很少但计算操作很多。这会导致计算限制问题,该问题可以受益于更快的计算算法和具有更高峰值计算吞吐量的硬件。相反,低算术强度意味着相对于内存操作的数量而言,计算操作非常少。

在这种情况下,问题是内存限制,可以通过减少内存流量来实现加速,而不一定通过减少计算来实现。这是因为,在低算术强度工作负载中,硬件中的计算单元通常未得到充分利用,因为处理器处于空闲状态,等待从内存接收数据。

我们发现,LLM 推理的生成工作负载（即解码工作负载)相对于其他神经网络工作负载 2 表现出极低的算术强度。例如,生成推理中矩阵乘法中计算操作与内存操作的比率仅为2,这意味着每2次乘法/加法,我们需要执行一次内存操作[36]。这是因为生成推理几乎完全由矩阵向量运算组成,这限制了可实现的数据重用,因为每个权重矩阵负载仅用于处理单个向量。

这与编码工作负载相反,在编码工作负载中,整个输入序列是并行处理的,并且权重矩阵负载可以在序列中不同标记的多个激活向量之间分摊。

这个 2 的算术强度需要与典型 GPU 上的内存操作与计算操作的比率进行对比,后者要高出几个数量级。例如,A5000

2准确地说,我们将讨论限制为仅使用解码器模型的单批推理,其中算术涉及矩阵向量运算。对于大批量推理或不同的模型架构,计算可能变得很重要。

GPU 的峰值计算吞吐量比 DRAM 带宽高 290 倍3。这意味着内存带宽是生成 LLM 推理中的瓶颈,而不是计算。计算和内存带宽之间的差异,加上深度学习不断增长的内存需求,被称为内存墙问题[24]。由于 LLM 推理受内存带宽限制,因此增加更多计算开销不会损害性能。特别是,这意味着我们可以自由地探索计算量更大但内存带宽更高效的策略来表示数据[75]。

为了进一步说明生成式 LLM 中的内存墙问题,我们使用了一种简单的基于屋顶线的性能建模方法 [36] 来研究 LLaMA-7B 在 A5000 GPU 上的运行时。结果如图 2 所示,其中我们绘制了用于权重值的不同位精度的运行时间。在这里,我们假设所有情况下的激活和计算都保持在 FP16。尽管如此,我们可以清楚地看到,随着位精度的降低,运行时间呈线性下降。这个结果是预期的,因为主要瓶颈是内存而不是计算。

此外,如图 2 所示,很明显,全连接层是生成推理的主要瓶颈,超过了激活到激活矩阵乘法或非线性运算的影响。值得注意的是,LLaMA 中的大部分内存操作都与在全连接层中加载权重矩阵相关。具体来说,在 LLaMA-7B 的情况下,对于序列长度为 128 的权重矩阵占总内存操作的 99.7%。即使对于较长的序列长度（例如 2048）,该值仍然显着,为 95.8%。

综上所述,在生成 LLM 推理中,将权重矩阵加载到内存中是主要瓶颈,而 FP16 域中的反量化和计算成本相对较小。因此,通过仅将权重量化为较低的精度,同时使激活保持完全精度,除了减小模型大小之外,我们还可以获得显着的加速。这一观察结果也与 GPTQ [19] 和 Zero Quant [75] 的研究结果一致,后者得出的结论是,即使量化权重在浮点域内进行反量化和计算,但加载权重时内存流量的减少仍然会带来显着的延迟改善。鉴于这种认识,适当的策略是最小化内存大小,即使它可能会增加算术运算的开销。

4 方法论

在这里,我们讨论 SqueezeLLM 中包含的两个主要思想:(i) 基于灵敏度的非均匀量化（第 4.1 节)和 (ii) 密集和稀疏量化（第 4.2 节）。然后我们展示如何在硬件中有效地实现这种方法（第 4.3 节）。

4.1 基于灵敏度的非均匀量化

在图 3（上)中,我们绘制了 LLaMA-7B 中参数的示例权重分布。该分布清楚地表明了一种不均匀的模式,其中大多数权重值都以零为中心,同时也有一些异常值。因此,主要

3A5000 GPU 的峰值计算吞吐量为每秒 222 TeraFLOP,峰值内存带宽为每秒 768 GB。

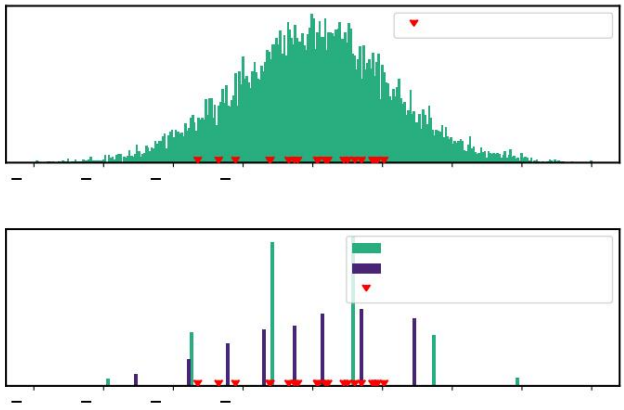

图 3：（上）非量化的原始权重分布
LLaMA-7B 型号。权重值来自单个输出通道
在最后的向下投影层中，以及前 20 个最敏感的值，
由方程式确定4、用红色标记。 （下）重量
使用两种不同聚类进行 3 位量化后的分布
方法：（非加权）k 均值聚类和基于灵敏度
加权 k 均值聚类。请注意，在后一种情况下，量化
值更多地聚集在敏感值周围。

量化的任务是找到一种最佳方式来分配不同的
以保留模型的方式量化值（例如，3 位为 8）
性能和能力。正如我们之前讨论的，广泛使用的
方法包括最近的 LLM 量化工作 [13, 19, 45]
是均匀量化，其中权重范围被均匀划分
分成多个 bin，每个 bin 由一个整数表示。
这种 LLM 量化方法有两个主要问题。首先，均匀分布量化值不是最优的

因为神经网络中的权重分布通常是不均匀的，如图 3 所示。其次，均匀量化的主要优点是
快速有效，但精度降低

计算，这不会导致端到端延迟的改善
在内存限制的 LLM 推理中。鉴于上述限制，我们选择了非均匀量化[9,25,34,72,83]，

这使得代表的分配更加灵活
没有任何约束的值。
寻找最优的非均匀分布很简单
量化配置转化为解决 k 均值问题。给定权重分布，目标是确定质心

最好地代表这些值（例如，对于 3 位量化，=8）。
非均匀量化的优化问题可以是
制定如下：

$$(\quad)^{*} = \arg\min \; // \; - \; //^{2}_{2}, \qquad (1)$$

其中 表示权重和指定权重值（即 [( ) for 是对应的 quan
ε]），由不同值表示 { 1, · · ·
, }。可以看出，最优解
( ) 可以通过应用一维k-means聚类算法来获得，该算法将参数聚类成簇

并将每个簇的质心指定为 s。虽然这种方法
已经优于均匀量化，我们提出了一种改进的
方法通过结合基于灵敏度的 k 均值聚类。

基于灵敏度的 K 均值聚类。量化模型的目标是以低位精度表示模型权重，同时确保模型
输出中的扰动最小[14]。

虽然量化会在每一层中引入误差或扰动，
我们需要最小化关于
最终的损失项，而不是关注各个层，因为它提供了量化后端到端性能下降的更直接的衡
量标准[43]。为了实现这一点，我们需要放置

k 均值质心更接近更敏感的值

相对于最终损失，而不是处理所有权重值
同样，如方程式所示。 1. 要确定哪些值更敏感，
我们可以进行泰勒级数展开来分析模型如何
输出随参数扰动而变化：

$$L(\quad) \simeq L(\quad) - \quad(-) + \quad \frac{1}{2}(\text{-}) \quad (\text{-})$$

$$\simeq L(\quad) + (\frac{1}{2}) \quad (\text{-}), \qquad (2)$$

其中 是梯度并且 = E[$\frac{^{2}}{^{2}}$ L ( )] 是 处损失的二阶导数（即
Hessian）。假设模型有
收敛到局部最小值，梯度可以近似
为零，这给了我们以下计算公式
量化后模型会受到很大的扰动：

$$(\quad)^{*} = \arg\min(-\quad) \quad (\text{-} \quad)。 \qquad (3)$$

在新的优化目标中，与式（1)相比， 1、量化后各个权重的扰动，即-进行加权
，
通过二阶导数引入的比例因子，
。这凸显了最小化扰动的重要性
具有较大 Hessian 值的权重，因为它们对最终输出的整体扰动有更大的影响。换句话说，

二阶导数作为重要性的衡量标准
每个权重值。
虽然可以使用第二个反向传播来计算 Hessian 信息 [77]，但这种方法对于法学硕
士来说成本高昂，因为
增加了内存需求。因此，我们使用近似值
到基于所谓的 1 采样梯度或 Fisher 的 Hessian 矩阵
信息矩阵。特别地，Fisher信息可以是
根据样本数据集计算如下：

$$\simeq F = \frac{1}{| \; |} \Sigma_{\varepsilon} \quad 。 \qquad (4)$$

这只需要计算一组样本的梯度，即
可以使用现有框架进行有效计算。使
方程中的优化目标3. 更可行的是，我们进一步将Fisher信息矩阵近似为对角矩阵：

假设交叉权重相互作用可以忽略不计。这
我们的目标简化如下：

$$(\quad)^{*} \simeq \arg\min(-\quad) \quad \mathrm{diag}(F)(-\quad)$$

$$= \arg\min \Sigma_{=1} \; F\text{-}(\quad)^{2}。 \qquad (5)$$

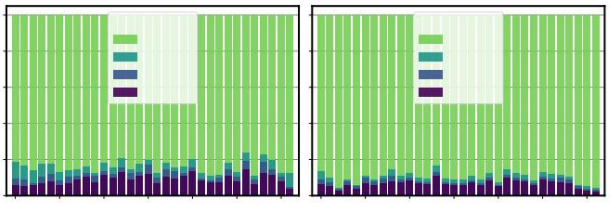

图 4:LLaMA-7B 模型中不同层的 MHA 模块中的输出投影和 FFN 模块中的向下投影的绝对权重值的分布（按最大值归一化）。请注意,分布在所有层中都表现出异常值模式,其中 99% 的值聚集在整个范围的 10% 内。

使用方程式的一个重要结果。 5 是加权 k 均值聚类设置,其中质心将被拉近这些敏感权重值。在图 3 中,我们提供了 LLaMA-7B 的单个输出通道的示例。上图显示了量化前的原始权重分布,其中我们还表示 x 轴上的前 20 个敏感值（根据 Fisher 信息计算）。在下图中,将均匀量化（绿色条)分配的量化值与基于灵敏度的 k 均值方法（紫色条)分配的量化值进行了比较。我们可以清楚地看到,我们基于灵敏度的方法通过将质心放置在敏感值附近,实现了更好的权衡,有效地最小化了量化误差。

为了确保这种方法的有效性,我们使用各种量化方法在 C4 数据集上使用 LLaMA-7B 模型进行了一系列广泛的消融研究。 FP16 中的基线模型的困惑度达到 7.08。通过使用舍入到最近值 (RTN) 的 3 位均匀量化,可得到 28.26 的困惑度值。在不考虑灵敏度的情况下采用非均匀量化可将这种困惑度提高到 18.08。值得注意的是,基于灵敏度的 k 均值方法取得了显着的改进,将困惑度降低至 7.75,如图 1 所示。有关这些消融实验的更多详细信息在第 2 节中提供。 8.2.

4.2 密集和稀疏量化 将 LLM 中的权重量化为低位精度时的另一个挑战涉及离群值 [3,12,68,69]。在图 4 中,我们绘制了 LLaMA 7B 模型中不同层的 MHA 模块中的输出投影和 FFN 模块中的向下投影的权重值分布（按最大值归一化）。该图表明,大约 99.9% 的权重值集中在小于或约为整个权重分布 10% 的狭窄范围内。简单地量化如此大范围的权重会显着降低性能,尤其是在 3 位等低精度情况下。然而,图 4 中的观察结果也暗示着机会。只需去除少量离群值（例如0.1%）,权重值的范围就可以缩小10倍,从而显着提高量化分辨率。这将导致基于灵敏度的 k 均值质心更多地关注敏感值而不是一些异常值。

受此观察的启发,我们引入了一种方法,通过将权重矩阵执行非常简单但有效的分解为稠密矩阵 ( ) 和稀疏矩阵 ( ) 来过滤掉权重矩阵中的异常值。稀疏部分是通过计算给定层中的异常值并将其从权重矩阵中取出来计算的。

余数是一个密集矩阵,由于其值范围显着减小（在某些情况下超过 10×）,可以更有效地量化：

$$= + \text{st} = [\text{最小值} \leqslant \leqslant \text{最大值}]$$
$$\text{且} = [< \text{最小值或} > \text{最大值}], \quad (6)$$

其中 min,根据分布 $_{\text{最大限度}}$ 是定义异常值的阈值,可以是的百分位数计算。

重要的是,请注意,这种分解的开销是最小的,因为异常值的数量很少。即使在最激进的量化实验中,我们也没有发现有必要使用超过 0.5% 的稀疏度。因此,可以使用压缩稀疏行（CSR)格式等方法有效地存储稀疏矩阵。通过密集和稀疏分解,推理也很简单。矩阵运算很容易并行化为= +,这将使我们能够有效地重叠两个内核。特别是,密集部分（·)可以使用非均匀量化方案有效地计算,而稀疏部分（·)可以从稀疏库中受益[1]。

,

基于灵敏度的稀疏矩阵。除了提取稀疏矩阵之外,我们发现在权重矩阵中提取少量高度敏感的值也很有帮助,以确保这些值准确表示而没有任何错误。这些值可以根据第 2 节中讨论的 Fisher 信息轻松计算。 4.1.这有两个好处。首先,通过以 FP16 精度保留这些敏感值,我们可以最大限度地减少它们对模型最终输出扰动的影响。其次,我们防止方程的质心。 5 避免了向敏感值倾斜,从而也改善了不太敏感的权重值的量化误差。我们观察到,跨层仅提取这些敏感值的 0.05% 就可以显着提高量化性能（第 8.2 节）。

在模型性能方面,当应用于 C4 上 LLaMA-7B 的 3 位量化时,仅过滤掉 0.05% 的敏感值即可将困惑度从 7.75 显着提高到 7.67。通过进一步去除 0.4% 的异常值,从而使整体稀疏度达到 0.45%,我们将困惑度从 7.67 进一步降低到 7.56（图 1）。据我们所知,这是 LLaMA 模型 3 位量化的最佳报告结果。

4.3 密集和稀疏内核实现 一个自然要考虑的问题是非均匀和密集和稀疏分解对延迟的影响。我们发现有效地实现这些方法非常简单。

我们首先实现了 3 位和 4 位基于密集查找表的内核,以便在压缩矩阵和未压缩激活向量之间执行矩阵向量乘法。这些内核经过优化,可以以压缩格式加载权重矩阵,并对其进行逐个反量化,以最大限度地减少内存带宽利用率。压缩矩阵在骰子中存储 3 位或 4 位,对应于包含 FP16 的查找表中的条目

与从非均匀量化获得的箱相关的值。使用查找表进行反量化后,所有算术都以全精度执行。

为了有效地处理我们的密集和稀疏表示,我们还开发了用于密集和稀疏矩阵向量乘法的 CUDA 内核。我们使用自定义稀疏内核来加载 CSR 格式的矩阵以及密集激活

向量,受到[17]中的实现的启发。此外,我们实现了一个混合内核来处理权重矩阵中存在偏斜非零分布的情况,其中权重矩阵中的少量通道具有大量非零(参见附录 8.1 中的图 7)。我们使用密集矩阵向量运算与稀疏矩阵向量运算并行处理这一少量行,以减少稀疏矩阵密集向量运算的延迟。在我们的实验中,我们隔离了每个权重矩阵中非零数最多的 10 行进行并行处理。密集非均匀内核和混合稀疏 CSR 内核在一次调用中启动,以避免对这些单独操作的输出向量求和所产生的开销。

部署和分析。为了减少静态内存消耗,我们采用压缩内存格式来存储量化模型。此外,我们将自定义内核集成到 PyTorch 中,以直接在压缩格式上启用端到端推理。为了评估 SqueezeLLM 的性能,我们使用不同稀疏级别的 3 位和 4 位量化测量在单个 A6000 机器上生成 128 和 1024 个令牌的延迟和峰值内存使用情况,并与完整的 16 位进行比较精确推理和 GPTQ。由于[19]中的 GPTQ(特别是分组版本)的官方实现不可用,我们基于最活跃的开源代码库 4实现了用于单批次推理的优化内核。

5 评价

5.1 实验设置模型和数据集。我们使

用 LLaMA [63] 和 Vicuna (v1.1) [8] 模型对 SqueezeLLM 在各种任务上进行了全面评估。首先,在语言建模评估中,我们将 SqueezeLLM 应用于 LLaMA-7B、13B 和 30B 模型,并在块大小为 2048 的 C4 [51] 和 WikiText2 [47] 数据集上测量量化模型的困惑度。还使用指令调整的 Vicuna-7B 和 13B 模型通过零样本 MMLU [29] 评估特定领域的知识和解决问题的能力。我们使用语言模型评估工具对所有任务进行零样本评估 [21]。最后,我们按照[8]中提出的方法评估指令跟随能力。为此,我们生成了 80 个示例问题的答案,并将它们与使用 GPT-4 分数的 FP16 对应问题生成的答案进行比较。为了最大限度地减少排序效应,我们在两个订单中都提供了 GPT-4 的答案,总共产生了 160 个查询。

基线方法。我们将 SqueezeLLM 与法学硕士的几种 PTQ 方法进行比较,包括舍入到最近(RTN)方法以及最先进的 GPTQ [19] 和 AWQ [45]。

为了确保公平的性能比较,我们在所有实验中使用 GPTQ 和激活排序,这解决了否则会发生的性能显着下降的问题。

量化细节。对于 SqueezeLLM,我们采用逐通道量化,其中每个输出通道都分配有一个单独的查找表。我们使用 3 种不同的稀疏级别:0%(仅密集)、排除敏感值的 0.05% 和另外排除异常值的 0.45%(基于其大小)。为了测量灵敏度,我们使用 LLaMA 模型的 C4 训练集中的 100 个随机样本,以及 Vicuna 模型的 Vicuna 训练集。虽然分组也可以与我们的方法结合起来,但我们发现与提取稀疏的敏感值和异常值相比,它不是最佳的(参见附录 8.2.3)。

5.2 主要结果

表 1 显示了 LLaMA-7B、13B 和 30B 的量化结果,以及与舍入到最近值 (RTN) 以及最先进的 PTQ 方法(包括 GPTQ [19] 和 AWQ [45])的比较。这些模型根据其平均位宽(即模型大小)进行分组,以便更好地比较大小与复杂度的权衡。图 5 还专门针对使用 3 位量化的 C4 数据集说明了模型大小和复杂度之间的权衡。下面我们以 LLaMA-7B 作为主要示例来讨论仅密集量化和密集稀疏量化的影响,并随后讨论这些趋势如何扩展到更大的模型。

仅密集量化。在第一组选项卡中。如图 1 (a) 所示,我们比较了稀疏度为 0% 的纯密集 SqueezeLLM 和未对 LLaMA-7B 进行分组的 GPTQ。通过 4 位量化,与 FP16 基线相比,我们的方法表现出最小的退化,C4 和 Wiki Text2 上的困惑度仅下降约 0.1,同时模型大小减少了 3.95 倍。此外,与非分组 GPTQ 相比,我们的方法显示困惑度显着提高,高达 0.22。

对于 3 位量化,两种方法之间的性能差距变得更加明显。 SqueezeLLM 在 C4 和 WikiText2 上的表现明显优于 GPTQ,分别为 1.80 和 1.22 个点,压缩率为 5.29 倍。这与 FP16 基线仅相差 0.67 和 0.55 个点。这些结果证明了有效的

超低比特量化的基于灵敏度的非均匀方法的有效性。

密集和稀疏量化。通过利用密集和稀疏量化方案,我们进一步减少了 FP16 基线和量化模型之间的困惑度差距,如表 2 的第二组和第三组所示。 1(一)。这种改进在 3 位量化的情况下尤其显着,其中仅排除 0.05% 和 0.45% 的值,分别会产生大约 0.1 和 0.2 的困惑度改进。通过密集和稀疏分解,我们实现了几乎无损的压缩,4 位和 3 位的 FP16 基线的困惑度偏差分别小于 0.1 和 0.5。

GPTQ 和 AWQ 都使用分组策略来增强性能,但模型大小方面的开销很小。然而,我们证明,在所有场景中,稀疏度分别为 0.05% 和 0.45% 的 SqueezeLLM 始终优于组规模分别为 256 和 128 的 GPTQ 和 AWQ,同时保持

4https://github.com/qwopqwop200/GPTQ-for-LLaMa

表 1:LLaMA-7B、13B 和 30B 量化的困惑度比较
使用不同的方法（包括舍入到最接近的值）分为 4 位和 3 位
(RTN)、C4 和 WikiText-2 上的 GPTQ [19] 和 AWQ [45]。平均
还包括位宽和压缩率以供比较。
我们比较了 GPTQ、AWQ 和 SqueezeLLM 的性能
基于相似模型尺寸的分组。在第一组中，我们比较
具有非分组 GPTQ 的仅密集 SqueezeLLM。在随后的
组中，我们将具有不同稀疏程度的 SqueezeLLM 与
GPTQ 和 AWQ 具有不同的组大小。请注意，所有 GPTQ 结果
具有激活重新排序功能。直观表示见图 5。

(a) LLaMA-7B

位宽	4位		3位		
方法	平均。位 PPL (↓) （补偿率） C4 Wiki （补偿率） C4 Wiki		平均。位	PPL (↓)	
基线	16	7.08 5.68	16	7.08	5.68
实时网络	4 (4.00×)	7.73 6.29	3 (5.33×)	28.26 25.61	
通用PTQ	4 (4.00×)	7.43 5.94	3 (5.33×)	9.55	7.55
挤压LLM	4.05 (3.95×) 7.21 5.79		3.02 (5.29×) 7.75 6.32		
GPTQ (g256)	4.12 (3.89×) 7.25 5.81		3.12 (5.13×) 8.09 6.43		
预警 (g256)	4.12 (3.89×) 7.29 5.87		3.12 (5.13×) 8.04 6.51		
挤压LLM (0.05%)	4.07 (3.93×) 7.20 5.79		3.05 (5.25×) 7.67 6.20		
GPTQ (g128)	4.24 (3.77×) 7.21 5.78		3.24 (4.93×) 7.89	4.24 (3.77×)	6.27
AWQ (g128)	7.22 5.82	3.24 (4.93×) 7.90	挤压LLM (0.45%)	4.27 (3.75×)	6.44
7.18 5.77	3.24 (4.9) 3×) 7.56 6.13				

(b) LLaMA-13B

位宽	4位		3位		
方法	平均。位 PPL (↓) （补偿率） C4 Wiki （补偿率） C4 Wiki		平均。位	PPL (↓)	
基线	16	6.61 5.09	16	6.61	5.09
实时网络	4 (4.00×)	6.99 5.53	3 (5.33×)	13.24 11.78	
通用PTQ	4 (4.00×)	6.84 5.29	3 (5.33×)	8.22 6.22	
挤压LLM	4.04 (3.96×) 6.71 5.18		3.02 (5.30×) 7.08 5.60		
GPTQ (g256)	4.12 (3.88×) 6.74 5.20		3.12 (5.12×) 7.42 5.68		
预警 (g256)	4.12 (3.88×) 6.72 5.20		3.12 (5.12×) 7.18 5.63		
挤压LLM (0.05%)	4.07 (3.94×) 6.69 5.17		3.04 (5.26×) 7.01 5.51		
GPTQ (g128)	4.25 (3.77×) 6.70 5.17		3.25 (4.92×) 7.12 5.47		
AWQ (g128)	4.25 (3.77×) 6.70 5.21		3.25 (4.92×) 7.08 5.52		
挤压LLM (0.45%)	4.26 (3.76×) 6.68 5.17		3.24 (4.94×) 6.92 5.45		

(c) LLaMA-30B

位宽	4位		3位		
方法	平均。位 PPL (↓) （补偿率） C4 Wiki （补偿率） C4 Wiki		平均。位	PPL (↓)	
基线	16	5.98 4.10	16	5.98	4.10
实时网络	4 (4.00×)	6.33 4.54	3 (5.33×)	28.53 14.89	
通用PTQ	4 (4.00×)	6.20 4.43	3 (5.33×)	7.31 5.76	
挤压LLM	4.03 (3.97×) 6.06 4.22		3.02 (5.31×) 6.37 4.66		
GPTQ (g256)	4.12 (3.88×) 6.08 4.26		3.12 (5.12×) 6.58 4.87		
预警 (g256)	4.12 (3.88×) 6.07 4.24		3.12 (5.12×) 6.45 4.71		
挤压LLM (0.05%)	4.06 (3.94×) 6.05 4.20		3.04 (5.26×) 6.31 4.56		
GPTQ (g128)	4.25 (3.77×) 6.07 4.24		3.25 (4.92×) 6.47 4.83		
AWQ (g128)	4.25 (3.77×) 6.05 4.21		3.25 (4.92×) 6.38 4.63		
挤压LLM (0.45%)	4.25 (3.77×) 6.04 4.18		3.25 (4.92×) 6.23 4.44		

表 2:应用于 Vicuna-7B 和 13B 模型时 PTQ 方法在零样本 MMLU [29] 精度上的比较。

方法	平均。位 Vicuna-7B (↑) Vicuna-13B (↑)		
基线	16	39.1%	41.2%
AWQ (g128)	4.25	38.0%	40.4%
挤压LLM 挤压	4.05	38.8%	39.2%
LLM (0.45%)	4.26	39.4%	41.0%
AWQ (g128)	3.25	36.5%	37.6%
挤压LLM 挤压	3.02	36.0%	37.2%
LLM (0.45%)	3.24	37.7%	39.4%

较小或类似的模型尺寸。这对于
3 位量化，其中 SqueezeLLM 的稀疏度为 0.45%
组规模为 128 时，性能优于 GPTQ 和 AWQ
超过 0.3 个困惑点。

较大模型的结果。在选项卡中。 1 (b, c)，我们观察到
LLaMA-7B 中观察到的趋势扩展到 LLaMA-13B 和 30B mod
埃尔斯。如图 5 所示，SqueezeLLM 在所有模型大小和模型中始终优于最先进的 PTQ 方法。

量化位宽。值得注意的是，即使是仅密集版本
SqueezeLLM 实现了与分组 GPTQ 相当的困惑度
和 AWQ。通过结合稀疏性，我们获得了进一步的困惑
改进，将与 FP16 基线的差距缩小到小于
4 位和 3 位量化的困惑点分别为 0.1 和 0.4。值得注意的是，通过 3 位量化，我们的方法实现了

与 FP16 基线相比，困惑度差距减少了 2.1 倍
与现有方法相比。据我们所知，这
是迄今为止报告的量化这些模型的最佳结果。

附录 8.2 中提供了关于我们的设计选择的进一步消融研究，包括灵敏度指标、稀疏度水平和分组。

5.3 指令跟随的量化
楷模

指令调优已成为改进的常用方法
模型响应用户命令的能力 [8,22,61]。我们
按顺序探索指令跟随模型的量化
展示我们的方法在准确性方面的优势
通过将 SqueezeLLM 应用到 Vicuna-7B 和 13B 来保存 [8]，
并通过以下方法评估性能。

零样本 MMLU 评估。我们首先比较基线
零样本多任务问题解决的量化模型
MMLU 的基准[29]。所有任务的加权准确率
在选项卡中提供。 2 对于基线 Vicuna 7B 和 13B 型号，
以及使用 AWQ [45] 和 SqueezeLLM 量化的模型。作为
我们可以看到，与 AWQ 方法相比，SqueezeLLM 对 Vicuna 7B 和 13B 都实现了更高的准确度，并且还保留了

具有 4 位量化的 FP16 基线模型的准确性。
此外，值得注意的是 4 位量化版本
使用 SqueezeLLM 的 Vicuna-13B 内存占用减少 2 倍
比 FP16 中的 7B 基线模型高出 2%
准确性。因此，SqueezeLLM 不仅提高了性能，而且
还减少了内存需求。

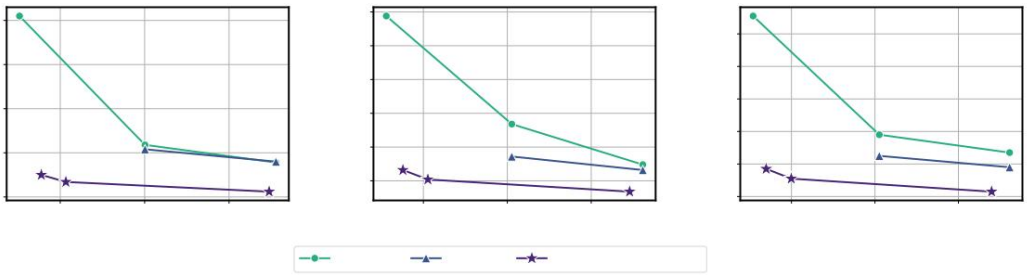

图 5:使用 GPTQ [19]、AWQ [45] 和 SqueezeLLM（我们的)量化为 3 位的 LLaMA-7B、13B 和 30B 模型的困惑度比较,并在 C4 基准上进行评估。 x 轴是相对于 FP16 中模型尺寸的相对模型尺寸。通过调整 GPTQ 和 AWQ 的组大小以及我们方法的稀疏级别来实现模型大小和困惑度之间的不同权衡。我们的量化方法在所有模型大小范围内始终显着优于 GPTQ 和 AWQ,在较低位和较小模型大小方面差距更明显。请注意,所有 GPTQ 结果均经过激活重新排序。

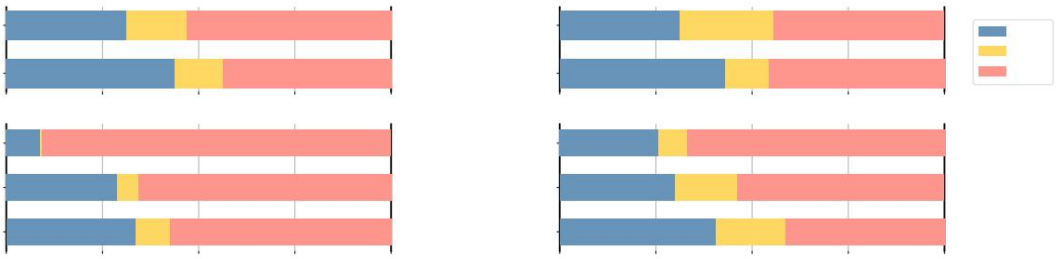

图 6:应用于 Vicuna 模型指令时的 PTQ 方法比较。获胜/平局/失败表示量化网络相对于基线 FP16 Vicuna 网络获胜/平局/失败的次数。该评估是使用 Vicuna [8] 的方法和 v1.1 模型权重进行的。 GPTQ 和 AWQ 结果都是我们根据已发布的代码库复制的。

遵循指令的能力。另一种评估指令跟踪能力的方法是将生成的结果输入 GPT 4 并要求其对结果进行排序,这就是[8]所使用的方法。

值得注意的是,输入 GPT-4 的模型顺序可能会产生偏差,因为据观察,GPT-4 稍微偏向第一个模型 [8]。为了解决这个问题,我们通过交换顺序重复所有比较两次。结果如图6所示。

没有稀疏性的 SqueezeLLM 通过 4 位量化为 Vicuna-7B 和 13B 模型实现了近乎完美的性能（即 50/50 分割）,在相同模型大小的情况下优于 GPTQ。在 3 位量化的情况下,即使不考虑稀疏性,SqueezeLLM 的性能也优于 GPTQ 和组大小为 128 的最先进的 AWQ 方法。此外,事实证明,添加 0.45% 的稀疏度水平对于 Vicuna-13B 模型非常有效,对于 3 位量化也实现了近乎完美的 50/50 分割。

5.4 硬件部署和分析虽然排列分组是限制量化范围的有效方法,但我们的密集和稀疏方案可以通过更简单的内核实现更高的精度。我们在选项卡中显示了 SqueezeLLM 的延迟和峰值 GPU 内存使用情况。生成 128 个令牌时,在 A6000 GPU 上针对不同配置的结果为 3。我们观察到 SqueezeLLM 中基于 LUT 的非均匀方法（第 4 行)与 FP16 基线相比显示出高达 2.3 倍的加速,并且表现出与非分组 GPTQ 的统一量化（第二行)相当的延迟和峰值内存使用量。这表明与基于 LUT 的反量化相关的开销很小,特别是考虑到它带来的相当大的困惑度增益。

此外,当纳入稀疏性时,我们仍然观察到相对于 FP16 基线的延迟增益。如表所示。如图 3 所示,在 FP16 中保留 0.05% 的敏感值只会在不同模型大小上增加大约 20% 的延迟开销,同时与基线相比仍可提供高达 1.9 倍的速度提升。相对于仅密集实现,在 FP16 中保留 0.45% 的参数仅增加 40-45% 的延迟开销,同时与 FP16 基线相比仍能实现 1.7 倍的加速。相反,当考虑排列时,GPTQ 运行时间会严重降低。这种延迟损失是由于排列造成的,这意味着同一通道中的元素需要使用不同的缩放因子（使用组索引访问)进行缩放;高效执行这些分布式内存访问具有挑战性,因为 GPU 严重依赖合并内存访问来优化内存带宽利用。这显示了我们的密集和稀疏量化方法如何实现相对于 GPTQ 更高的精度和更好的性能。

表 3:在硬件上生成 128 个令牌时量化为 3 位的 LLaMA 7B、13B 和 30B 的延迟和内存使用情况的硬件分析
A6000 GPU。第一行是非量化的 FP16 基线,第二行和第三行分别是非分组和分组的 GPTQ。
请注意,所有 GPTQ 结果均具有激活顺序。第四、五、六行显示了具有不同稀疏度的 SqueezeLLM 的性能
级别,第四行表示仅密集 SqueezeLLM。表 2 中提供了序列长度为 1024 的相同分析。 6.

方法	延迟（秒）			峰值内存 (GB)		
	LLaMA-7B	LLaMA-13B	LLaMA-30B	LLaMA-7B	LLaMA-13B	LLaMA-30B
FP16	3.2	5.6	OOM	12.7	24.6	OOM
通用PTQ	1.4	2.1	4.1	2.9	5.3	12.4
GPTQ (g128)	13.7	24.2	61.9	3.0	5.6	12.9
挤压LLM 挤压	1.5	2.4	4.9	2.9	5.4	12.5
LLM (0.05%)	1.8	2.9	5.9	3.0	5.5	12.7
挤压LLM (0.45%)	2.2	3.4	7.2	3.2	5.9	13.8

6 结论和局限性

我们提出了 SqueezeLLM 量化框架
尝试解决与以下相关的内存墙问题
生成式 LLM 推理。我们的硬件分析结果一目了然
证明解码器模型推理的主要瓶颈
是内存带宽而不是计算带宽。根据这一观察，
SqueezeLLM 融合了两个新颖的想法,允许超低
LLM 的精确量化,生成性能的下降可以忽略不计。第一个想法是基于灵敏度的非均匀量化方法;第二
个想法很简单,但是

有效的密集和稀疏分解旨在处理
对量化产生负面影响的异常值。我们在广泛的模型和数据集上对 SqueezeLLM 进行了评估,这些模
型和数据集评估了
量化模型的语言建模、问题解决和指令跟踪能力,我们已经证明了

我们的量化方法可以始终优于以前最先进的方法。此外,我们还展示了

在相同的内存限制下,我们的量化模型
与全精度相比可以实现性能改进
楷模。虽然我们的实证结果主要集中在发电
任务,这项工作中提出的想法并不局限于
解码器架构。然而,我们还没有对我们的框架在仅编码器上的有效性进行全面的评估

或编码器-解码器架构,以及其他神经网络
架构。此外,值得注意的是,我们的硬件性能建模方法依赖于基于仿真的

使用屋顶线模型的方法,这需要简化
关于硬件推理管道的假设。

7 致谢。

作者要感谢 Kartkeya Mangalam，
Nicholas Lee 和 Thanakul Wattanawong 进行了有益的讨论
和头脑风暴。我们感谢 Google 的慷慨支持
Cloud、Google TRC 团队,特别是 Jonathan Caton、Jing Li，
叶佳玉和大卫·帕特森教授。 Keutzer 教授的实验室受到赞助
由英特尔公司、英特尔 VLAB 团队、英特尔 One-API 卓越中心以及 Furiosa、Berkeley Deep 的慷慨
资助
开车,还有BAIR。我们的结论并不一定反映
我们赞助商的立场或政策,没有官方认可
应该可以推断。

参考

[1] https://developer.nvidia.com/cusparse。
[2] 白浩丽、张伟、侯鲁、尚立峰、金晶、蒋欣、刘群、Michael
吕和欧文·金。 BinaryBERT:突破 BERT 量化的极限。 arXiv
预印本 arXiv:2012.15701,2020。
[3] 耶利塞·邦达连科、马库斯·内格尔和蒂门·布兰克沃特。理解和
克服高效 Transformer 量化的挑战。 arXiv 预印本
arXiv:2109.12948,2021。
[4] Tom B Brown、Benjamin Mann、Nick Ryder、Melanie Subbiah、Jared Ka plan、
Prafulla Dhariwal、Arvind Neelakantan、Pranav Shyam、Girish Sastry、
阿曼达·阿斯克尔,等人。语言模型是小样本学习者。 arXiv 预印本
arXiv:2005.14165,2020。
[5] 蔡耀辉、姚哲伟、董振、Amir Gholami、Michael W Mahoney 和
库尔特·科伊策. ZeroQ:一种新颖的零样本量化框架。诉讼中
IEEE/CVF 计算机视觉和模式识别会议的页面
13169–13178, 2020。
[6] Charlie Chen、Sebastian Borgeaud、Geoffrey Irving、Jean-Baptiste Lespiau、Laurent
Sifre 和 John Jumper。加速大型语言模型解码
推测性抽样。 arXiv 预印本 arXiv:2302.01318, 2023。
[7]陈道源,李亚良,邱明辉,王振,李博芳,丁柏林,
邓洪波、黄军、林伟、周敬仁。 AdaBERT:任务自适应
带有可微神经架构搜索的 bert 压缩。 arXiv 预印本
arXiv:2001.04246,2020。
[8] 蒋伟林、李卓涵、林子、盛英、吴张浩、张浩、
郑连民、庄思源、庄永浩、Joseph E. Gonzalez、Ion Stoica、
和 Eric P. Xing。 Vicuna:一款开源聊天机器人,给 gpt-4 留下了 90% 的好评*
chatgpt 质量,2023 年 3 月。
[9] Yoojin Choi、Mostafa El-Khamy 和 Jungwon Lee。走向网络的极限
量化。 arXiv 预印本 arXiv:1612.01543, 2016。
[10] Aakanksha Chowdhery、Sharan Narang、Jacob Devlin、Maarten Bosma、Gaurav
米什拉、亚当·罗伯茨、保罗·巴勒姆、郑亨元、查尔斯·萨顿、塞巴斯蒂安·格尔曼等。 PALM:通过
路径扩展语言建模。 arXiv
预印本 arXiv:2204.02311, 2022。
[11] Insoo Chung、Byeongwook Kim、Yoonjung Choi、Se Jung Kwon、Yongkweon
Jeon、Baeseong Park、Sangha Kim 和 Dongsoo Lee。用于设备上神经机器翻译的极低位
转换器量化。 arXiv 预印本
arXiv:2009.07453,2020。
[12] 蒂姆·德特默斯、迈克·刘易斯、尤尼斯·贝尔卡达和卢克·泽特尔莫耶。 Gpt3。整型8
():大规模 Transformer 的 8 位矩阵乘法。神经进展
信息处理系统。
[13] 蒂姆·戴特默斯、鲁斯兰·斯维尔切夫斯基、瓦吉·埃吉扎里安、丹尼斯·库兹内代列夫、埃利亚斯
Frantar、Saleh Ashkboos、Alexander Borzunov、Torsten Hoefler 和 Dan Alis tarh。
SpQR:近无损 LLM 权重的稀疏量化表示
压缩。 arXiv 预印本 arXiv:2306.03078, 2023。
[14] 董震,姚哲伟,Daiyaan Arfeen,Amir Gholami,Michael W Mahoney,
和库尔特·科伊策。 HAWQ-V2:Hessian 感知迹加权量化
神经网络。 NeurIPS 19 超越一阶优化研讨会
机器学习方法,2019。
[15] 董震,姚哲伟,Amir Gholami,Michael W Mahoney,Kurt Keutzer。
HAWQ:具有混合精度的神经网络的 Hessian 感知量化。
IEEE 国际计算机视觉会议论文集,第
293-302,2019。
[16] 杜楠,黄艳平,戴安德,唐西蒙,Dmitry Lepikhin,
徐元中、Maxim Krikun、周彦琪、Adams Wei Yu、Orhan Firat 等。
GLAM:专家混合的语言模型的有效扩展。国际机器学习会议,第 5547-5569 页。 PMLR,
2022 年。

[17] 格奥尔吉·叶夫图申科。使用 CUDA 的稀疏矩阵向量乘法。 https://medium.com/analytics-vidhya/sparse-matrix-vector-multiplication-with cuda-42d191878e8f,2019。

[18] 安吉拉·范、爱德华·格雷夫和阿尔芒·朱林。通过结构化压差按需减少变压器深度。 arXiv 预印本 arXiv:1909.11556, 2019。

[19] Elias Frantar、Saleh Ashkboos、Torsten Hoefler 和 Dan Alistarh。 GPTQ：生成式预训练 Transformer 的准确训练后量化。 arXiv 预印本 arXiv:2210.17323, 2022。

[20] 特雷弗·盖尔、埃里希·埃尔森和萨拉·胡克。深度神经网络的稀疏状态。 arXiv 预印本 arXiv:1902.09574, 2019。

[21] Leo Gau、Jonathan Tow、Stella Biderman、Sid Black、Anthony DiPofi、Charles Foster、Laurence Golding、Jeffrey Hsu、Kyle McDonell、Niklas Muennighoff、Jason Phang、Laria Reynolds、Eric Tang、Anish Thite、Ben Wang、Kevin Wang和安迪·邹。少量语言模型评估框架，2021 年 9 月。

[22] 耿新阳，Arnav Gudibande，刘浩，Eric Wallace，Pieter Abbeel，Sergey Levine，Dawn Song。 Koala：学术研究的对话模型。博客文章，2023 年 4 月。

[23] Amir Gholami、Sehoon Kim、Zhen Dong、Zhewei Yao、Michael W Mahoney 和 Kurt Keutzer。有效神经网络推理的量化方法的调查。 arXiv 预印本 arXiv:2103.13630, 2021。

[24] Amir Gholami、Zhewei Yao、Sehoon Kim、Michael W Mahoney 和 Kurt Keutzer。人工智能和记忆墙。 https://medium.com/riselab/ai-and-memory wall-2cb4265cb0b8,2021 年。

[25]龚云超，刘刘，杨明，卢博米尔·布尔德夫。使用矢量量化压缩深度卷积网络。 arXiv 预印本 arXiv:1412.6115, 2014。

[26] 顾嘉涛，詹姆斯·布拉德伯里，蔡明熊，维克多·OK·李，理查德·索彻。 非自回归神经机器翻译。 arXiv 预印本 arXiv:1711.02281, 2017年。

[27] 巴巴克·哈西比和大卫·G·斯托克。网络修剪的二阶导数：最佳脑外科医生。神经信息处理系统的进展，第 164-171 页，1993 年。

[28] 巴巴克·哈西比、大卫·G·斯托克和格雷戈里·J·沃尔夫。最佳脑外科医生和一般网络修剪。 IEEE 国际神经网络会议，第 293-299 页。 IEEE，1993。

[29] Dan Hendrycks、Collin Burns、Steven Basart、Andy Zou、Mantas Mazeika、Dawn Song 和 Jacob Steinhardt。测量大规模多任务语言理解。学习表征国际会议 (ICLR) 会议记录，2021 年。

[30] Jordan Hoffmann、Sebastian Borgeaud、Arthur Mensch、Elena Buchatskaya、Trevor Cai、Eliza Rutherford、Diego de Las Casas、Lisa Anne Hendricks、Jo hannes Welbl、Aidan Clark 等。训练计算优化的大型语言模型。 arXiv 预印本 arXiv:2203.15556, 2022。

[31] 黄亚峰，杨焕瑞，董震，Denis Gudovskiy，Tomoyuki Okuno，Yohei Nakata，杜远，张尚航，Kurt Keutzer。输出敏感度感知去量化。 2023 年。

[32] Forrest N Iandola、Albert E Shaw、Ravi Krishna 和 Kurt W Keutzer。 Squeeze BERT：计算机视觉可以教 NLP 哪些关于高效神经网络的知识？ arXiv 预印本 arXiv:2006.11316, 2020。

[33]Yongkweon Jeon、Chungman Lee、Eulrang Cho 和 Yeonju Ro。 BiQ先生：基于最小化重建误差的训练后非均匀量化。 IEEE/CVF 计算机视觉和模式识别会议论文集，第 12329–12338 页，2022 年。

[34] Sangil Jung、Changyong Son、Seohyung Lee、Jinwoo Son、Jae-Joon Han、Youngjun Kwak、Sung Ju Hwang 和 Changkyu Choi。通过任务损失优化量化间隔来学习量化深度网络。 IEEE/CVF 计算机视觉和模式识别会议论文集，第 4350-4359 页，2019 年。

[35] Sehoon Kim、Amir Gholami、Zhewei Yao、Michael W Mahoney 和 Kurt Keutzer。 I-BERT：纯整数 bert 量化。 arXiv 预印本 arXiv:2101.01321, 2021。

[36] Sehoon Kim、Coleman Hooper、Thanakul Wattanawong、Minwoo Kang、Ruohan Yan、Hasan Genc、Grace Dinh、Qijing Huang、Kurt Keutzer、Michael W Mahoney、Sophia Shao 和 Amir Gholami。变压器推理的全栈优化：一项调查。 arXiv 预印本 arXiv:2302.14017, 2023。

[37] Sehoon Kim、Karttikeya Mangalam、Jitendra Malik、Michael W Mahoney、Amir Gholami 和 Kurt Keutzer。大小型变压器解码器。 arXiv 预印本 arXiv:2302.07863, 2023。

[38] 尼基塔·基塔耶夫、卢卡斯·凯泽和安塞姆·列夫斯卡娅。改革者：高效的变压器。在国际学习表征会议上，2019 年。

[39]奥尔加·科瓦列娃、Saurabh Kulshreshtha、安娜·罗杰斯和安娜·拉姆希斯基。 Bert 克星：扰乱变压器的异常维度。 arXiv 预印本 arXiv:2105.06990, 2021。

[40] Eldar Kurtic、Daniel Campos、Tuan Nguyen、Elias Frantar、Mark Kurtz、Benjamin Fineran、Michael Goin 和 Dan Alistarh。最佳的 BERT 外科医生：针对大型语言模型的可扩展且准确的二阶剪枝。 arXiv 预印本 arXiv:2203.07259, 2022。

[41] Woosuk Kwon、Sehoon Kim、Michael W Mahoney、Joseph Hassoun、Kurt Keutzer 和 Amir Gholami。变压器的快速训练后修剪框架。 arXiv 预印本 arXiv:2204.09656, 2022。

[42] 兰振中、陈明达、塞巴斯蒂安·古德曼、凯文·金佩尔、Piyush Sharma 和 Radu Soricut。 ALBERT：一个精简版 BERT，用于语言表示的自我监督学习。 arXiv 预印本 arXiv:1909.11942，2019。

[43]Yann LeCun、John S Denker 和 Sara A Solla。最佳脑损伤。神经信息处理系统的进展，第 598-605 页，1990 年。

[44] 李秀玉，连龙，刘一江，杨欢瑞，董震，康丹尼尔，张尚航，Kurt Keutzer。 Q-扩散：量化扩散模型。 arXiv 预印本 arXiv:2302.04304, 2023。

[45]吉林，唐家明，唐浩天，尚扬，党兴宇，韩松。 Awq：用于 llm 压缩和加速的激活感知权重量化。 2023 年。

[46]刘一江，杨欢瑞，董振，Kurt Keutzer，杜丽，张尚航。 NoisyQuant：视觉变压器的噪声偏差增强训练后激活量化。 IEEE/CVF 计算机视觉和模式识别会议论文集，第 20321–20330 页，2023 年。

[47] 斯蒂芬·梅里蒂、蔡明熊、詹姆斯·布拉德伯里和理查德·索彻。指针哨兵混合模型，2016。

[48] 保罗·米歇尔、奥马尔·利维和格雷厄姆·纽比格。十六个头真的比一个好吗？ arXiv 预印本 arXiv:1905.10650, 2019。

[49] Sangyun Oh、Hyeonuk Sim、Jounghyun Kim 和 Jongeun Lee。非均匀步长量化可实现准确的训练后量化。计算机视觉 - ECCV 2022：第 17 届欧洲会议，以色列特拉维夫，2022 年 10 月 23-27 日，会议记录，第 XI 部分，第 658-673 页。施普林格，2022。

[50] 大卫·A·帕特森。延迟滞后于带宽。 ACM 的通讯， 47（10）：71-75，2004年。

[51] Colin Raffel、Noam Shazeer、Adam Roberts、Katherine Lee、Sharan Narang、Michael Matena、Yanqi Zhou、Wei Li 和 Peter J Liu。使用统一的文本到文本转换器探索迁移学习的局限性。机器学习研究杂志，21（1）：5485–5551，2020。

[52] 维克多·桑、托马斯·沃尔夫和亚历山大·拉什。运动修剪：通过微调实现自适应稀疏。神经信息处理系统的进展，33：20378–20389，2020。

[53] Teven Le Scao、Angela Fan、Christopher Akiki、Ellie Pavlick、Suzana Ili ́、Daniel Hesslow、Roman Castagné、Alexandra Sasha Luccioni、François Yvon、Matthias Gallé 等。 Bloom：176b 参数的开放获取多语言语言模型。 arXiv 预印本 arXiv:2211.05100, 2022。

[54] Tal Schuster、Adam Fisch、Jai Gupta、Mostafa Dehghani、Dara Bahri、Vinh Q Tran、Yi Tay 和 Donald Metzler。自信的自适应语言建模。 arXiv 预印本 arXiv:2207.07061, 2022。

[55] 申盛，董震，叶家宇，马林建，姚哲伟，Amir Gholami，Michael W Mahoney，Kurt Keutzer。 Q-BERT：基于Hessian的bert超低精度量化。 AAAI 人工智能会议记录，第 34 卷，第 8815-8821 页，2020 年。

[56] Gil Shomron、Freddy Gabbay、Samer Kurzum 和 Uri Weiser。训练后稀疏感知量化。神经信息处理系统的进展，34：17737–17748，2021。

[57] Shaden Smith、Mostofa Patwary、Brandon Norick、Patrick LeGresley、Samyam Rajbhandari、Jared Casper、Zhun Liu、Shrimai Prabhumoye、George Zerveas、Vijay Korthikanti 等人。使用 deepspeed 和 megatron 训练大型生成语言模型 megatron-turing nlg 530b。 arXiv 预印本 arXiv:2201.11990, 2022。

[58] 苏大卫，郭乐，陈亮。进化后的变压器。国际机器学习会议，第 5877-5886 页。 PMLR，2019。

[59] David R So、Wojciech Mańke、Hanxiao Liu、Zihang Dai、Noam Shazeer 和 Quoc V Le。入门：寻找用于语言建模的高效转换器。 arXiv 预印本 arXiv:2109.08668, 2021。

[60] 孙志清，于洪坤，宋晓丹，刘仁杰，杨一鸣，周丹尼。 MobileBERT：适用于资源有限设备的紧凑型任务无关 bert。 arXiv 预印本 arXiv:2004.02984, 2020。

[61] Rohan Taori、Ishaan Gulrajani、Tianyi 张、Yann Dubois、Xuechen Li、Car los Guestrin、Percy Liang 和 Tatsunori B. Hashimoto。斯坦福羊驼：遵循指令的美洲驼模型。 https://github.com/tatsu-lab/stanford_alpaca， 2023 年。

[62] Romal Thoppilan、Daniel De Freitas、Jamie Hall、Noam Shazeer、Apoorv Kul shreshtha、Heng-Tze Cheng、Alicia Jin、Taylor Bos、Leslie Baker、Yu Du 等。 Lambda：对话应用程序的语言模型。 arXiv 预印本 arXiv:2201.08239, 2022 年。

[63] 雨果·图夫龙、马蒂厄·科德、马蒂斯·杜兹、弗朗西斯科·马萨、亚历山大·萨布莱罗勒和埃尔韦·杰古。通过注意力训练数据高效的图像转换器和蒸馏。国际机器学习会议，第 10347-10357 页。 PMLR，2021。

[64] Hugo Touvron、Thibaut Lavril、Gautier Izacard、Xavier Martinet、Marie-Anne Lachaux、Timothée Lacroix、Baptiste Rozière、Naman Goyal、Eric Hambro、Faisal Azhar 等。 LLaMA：开放高效的基础语言模型。 arXiv 预印本 arXiv:2302.13971, 2023。

[65] Elena Voita、David Talbot、Fedor Moiseev、Rico Sennrich 和 Ivan Titov。分析多头自注意力：专门的头负责繁重的工作，剩下的

可以修剪。 arXiv 预印本 arXiv:1905.09418, 2019。
[66] 王涵瑞,吴章豪,刘志坚,蔡涵,朱立庚,甘闯,韩松。 HAT:用于高效自然语言处理的硬件感知转换器。 arXiv 预印本 arXiv:2005.14187, 2020。

[67] 王思农,李贝琳达,马迪安·哈布萨,韩芳,马浩。 Linformer:具有线性复杂度的自注意力。 arXiv 预印本 arXiv:2006.04768, 2020。
[68] 魏秀英,张云晨,李宇航,张相国,龚瑞浩,郭金阳,刘相龙。异常值抑制+:通过等效且最优的移位和缩放对大型语言模型进行精确量化。 arXiv 预印本 arXiv:2304.09145, 2023。

[69] 魏秀英,张云辰,张相国,龚瑞浩,张尚航,张琪,于凤伟,刘相龙。异常值抑制:突破低位转换器语言模型的极限。 arXiv 预印本 arXiv:2209.13325, 2022。

[70] Sean Welleck、Kianté Brantley、Hal Daumé Iii 和 Kyunghyun Cho。非单调顺序文本生成。国际机器学习会议,第 6716-6726 页。 PMLR,2019。

[71] 塞缪尔·威廉姆斯、安德鲁·沃特曼和大卫·帕特森。 Roofline:针对多核架构的富有洞察力的视觉性能模型。 ACM 通讯,52(4):65–76,2009 年。

[72] 吴嘉祥,冷丛,王宇航,胡庆浩,程健。适用于移动设备的量化卷积神经网络。 IEEE 计算机视觉和模式识别会议论文集,第 4820-4828 页,2016 年。

[73] 吴章浩,刘志坚,林吉,林玉君,韩松。精简版变压器
长短期关注。 arXiv 预印本 arXiv:2004.11886, 2020。
[74] 徐进,谭旭,罗仁前,宋凯涛,李健,秦涛,刘铁岩。
NAS-BERT:与神经架构搜索无关的任务和自适应大小的 bert 压缩。第 27 届 ACM SIGKDD 知识发现和数据挖掘会议论文集,第 1933-1943 页,2021 年。

[75] 姚哲伟,Reza Yazdani Aminabadi,张敏佳,吴晓霞,李从龙,何宇雄。 ZeroQuant:针对大型变压器的高效且经济的训练后量化。 arXiv 预印本 arXiv:2206.01861, 2022。

[76] 姚哲伟、董震、郑章成、Amir Gholami、于嘉丽、Eric Tan、王乐源、黄启晶、王一达、Michael W Mahoney 和 Kurt Keutzer。
HAWQV3:二元神经网络量化。 arXiv 预印本 arXiv:2011.10680,
2020.
[77] 姚哲伟、Amir Gholami、Kurt Keutzer 和 Michael W Mahoney。 Pyhessian:通过粗麻布透镜的神经网络。 arXiv 预印本 arXiv:1912.07145,
2019.
[78] 尹宜春,陈成,尚立峰,蒋欣,陈晓,刘群。
AutotinyBERT:高效预训练语言模型的自动超参数优化。 arXiv 预印本 arXiv:2107.13686,
2021。
[79] 于世兴、姚哲伟、Amir Gholami、Zhen Dong、Sehoon Kim、Michael W Mahoney 和 Kurt Keutzer。 Hessian 感知剪枝和最佳神经植入。
IEEE/CVF 计算机视觉应用冬季会议论文集,第 3880-3891 页,2022 年。

[80] 袁志航,牛林,刘家伟,刘文宇,王兴刚,商玉章,孙光宇,吴强,吴家祥,吴秉哲。 RPTQ:大型语言模型基于重新排序的训练后量化。 arXiv 预印本 arXiv:2304.01089, 2023。

[81] 阿里·哈迪·扎德、伊萨克·埃多、奥马尔·穆罕默德·阿瓦德和安德烈亚斯·莫索沃斯。 GOBO:量化基于注意力的 NLP 模型,以实现推理中的低延迟和节能。 2020 年第 53 届 IEEE/ACM 国际微架构研讨会 (MICRO),第 811-824 页。 IEEE,2020。

[82] Ofir Zafrir、Guy Boudoukh、Peter Izsak 和 Moshe Wasserblat。 Q8BERT:泉
tized 8 位伯特。 arXiv 预印本 arXiv:1910.06188, 2019。
[83] 张冬青,杨蛟龙,叶东强子,华刚。 LQ-Nets:高度准确且紧凑的深度神经网络的学习量化。在欧洲计算机视觉会议 (ECCV) 会议记录中,第 365-382 页,

2018.
[84] Susan 张,Stephen Roller,Naman Goyal,Mikel Artetxe,Moya Chen,Shuohui Chen,Christopher Dewan,Mona Diab,Xian Li,Xi Victoria Lin,等。 OPT:开放预训练的 Transformer 语言模型。 arXiv 预印本 arXiv:2205.01068, 2022。
[85] 张伟,侯鲁,尹宜春,尚立峰,陈晓,蒋欣,刘群。 TernaryBERT:蒸馏感知的超低位 bert。 arXiv 预印本 arXiv:2009.12812, 2020。

[86] 张一凡、董振、杨欢瑞、卢明、曾正清、郭延东、Kurt Keutzer、杜丽、张尚航。 Qd-bev:用于多视图 3D 对象检测的量化感知视图引导蒸馏。 2023 年。

[87] 赵瑞奇、胡宇伟、乔丹·多策尔、克里斯·德萨、张志如。使用离群值通道分割改进神经网络量化,无需重新训练。
国际机器学习会议,第 7543-7552 页。 PMLR,2019。

8 附录

8.1 每通道稀疏模式中的数据倾斜

图 7 提供了第一个 LLaMA-7B 块中不同线性层上每个输出通道的非零条目分布。这个情节
显示非零分布严重倾斜，少数通道包含更大比例的非零值。这
偏斜分布使得使用稀疏矩阵有效执行计算具有挑战性，因为很难分布
非零元素均匀分布在并行处理单元上。这促使我们修改内核来处理具有大量的通道
异常值以减少稀疏矩阵的运行时开销。如选项卡中所述。 4、我们观察到运行时开销增加超过 100%
当采用基于 CSR 的标准内核时。然而，当使用密集矩阵向量运算单独处理前 10 行时，
我们能够将敏感值的运行时开销大幅减少到 20%，敏感值和异常值的运行时开销减少到 40-45%。

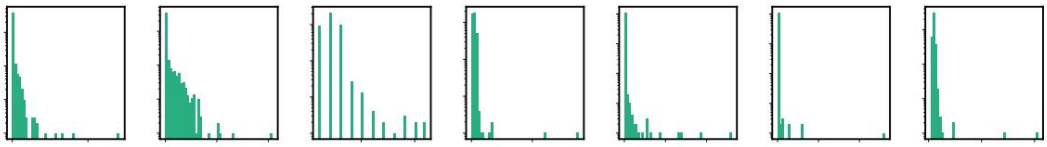

图 7：第一个 LLaMA-7B 块中 7 个不同线性层中每个输出通道的非零条目数量直方图。直方图
揭示了一些通道的存在，这些通道比其他通道包含更多的非零条目，突出了稀疏模式中的倾斜
跨越线性层内的不同通道。

表 4：在硬件上生成 128 个令牌时量化为 3 位的 LLaMA 7B、13B 和 30B 的延迟和内存使用情况的硬件分析
A6000 GPU。前两行显示了使用标准内核处理 CSR 时具有不同稀疏级别的 SqueezeLLM 的性能
矩阵。最后两行显示了使用混合内核处理 CSR 矩阵的不同稀疏度级别的 SqueezeLLM 的性能
它隔离出非零值最多的前 10 个通道并单独处理它们。

稀疏内核	方法	延迟（秒）			峰值内存 (GB)		
		LLaMA-7B	LLaMA-13B	LLaMA-30B	LLaMA-7B	LLaMA-13B	LLaMA-30B
	挤压LLM	1.5	2.4	4.9	2.9	5.4	12.5
标准	挤压LLM (0.05%)	3.4	5.6	12.0	3.0	5.4	12.6
	挤压LLM (0.45%)	3.9	6.2	12.5	3.2	5.8	13.7
杂交种	挤压LLM (0.05%)	1.8	2.9	5.9	3.0	5.5	12.7
	挤压LLM (0.45%)	2.2	3.4	7.2	3.2	5.9	13.8

8.2 消融研究

8.2.1 基于灵敏度的量化。在我们的消融研究中，我们研究了敏感度感知加权聚类对
非均匀量化的性能。在选项卡中。 5，我们比较了敏感度感知和敏感度不可知方法的性能
在 LLaMA-7B 模型的 3 位量化的背景下。对于与灵敏度无关的量化，我们应用非加权 k 均值聚类
稀疏度为 0%、0.05% 和 0.45%。结果表明，虽然单独的非均匀量化可以减少困惑
在不考虑灵敏度的情况下从 28.26（RTN 均匀量化）到 18.08，结合灵敏度感知聚类对于
将困惑度降低到 7.75。这种改进在所有稀疏级别上都是一致的。

表 5：比较 3 位 LLaMA-7B 模型上的灵敏度不可知和基于灵敏度的非均匀量化的消融研究
量化，通过 C4 基准的困惑度来衡量。 FP16 中的基线模型的困惑度达到 7.08。

方法	与灵敏度无关 (↓)	基于灵敏度 (↓)
挤压LLM	18.08	7.75
挤压LLM (0.05%)	8.10	7.67
挤压LLM (0.45%)	7.61	7.56

8.2.2 稀疏程度对 SqueezeLLM 的影响。在图 8（左）中，我们展示了 3 位量化 LLaMA-7B 模型在 C4 基准上的困惑度结果，其中提取的敏感值作为稀疏矩阵的百分比不同，范围从 0% 到 0.2%。该图表明，随着敏感值的稀疏程度超过 0.05%，困惑度增益会减小。因此，我们在所有实验中将敏感值保持在 0.05% 的固定稀疏水平。

此外，在图 8（右）中，我们将不删除敏感值作为稀疏矩阵（仅删除异常值）时的性能与删除 0.05% 的敏感值的情况进行比较。在这两种情况下，我们通过增加稀疏矩阵中包含的异常值的百分比来控制稀疏水平以获得权衡曲线。结果表明，同时具有敏感值和离群值的稀疏配置始终优于仅具有离群值的配置。

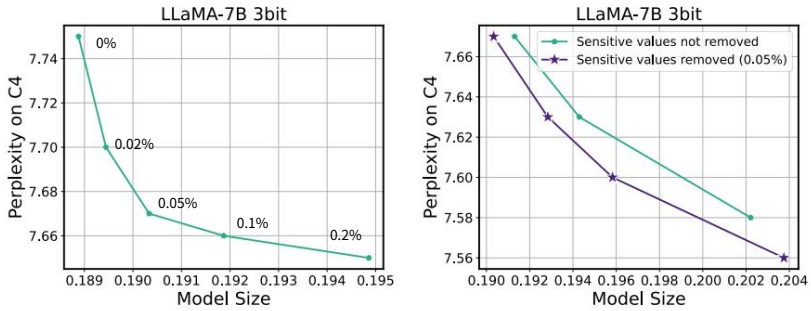

图 8：（左）模型大小（通过 FP16 模型的大小标准化）和稀疏矩阵中包含的不同百分比的敏感值的困惑度权衡。这里，稀疏矩阵中不包含异常值。 （右）作为稀疏矩阵不去除敏感值（仅去除异常值）与去除 0.05% 敏感值的情况的性能比较。在这两种情况下，都是通过控制稀疏矩阵中包含的异常值的百分比来获得权衡。

8.2.3 分组对 SqueezeLLM 的影响。在图 10 中，我们探索了将分组合并到 SqueezeLLM 作为提高量化性能的替代方法的有效性。我们比较了三种配置：SqueezeLLM，其中 (i) 使用组大小 1024 和 512（绿色）进行分组，(ii) 将组大小 1024 与稀疏度 0.05% 相结合的混合方法（蓝色），以及 (iii)具有不同稀疏程度（紫色）的密集和稀疏分解方法，其中保留 0.05% 的敏感值并调整异常值的百分比。结果清楚地表明，与纯密集和稀疏分解方法相比，分组和混合方法都会导致次优权衡。

这可以归因于两个因素。首先，密集稀疏分解是异常值问题的直接解决方案。相比之下，虽然分组可以通过将异常值隔离在各个组中来在一定程度上减轻异常值的影响，但它并没有为这个问题提供直接的解决方案。此外，当与非均匀量化结合时，分组可能会在存储要求方面带来大量开销，因为它需要为每组存储一个 LUT。与仅需要存储每组的缩放和零点值的统一量化方法相比，这可能是相当大的开销。

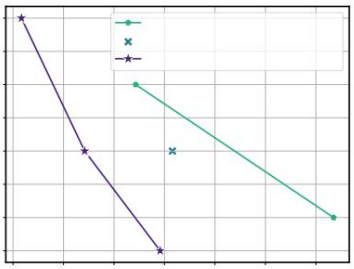

图 9：模型大小（按 FP16 模型的大小标准化）以及 LLaMA-7B 模型 3 位量化的分组和密集和稀疏分解的复杂度权衡。在这里，我们将 SqueezeLLM 与 (i) 使用组大小 1024 和 512（绿色）进行分组，(ii) 将组大小 1024 与稀疏性水平 0.05%（蓝色）相结合的混合方法，以及 (iii)具有不同稀疏程度的密集和稀疏分解方法（紫色）。与分组和混合方法相比，纯密集和稀疏分解实现了更好的大小与复杂度权衡。

8.2.4 OBD 框架与 OBS 框架非均匀量化的比较。虽然我们的方法采用
最佳脑损伤（OBD)框架,以最小化量化过程中模型最终输出的扰动,它是
值得注意的是,最佳脑外科医生（OBS)框架也可以被视为替代方案。大多数现有的 LLM 解决方案
包括 GPTQ [19]、AWQ [45] 和 SpQR [13] 在内的量化都利用了 OBS 框架,旨在最大限度地减少
各个层中的输出激活。在这项消融研究中,我们证明了 OBD 框架优于 OBS 框架。

在OBD框架下,可以重新制定确定非均匀量化配置的优化目标
as arg min // - 其中表示一批输入激活,该对象可以近似为加权 k 均值聚类
问题,其中每个权重均按相应输入激活大小的平方进行加权。这确实导致了基于激活的
AWQ 框架中的敏感性/重要性指标[45]。

在图 8.2.4 中,我们比较了使用 OBS 框架对 LLaMA-7B 模型进行 3 位量化的 C4 数据集的困惑度与
OBD 框架。在通过调整提取的异常值数量获得的所有稀疏级别中,基于 OBD 的 SqueezeLLM
框架的性能比使用 OBS 框架的替代方案高出 0.3 左右的困惑点。

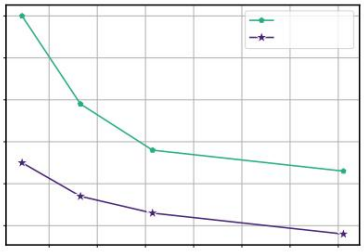

图 10:LLaMA-7B 模型的 3 位量化的模型大小（通过 FP16 模型的大小标准化)和困惑度权衡
用于确定非均匀量化的最佳脑外科医生 (OBS) 框架与最佳脑损伤 (OBD) 框架
配置。通过调整所提取的异常值的稀疏程度来实现权衡。在所有稀疏级别,OBD 框架,
它是 SqueezeLLM 的基础,作为替代方法,其性能始终优于 OBS 框架。

8.3 其他硬件分析结果

在选项卡中。 6,我们使用序列长度 1024 提供额外的硬件分析结果。所有实验设置和详细信息均在
与 Sec 相同。 5.4 和选项卡。 3.

表 6:在硬件上生成 1024 个令牌时量化为 3 位的 LLaMA 7B、13B 和 30B 的延迟和内存使用情况的硬件分析
A6000 GPU。第一行是非量化的 FP16 基线,第二行和第三行分别是非分组和分组的 GPTQ。
请注意,所有 GPTQ 结果均具有激活顺序。第四、五、六行显示了具有不同稀疏度的 SqueezeLLM 的性能
级别,第四行表示仅密集 SqueezeLLM。

方法	延迟（秒）			峰值内存 (GB)		
	LLaMA-7B	LLaMA-13B	LLaMA-30B	LLaMA-7B	LLaMA-13B	LLaMA-30B
FP16	26.5	47.0	OOM	13.1	25.2	OOM
GPT-Q	12.6	19.0	36.8	3.3	6.0	13.8
GPT-Q (g128)	110.7	176.1	500.8	3.4	6.2	14.3
挤压LLM 挤压	13.6	21.2	42.6	3.4	6.1	13.9
LLM (0.05%)	16.1	24.9	49.3	3.4	6.2	14.1
挤压LLM (0.45%)	19.1	29.0	58.9	3.6	6.6	15.1