doi:10.11887/j.cn.202401012 http://journal.nudt.edu.cn

注意力机制量化剪枝优化方法

何源宏[1,2],姜晶菲[1,2*],许金伟[1,2]
(1. 国防科技大学 计算机学院,湖南 长沙 410073;
2. 国防科技大学 并行与分布计算全国重点实验室,湖南 长沙 410073)

摘 要:面向基于注意力机制模型的巨大计算和访存开销问题,研究量化和剪枝协同优化的模型压缩技术,提出针对注意力机制中查询、键、值、概率共四个激活值矩阵的对称线性定点量化方法。同时,提出概率矩阵剪枝方法和渐进式剪枝策略,有效降低剪枝精度损失。在不同数据集上的实验结果表明,针对典型基于注意力机制模型BERT,在较低或者没有精度损失的情况下该优化方法可达到4位或8位定点量化、0.93~0.98的稀疏度,大幅度降低模型计算量,为加速量化稀疏模型的推理奠定良好的基础。

关键词:自然语言处理;注意力机制;量化;剪枝
中图分类号:TP18 **文献标志码**:A **开放科学(资源服务)标识码(OSID)**:
文章编号:1001-2486(2024)01-113-08

Quantization and pruning optimization method for attention mechanism

HE Yuanhong[1,2], *JIANG Jingfei*[1,2*], *XU Jinwei*[1,2]
(1. College of Computer Science and Technology, National University of Defense Technology, Changsha 410073, China;
2. National Key Laboratory of Paralle and Distributed Computing, National University of Defense Technology, Changsha 410073, China)

Abstract: To address the significant computation and memory overhead of models based on attention mechanism, model compression techniques, such as collaborative optimization of quantization and pruning, were studied. A symmetric linear fixed point quantization method was proposed for four activation matrices of query, key, value and probability in the attention mechanism. Meanwhile, a probability matrix pruning method and a progressive pruning strategy were proposed to effectively reduce the pruning accuracy loss. Experimental results on different datasets show that for the typical attention-based model BERT, this optimization method can achieve 4 bit or 8 bit fixed point quantization and 0.93~0.98 sparsity with little or no accuracy loss, which greatly reduces the model computation and lays a strong foundation for accelerating the inference of quantized sparse models.

Keywords: natural language processing; attention mechanism; quantization; pruning

近年来,基于Transformer的模型(如BERT[1]、GPT-2[2])在机器翻译、句子分类、问题回答等自然语言处理(natural language processing, NLP)任务中实现了最先进的(state-of-the-art, SOTA)成果。在一些具有挑战性的任务上,典型的BERT模型处理效果甚至超过人类[3]。相比传统的循环神经网络[4](recurrent neural network, RNN)模型和长短期记忆[5](long short-term memory, LSTM)模型,基于Transformer的模型采用注意力机制[6],能更有效地捕获输入序列中的上下文信息,使得模型精度显著提高。

在注意力机制的原始计算流程中,注意力机制的输入由查询($\boldsymbol{Q}$)、键($\boldsymbol{K}$)和值($\boldsymbol{V}$)三个激活值矩阵组成。注意力机制首先通过$\boldsymbol{Q}$和$\boldsymbol{K}$的相乘计算得到分数矩阵($\boldsymbol{S}$),然后采用归一化指数函数(softmax)对分数矩阵进行逐行操作得出概率矩阵($\boldsymbol{P}$)。最后将$\boldsymbol{P}$与$\boldsymbol{V}$相乘得到输出。相较于RNN和LSTM模型只计算了输入的局部信息,注意力机制计算了每一对查询向量和键向量的注意力结果,从而实现了对全局关系的提取。但由于注意力机制的计算开销和输入序列的长度的平方成正比,全局信息提取带来模型精度提升的同时也使得计算复杂度大幅增加。

收稿日期:2022-10-17
基金项目:重点实验室稳定支持重点资助项目(WDZC20215250103)
第一作者:何源宏(1998—),男,湖南宁乡人,硕士研究生,E-mail:heyuanhongcs@nudt.edu.cn
***通信作者**:姜晶菲(1974—),女,辽宁昌图人,研究员,博士,硕士生导师,E-mail:jingfeijiang@nudt.edu.cn

例如,对于涉及图像或长文本的任务,输入的序列长度可能高达 16×10^3,而对于具有 16×10^3 个 Token 的单个输入序列,BERT-Base 中一个自注意力模块的浮点运算次数[7]高达 861.9×10^9。基于注意力机制模型的计算复杂性给实时响应系统和移动设备上的开发部署带来了巨大挑战,模型的轻量化方法是解决计算难题的关键手段。

现有的大部分工作[7-19]着重于对基于注意力机制模型的权重进行量化和剪枝,取得了较好的效果。但是在计算复杂度较高的注意力机制中,大多数模型仍采用单精度浮点表示的稠密激活值矩阵进行运算,激活值不属于权重量化剪枝的范畴,因此计算开销依然很高。本文对基于注意力机制模型的优化训练方法展开研究,通过有效的量化和剪枝方法对激活值矩阵进行量化和剪枝,从而达到大幅降低基于注意力机制模型的计算量和访存量的目的,使得模型推理更适应轻量化智能应用的需求。

1 相关工作

深度神经网络往往被认为是过度参数化的,目前学术界提出了许多方法去除冗余的参数来实现存储或者计算的优化。使用的方法包括量化、剪枝、知识蒸馏、参数共享、专用的 FPGA 和 ASIC 加速器等[20-23]。

1.1 深度神经网络的量化方法

量化指用低精度数(如定点 8 位)表示高精度数(如浮点 32 位)。除了可以减少神经网络模型占用的空间大小,量化还能在支持低精度运算的硬件中提升模型运算速度。文献[8]利用量化感知训练(quantization-aware training, QAT)和对称线性量化将 BERT 量化到 8 位定点整数,同时在下游任务中几乎没有精度损失,从而节省 75% 的存储空间占用。文献[9]利用权重矩阵的二阶海森信息对 BERT 进行混合精度和分组量化。文献[10]利用聚类的思想将 BERT 中 99.9% 的权重量化到 3 bit,剩余权重按照原样存储,但是需要在专用硬件上实现推理过程。为了得到更高的模型压缩比,文献[11]使用三值化的权重分割来获得二值化的权重,文献[12]通过引入二值注意力机制和知识蒸馏[24]得到二值化的权重,虽然二值化后的模型理论上相较于原始模型可以获得 32 倍的压缩比,但是上述两种方法均会导致模型精度相对于原始模型精度有显著的下降,难以在实际应用中起到较好的效果。

1.2 深度神经网络的剪枝方法

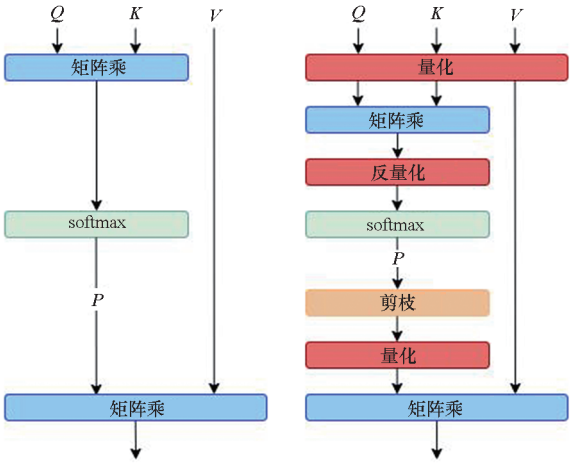
剪枝分为结构化剪枝和非结构化剪枝。非结构化剪枝去除不重要的神经元,可以显著减少模型的参数量和理论计算量,但通用平台擅长的规则计算难以利用其稀疏性,需要专用的硬件或者计算库来支持稀疏矩阵运算。magnitude 剪枝[13-15]是应用最广泛也是效果较好的非结构化剪枝方法之一,该方法认为如果一个权重或激活值的绝对值越小,那么对后续结果的影响也越小,则可以将其置为 0。文献[14]发现在 BERT 的预训练过程中,对权重使用 magnitude 剪枝在低稀疏度(0.3 ~ 0.4)情况下不会影响模型在下游任务中的精度,而在高稀疏度(高于 0.7)情况下模型难以在下游任务中取得较好的效果。文献[16-17]提出了一种基于训练中权重移动方向的剪枝方法,该方法认为在训练过程中,权重的更新如果越靠近 0,则表明该权重越不重要;权重的更新如果越远离 0,则表明该权重越重要。

结构化剪枝通常以神经网络中的一个注意力头[18-19]或整层[18]为剪枝的基本单位。结构化剪枝的相关工作输出的模型在计算模式上更匹配 CPU、GPU 规则计算构架,因此仍然可以通过 CPU 和 GPU 完成推理加速,但是存在稀疏度远低于非结构化剪枝的稀疏度等劣势。本文面向实时响应系统和低算力移动设备的计算优化需求,主要研究非结构化剪枝及量化在基于注意力机制模型上的计算优化技术,能够在较好平衡任务准确率的条件下达到良好的计算压缩效果。这种量化剪枝技术能输出更为精简、采用低位宽数据表示的稀疏化模型。面向该类模型,很多工作研究了低位宽智能加速器,通过专用结构高效处理低位宽数据,为低位宽计算和非结构化稀疏计算定制更适合的运算结构,如 4 位定点运算器、专用稀疏格式的数据运算通路等。相较于 CPU、GPU 等面向规则计算的体系结构,这些结构能进一步获得大幅能效提升,更加适用于面向实时响应系统和低算力移动设备应用的需求。

2 量化剪枝优化方法

注意力机制的原始计算流程和优化后的计算流程如图 1 所示,在本文提出的注意力机制模型量化剪枝优化总体流程中,输入激活值矩阵 $\boldsymbol{Q}$、$\boldsymbol{K}$、$\boldsymbol{V}$ 是输入信息在不同空间的表达,在 BERT 模型中具有相同的维度:①对于给定的输入激活值矩阵 $\boldsymbol{Q}$、$\boldsymbol{K}$、$\boldsymbol{V}$,首先量化 $\boldsymbol{Q}$ 和 $\boldsymbol{K}$ 到 4 位或 8 位定点整数,计算分数矩阵的结果;②将分数矩阵进行反

量化并送入 softmax 得到 $\boldsymbol{P}$;③将 $\boldsymbol{P}$ 中绝对值低于阈值的数剪枝为 0;④将稀疏的 $\boldsymbol{P}$ 量化到 4 位或者 8 位定点整数并将其与量化后的 $\boldsymbol{V}$ 相乘,得到注意力机制的输出结果。第②步和第③步进行反量化和量化操作的原因是直接将线性量化的数据送入 softmax,这一类非线性运算会显著影响模型在数据集上的精度[25],同时为了后续量化计算,需要将剪枝后的结果进行量化。考虑到对计算过程进行简化,该方法根据是否为同一矩阵乘的输入采取了分组量化,即 $\boldsymbol{Q}$ 和 $\boldsymbol{K}$ 为一组,$\boldsymbol{P}$ 和 $\boldsymbol{V}$ 一组。同一组的两个矩阵量化到相同的比特数,而不同组之间可以量化到不同比特数。



(a) 原始计算流程　(b) 优化后计算流程

(a) Original computational flow　(b) Optimized computational flow

图 1　注意力机制的原始计算流程和优化后的计算流程

Fig. 1　Original computational flow of attention mechanism and optimized computational flow of attention mechanism

2.1　激活值矩阵的对称线性量化方法

对称线性量化、非对称线性量化、非线性量化、*K*-means 量化等是典型的模型量化方法[21]。对称线性量化相较于其他方法所需的硬件计算更加简单,更易于高效的专用加速器实现。本文采取的激活值矩阵的 k bit 对称线性量化公式定义如下:

$$\begin{cases} Q(r,S,\alpha)=\mathrm{clip}[\mathrm{round}(r\times S),-\alpha,\alpha] \\ \mathrm{clip}(x,a,b)=\min[\max(x,a),b] \\ \alpha=2^{k-1}-1 \\ S=\dfrac{\alpha}{\mathrm{EMA}[\max(|r|)]} \end{cases} \tag{1}$$

其中:Q 是量化函数;r 是输入的原始数据;S 是量化缩放因子;α 是量化到 k bit 时的最大值,例如,当量化到 4 位时,$\alpha=7$。由于注意力机制的输入是动态变化的,故使用指数滑动平均(exponential moving average, EMA)来收集其信息,并在训练的过程中确定 S。clip 是截断函数,确保量化后的值不超出 k bit 所能表示的范围。round 函数起到四舍五入的作用。对称线性量化本质上是将输入 r 映射到对称区间$[-\alpha,\alpha]$之中。对于反量化公式定义如下:

$$DQ(S,q)=S\times q \tag{2}$$

式中,DQ 是反量化函数,q 表示输入的量化值。量化过程和反量化过程均采用相同的 S。训练过程中为了传递量化误差,本文采用量化感知训练实现伪量化;同时考虑到 round 函数的不可导性,采用直通估计器(straight-through estimator, STE)进行梯度反向传播。

2.2　渐进剪枝训练

渐进剪枝策略[26]能有效避免剪枝带来的模型精度显著下降,参考其原理,本文采用 magnitude 剪枝方法实现,在每次剪枝迭代时,将待剪枝的激活值矩阵中绝对值低于预设阈值的数剪枝为 0。训练过程采用三段式渐进剪枝方法,在总迭代次数为 T 的训练过程中,前 t_i 次迭代只进行模型的微调;在 t_i 到 t_f 次迭代的过程中稀疏度 s 从 0 提升到预设的目标稀疏度 s_t,在这期间 s 以三次函数的趋势呈先快后慢的非线性增长;最后维持目标稀疏度 s_t 进行微调实现精度的回调。其中稀疏度 s 表示一个矩阵中数值为 0 的权重所占的比值,具体取值如分段函数(3)所示:

$$s=\begin{cases} 0 & 0\leqslant t<t_i \\ s_t-s_t\left(1-\dfrac{t-t_i}{T-t_i-t_f}\right)^3 & t_i\leqslant t<T-t_f \\ s_t & \text{其他} \end{cases} \tag{3}$$

本文提出的量化剪枝优化方法与工作 Sanger[7]有一定的相似性。Sanger 首先将 $\boldsymbol{Q}$ 和 $\boldsymbol{K}$ 量化到 INT4,计算得到分数矩阵后,对其按行进行 softmax 得到 $\tilde{\boldsymbol{P}}$。这是为了以较小的计算代价获得 $\boldsymbol{P}$ 的近似结果,然后对 $\tilde{\boldsymbol{P}}$ 剪枝得到掩码矩阵。最后通过对掩码矩阵的打包和分割重新完成注意力机制的浮点运算。故 Sanger 实际的推理过程仍然以浮点的形式。但本文提出的优化方法在前向推理中直接采取量化的 $\boldsymbol{Q}$ 和 $\boldsymbol{K}$ 进行运算,并进一步使用剪枝量化后的 $\boldsymbol{P}$ 和量化后的 $\boldsymbol{V}$ 进行运算,最终完成注意力机制的推理过程,而不需要重新计算。后续的实验也表明,本文方法产生的稀疏度在大部分数据集中都远高于 Sanger。

3 实验分析

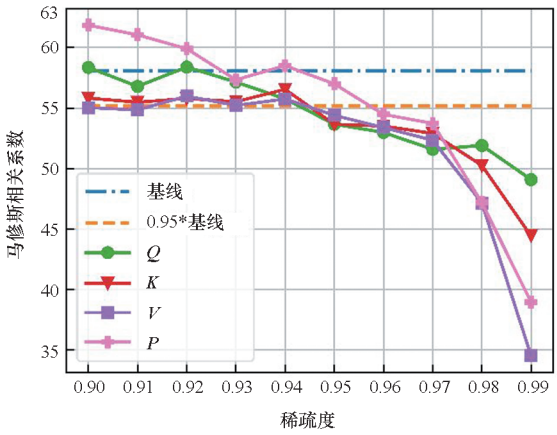
3.1 实验设置

本文在 BERT-Base[1] 模型上针对通用语言理解评估[27]（general language understanding evaluation，GLUE）基准进行了一系列的实验分析。GLUE 包含了 9 个数据集来对自然语言模型进行训练、评估和分析，分别为 CoLA、RTE、MRPC、STS-B、SST-2、QNLI、QQP、MNLI、WNLI，但是本文没有在 WNLI 数据集上进行评估，因为它通常被认为存在一定的问题[1]。此外，本文也在问答任务 SQuAD v1.1[28] 上进行了评估。对于 MRPC、QQP、SQuAD v1.1 数据集采用 F1 值评估；对于 STS-B 数据集用皮尔逊相关系数评估；对于 CoLA 数据集用马修斯相关系数评估；对于 GLUE 中剩余的数据集均采用准确率评估。所有数据集的评估指标都与模型的精度成正相关。

实验中 BERT 实现代码是基于 HuggingFace 的 Transformer 库[29]，并在 PyTorch 深度学习框架完成了量化剪枝实验和模型精度的评估。所有的实验均运行在单 GPU（NVIDIA Tesla V100 PCIe 32 GB）上。因为本文提出的方法不涉及预训练，故直接将预训练好的 BERT-Base 模型在数据集上进行量化剪枝训练。CoLA、RTE、MRPC、STS-B 数据集训练时的批大小为 8，学习率为 2×10^{-5}；SST-2、QNLI、QQP、MNLI、SQuAD v1.1 数据集训练时的批大小为 32，学习率为 3×10^{-5}。每次训练为 10 个 epoch：①前 3 个 epoch 对应渐进稀疏的第一阶段，模型的稀疏度为 0；②接下来 4 个 epoch 对应渐进剪枝的第二个阶段，此时稀疏度 s 随时间从 0 增长到目标稀疏度 s_t；③最后 3 个 epoch 对应渐进剪枝的第三个阶段，此时维持目标稀疏度 s_t 对模型进行微调完成最后的训练，该阶段可以在一定程度上缓解剪枝带来的模型精度损失。在整个训练过程中，都实施本文提出的对称线性量化流程。

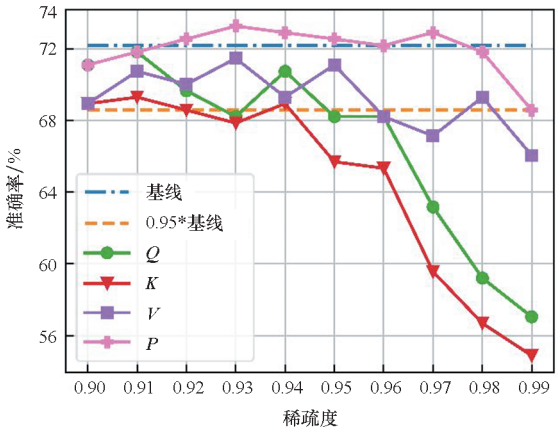
3.2 实验结果

图 2 表明了各数据集在不同的目标稀疏度 s_t（大于 0.9）下对 $\boldsymbol{Q}$、$\boldsymbol{K}$、$\boldsymbol{V}$、$\boldsymbol{P}$ 单独剪枝的结果，基线表示没有进行剪枝和量化的结果，0.95* 基线表示基线的 95% 的精度。首先，结果显示 $\boldsymbol{Q}$ 和 $\boldsymbol{K}$ 在剪枝效果上具有一致性：即 $\boldsymbol{Q}$ 和 $\boldsymbol{K}$ 的结果曲线具有相似的变化趋势。而 $\boldsymbol{P}$ 和 $\boldsymbol{V}$ 也有类似的性质。这表明同一个矩阵乘的两个输入具有相似的剪枝效果。其次，随着稀疏度的提升，除 CoLA 数据集外，$\boldsymbol{P}$ 和 $\boldsymbol{V}$ 的剪枝效果逐渐优于 $\boldsymbol{Q}$ 和 $\boldsymbol{K}$。本文认为这是因为 $\boldsymbol{P}$ 之前的 softmax 运算隐式的将重要的信息进行了提取，并赋予其较大的值，这也表明 $\boldsymbol{P}$ 更适合进行剪枝。最后，在较高的稀疏度（大于 0.95）下，所有矩阵剪枝后的模型精度均出现了明显的下降，尤其是 $\boldsymbol{Q}$ 和 $\boldsymbol{K}$。



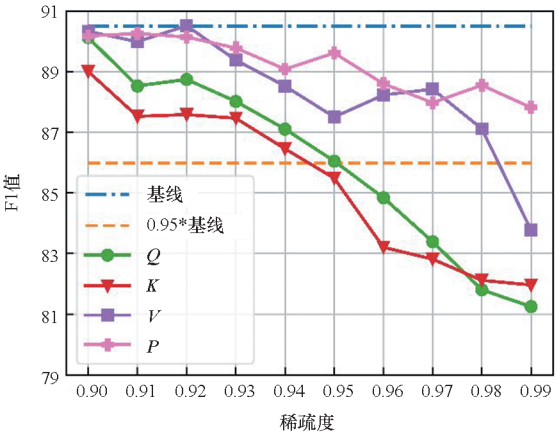
（a）CoLA 数据集

（a）CoLA dataset



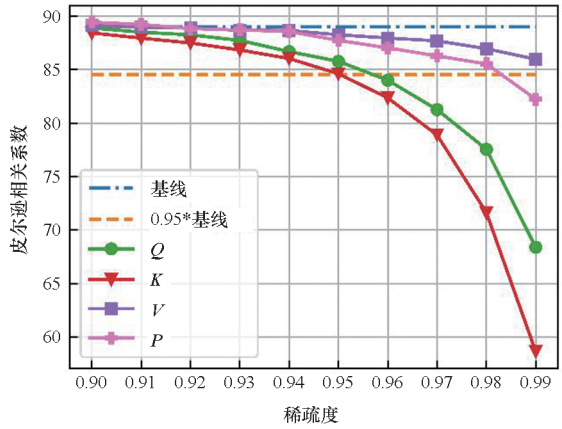
（b）RTE 数据集

（b）RTE dataset



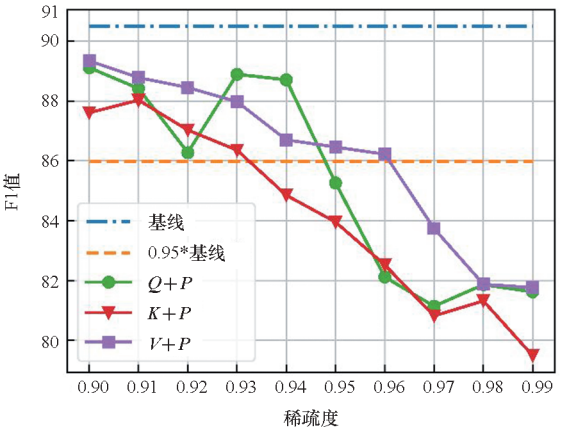
（c）MRPC 数据集

（c）MRPC dataset



(d) STS-B 数据集

(d) STS-B dataset



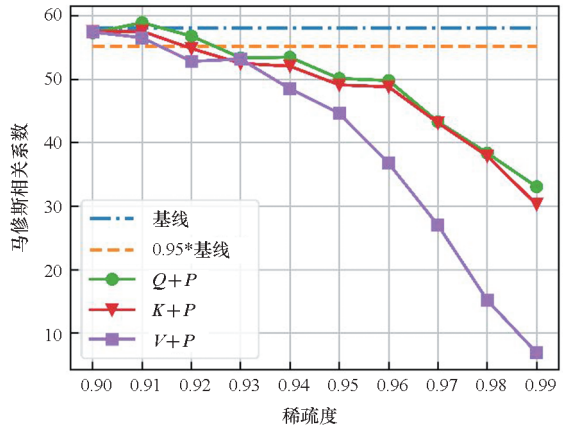
(c) MRPC 数据集

(c) MRPC dataset

图 2　Q、K、V、P 的单独剪枝

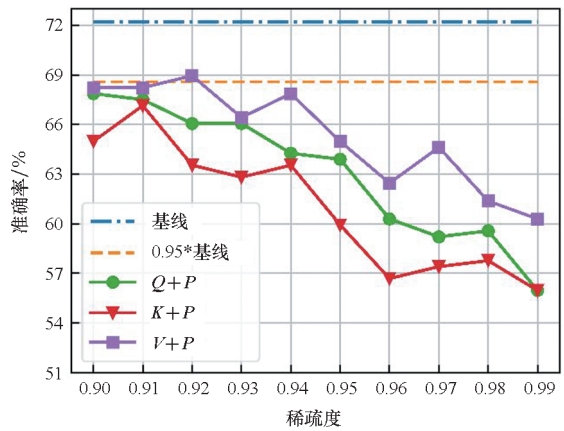
Fig. 2　Separate pruning of Q, K, V and P

在单独剪枝的基础上,本文还比较了 Q、K、V 分别和 P 进行联合剪枝后模型精度的差异,如图 3 所示。从图 2 和图 3 的对比可以看出,在大多数情况下,Q、K、V 在单独剪枝中具有最好效果的那一个矩阵,在和 P 进行联合剪枝时也会具有最好的效果。但是 Q、K、V 分别和 P 的联合剪枝比单独剪枝时的精度下降得更快。同时考虑到 P 的大小与输入序列长度的大小的二次方成正比,故本文在接下来的量化实验中只针对 P 进行剪枝。



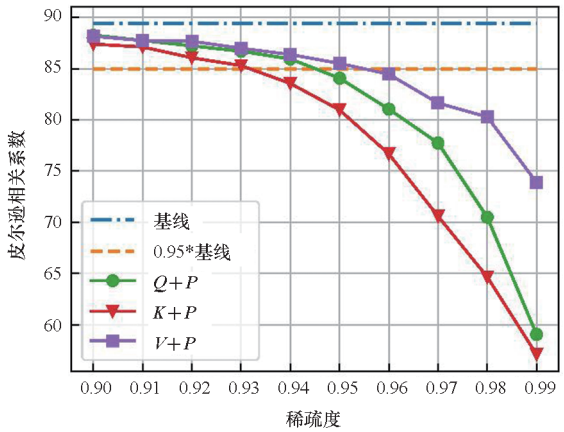
(a) CoLA 数据集

(a) CoLA dataset



(b) RTE 数据集

(b) RTE dataset



(d) STS-B 数据集

(d) STS-B dataset

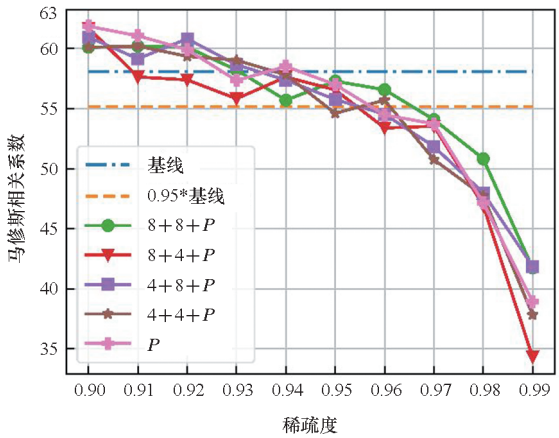
图 3　Q、K、V 分别和 P 的联合剪枝

Fig. 3　Joint pruning of Q, K, and V with P respectively

实验结合了量化和剪枝来探究量化带来的影响。实验设置了两种量化位宽,分别是 4 位和8 位。结合两种量化位宽的设定和分组量化,图 4 展示了四种量化方案和剪枝共同优化的结果。图例中的 8 +4 表明对于 Q 和 K 量化到 8 位,P 和 V 量化到 4 位。从图 4 中可以清楚地看到,在大多数情况下,量化对 P 的剪枝效果的影响较小。在 STS-B 数据集中,即使量化比特方案为4 +4,在相同稀疏度水平下,量化剪枝比非量化剪枝还能获得更好的结果,这表明量化和剪枝在实际的应用中具有正交性,同时量化能提升模型的鲁棒性。

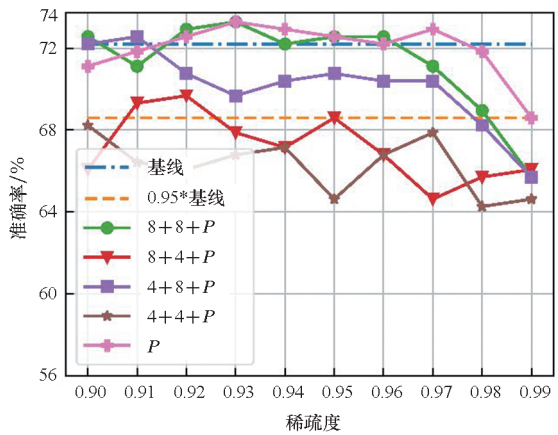
在 STS-B 和 CoLA 数据集中,不同量化比特方案产生的模型精度差异较小。但是在 RTE 和

MRPC 数据集中,不同的量化比特设定带来了明显的精度差异,表明不同的数据集对于量化的敏感程度不一样。同时较低的量化比特设定并不意味着较低的模型精度。例如,4 +4 的量化方案在 CoLA、MRPC、STS-B 数据集中并不是精度最低的,甚至在 STS-B 数据集中,4 +4 的量化方案能获得最好的整体效果,而 8 +8 的量化方案在整体效果表现上最差。



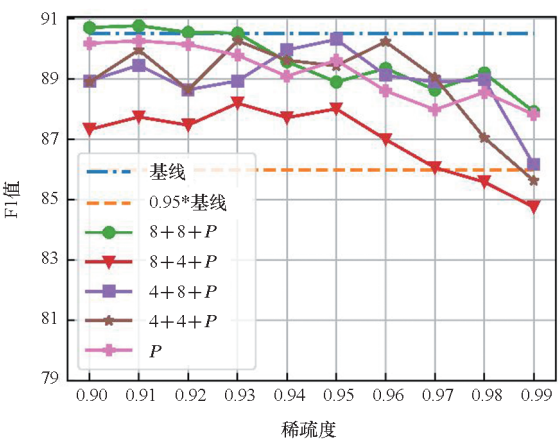
(a) CoLA 数据集

(a) CoLA dataset



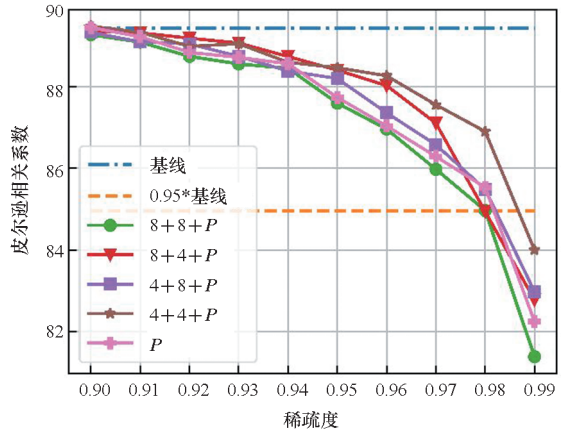
(b) RTE 数据集

(b) RTE dataset



(c) MRPC 数据集

(c) MRPC dataset



(d) STS-B 数据集

(d) STS-B dataset

图 4 量化剪枝协同优化

Fig. 4 Quantization and pruning collaborative optimization

在图 2 到图 4 的三组实验中,RTE 数据集均为结果波动较大的数据集之一,本文认为是 RTE 数据集对于量化和剪枝较敏感,特别是在量化和剪枝协同优化时,波动更为明显。总体来说,即使在较高的稀疏度下,本文的量化方案也不会使模型的精度产生显著的下降。

3.3 和 Sanger 的比较

这一节将本文提出的优化方法与 Sanger[7] 稀疏模式进行了比较,结果汇总见表 1。Sanger 首先通过低比特计算完成对于 $\boldsymbol{P}$ 的预测,再利用预测得到的 $\tilde{\boldsymbol{P}}$ 重新完成 $\boldsymbol{Q}$ 和 $\boldsymbol{K}$ 的稀疏计算。本文的方案最高能在 SST-2 数据集上获得 0.98 的稀疏度,此时量化方案为 8 +8,而 Sanger 的稀疏度仅仅为 0.90。表 1 说明本文提出的量化剪枝优化方法在 GLUE 基准和 SQuAD v1.1 数据集上的稀疏度全部超越了 Sanger,且最多比 Sanger 高出 0.14。同时模型精度也高于 Sanger,而相对于基线的精度下降也非常小。在 SST-2、CoLA、RTE 这三个数据集中,本文提出方法的模型精度甚至超越了基线。

一些相关研究表明[8-10],针对深度神经网络一定程度的压缩(量化、剪枝、知识蒸馏等)能够提升模型的鲁棒性,因此使得压缩后模型的精度略微超过基线模型,而当压缩率继续提升时模型的精度可能会逐渐下降。但目前尚无相关研究能够定量说明什么程度的压缩能起到泛化的作用。从定性的角度,本文分析深度学习模型一般具有较大的信息冗余和噪声,一定的压缩使得这些冗余噪声被更高比例去除,因此使得模型精度得以保持甚至略有上升。

表 1　不同数据集上的量化剪枝结果

Tab. 1　Quantization and pruning results on different datasets

数据集	评价指标	基线精度	Sanger		本文		
			稀疏度	精度	稀疏度	量化方案	精度
MRPC	F1 值	**90.50**	0.89	89.79	0.95	4 + 8	90.30
SST-2	准确率	92.36	0.90	92.08	0.98	8 + 8	**92.43**
STS-B	皮尔逊相关系数	**89.43**	0.81	89.03	0.93	8 + 4	89.07
CoLA	马修斯相关系数	58.08	0.79	58.69	0.93	4 + 4	**59.02**
RTE	准确率	72.20	0.85	68.95	0.96	8 + 8	**72.56**
MNLI	准确率	**83.90**	0.91	83.36	0.95	4 + 4	83.49
QQP	F1 值	**87.93**	0.88	87.58	0.95	8 + 4	87.61
QNLI	准确率	**91.71**	0.89	90.92	0.95	8 + 4	91.02
SQuAD v1.1	F1 值	**87.83**	0.91	87.30	0.93	8 + 8	87.38

3.4　计算轻量化分析

模型稀疏量化可以从计算次数和数据表示两方面降低模型推理的计算量,提升计算速度。以 BERT 模型的一个自注意力头模块为例,推理任务的绝大多数计算是 $\boldsymbol{Q}$、$\boldsymbol{K}$、$\boldsymbol{P}$、$\boldsymbol{V}$ 涉及的矩阵运算,对于两个 n 阶方阵的矩阵乘法,其他配套操作(包括量化、反量化、softmax、剪枝)的计算量约为其 $1/n$,当 n 较大时,这部分运算量可忽略。模型计算复杂度一般由矩阵乘加量代表。当把模型稀疏度提升至 $x\%$ 时,稀疏模型计算复杂性约可降为原模型稠密矩阵计算的 $1-x\%$。稀疏矩阵一般以 CSR 稀疏格式存储,大小约为稠密矩阵的 $2(1-x\%)$。同时,当模型将原有数据位宽 W 量化为 W/y 时,压缩后数据的存储量将降低为原来的 $1/y$。

根据上述分析以及表 1 所获得的 BERT 模型稀疏量化结果,在 $\boldsymbol{P}\times\boldsymbol{V}$ 计算稀疏度为 95%、所有矩阵 4 +4 量化协同优化下,得到式(4):

$$\text{加速比}=\frac{1}{a\times(1-s_P)\times\frac{1}{N_{PV}}+(1-a)\times\frac{1}{N_{QK}}} \tag{4}$$

整个模块的理论推理速度可提升约 16 倍。其中:a 为 $\boldsymbol{P}\times\boldsymbol{V}$ 计算量占比,$(1-a)$ 为 $\boldsymbol{Q}\times\boldsymbol{K}$ 计算量占比,在该模块中 a 约为 50%;s_P 为 $\boldsymbol{P}$ 的稀疏度;N_{PV} 和 N_{QK} 分别为 $\boldsymbol{P}$、$\boldsymbol{V}$ 和 $\boldsymbol{Q}$、$\boldsymbol{K}$ 量化到 N 比特后 N 位定点运算相较于 32 位浮点运算的加速比,根据硬件运算单元速度的一般结论,4 位定点乘加单元的运算速度为 32 位浮点乘加单元运算速度的 8 倍以上。就存储量而言,4 + 4 量化后 $\boldsymbol{Q}$、$\boldsymbol{K}$、$\boldsymbol{V}$ 占用存储空间的大小可降为原始的 1/8,同时在 95% 稀疏度下 $\boldsymbol{P}$ 的大小约为原始的 1/80。

上述分析面向可有效处理稀疏运算和低位宽量化运算的定制加速器,有较多前沿相关工作[3,7]基于此思路开展,相较于通用平台,加速器不仅能高效处理上述运算,还可消除 CPU、GPU 等由于指令和调度执行带来的额外控制开销,大幅提升最终性能。

4　结论

针对基于注意力机制模型计算复杂度和访存开销过高等问题,本文提出了一种量化剪枝协同优化方法,该方法通过采用对称线性量化和渐进剪枝来降低注意力机制的计算复杂量。本文提出的方法在较低或者没有精度损失的情况下,在 BERT 模型上有着较好的实验效果,对于 GLUE 基准和 SQuAD v1.1 数据集,该方法将注意力机制运算涉及的 $\boldsymbol{Q}$、$\boldsymbol{K}$、$\boldsymbol{V}$、$\boldsymbol{P}$ 采用分组量化,其中 $\boldsymbol{Q}$ 和 $\boldsymbol{K}$ 为一组,$\boldsymbol{V}$ 和 $\boldsymbol{P}$ 为一组,分别量化到 4 位或者 8 位,并且剪枝 $\boldsymbol{P}$,最终可以得到 0.93 ~ 0.98 的稀疏度,大幅度降低模型计算量。在 SST-2、CoLA、RTE 数据集上,该优化方法得到的模型在精度上甚至超过了基线。

本文提出的方法在各方面也优于 Sanger 稀疏模式,包括稀疏度和最终的模型精度。但本文所提出优化方法是非结构化稀疏模式,该稀疏模式具有稀疏度高和难以在通用平台加速等特点,故下一步的工作展望为在专用的硬件平台上充分利用其量化和稀疏特性,将量化剪枝的理论计算性能提升转换为实际推理能耗和延迟的降低。

参考文献(References)

[1] DEVLIN J, CHANG M W, LEE K, et al. BERT: pre-training of deep bidirectional transformers for language understanding [EB/OL]. (2019 - 05 - 24) [2022 - 10 - 11]. https://arxiv.org/abs/1810.04805.

[2] BROWN T B, MANN B, RYDER N, et al. Language models are few-shot learners[EB/OL]. (2020 - 07 - 22) [2022 - 10 - 11]. https://arxiv.org/abs/2005.14165.

[3] WANG H R, ZHANG Z K, HAN S. SpAtten: efficient sparse attention architecture with cascade token and head pruning[C]//Proceedings of the IEEE International Symposium on High-Performance Computer Architecture (HPCA), 2021.

[4] MIKOLOV T, KARAFIAT M, BURGET L, et al. Recurrent neural network based language model[C]//Proceedings of the 11th Annual Conference of the International Speech Communication Association, 2010.

[5] GRAVES A. Long short-term memory [M]//GRAVES A. Supervised sequence labelling with recurrent neural networks. Berlin, Heidelberg: Springer-Verlag Berlin Heidelberg, 2012: 37 - 45.

[6] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is all you need[C]//Proceedings of the 31st Conference on Neural Information Processing Systems, 2017.

[7] LU L Q, JIN Y C, BI H R, et al. Sanger: a co-design framework for enabling sparse attention using reconfigurable architecture [C]//Proceedings of the 54th Annual IEEE/ACM International Symposium on Microarchitecture, 2021.

[8] ZAFRIR O, BOUDOUKH G, IZSAK P, et al. Q8BERT: quantized 8 bit BERT[C]//Proceedings of the 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing-NeurIPS Edition (EMC2-NIPS), 2019.

[9] SHEN S, DONG Z, YE J Y, et al. Q-BERT: hessian based ultra low precision quantization of BERT[C]// Proceedings of the AAAI Conference on Artificial Intelligence, 2020.

[10] ZADEH A H, EDO I, AWAD O M, et al. GOBO: quantizing attention-based NLP models for low latency and energy efficient inference [C]//Proceedings of the 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO), 2020.

[11] BAI H, ZHANG W, HOU L, et al. BinaryBERT: pushing the limit of BERT quantization [EB/OL]. (2021 - 07 - 22) [2022 - 10 - 11]. https://arxiv.org/abs/2012.15701.

[12] QIN H T, DING Y F, ZHANG M Y, et al. BiBERT: accurate fully binarized BERT[EB/OL]. (2022 - 03 - 12) [2022 - 10 - 11]. https://arxiv.org/abs/2203.06390.

[13] CHEN T L, FRANKLE J, CHANG S Y, et al. The lottery ticket hypothesis for pre-trained BERT networks [C]// Proceedings of the 34th International Conference on Neural Information Processing Systems, 2020.

[14] GORDON M, DUH K, ANDREWS N. Compressing BERT: studying the effects of weight pruning on transfer learning[C]// Proceedings of the 5th Workshop on Representation Learning for NLP, 2020.

[15] HAN S, MAO H Z, DALLY W. Deep compression: compressing deep neural network with pruning, trained quantization and huffman coding[C]// Proceedings of the Computer Vision and Pattern Recognition, 2016.

[16] SANH V, WOLF T, RUSH A M. Movement pruning: adaptive sparsity by fine-tuning[C]// Proceedings of the 34th Conference on Neural Information Processing Systems, 2020.

[17] LI Y C, LUO F L, TAN C Q, et al. Parameter-efficient sparsity for large language models fine-tuning [EB/OL]. (2022 - 05 - 23)[2022 - 10 - 11]. https://arxiv.org/abs/2205.11005.

[18] HOU L, HUANG Z Q, SHANG L F, et al. DynaBERT: dynamic bert with adaptive width and depth[C]//Proceedings of the 34th Conference on Neural Information Processing Systems, 2020.

[19] MICHEL P, LEVY O, NEUBIG G. Are sixteen heads really better than one? [C]//Proceedings of the 33rd International Conference on Neural Information Processing Systems, 2019.

[20] GANESH P, CHEN Y, LOU X, et al. Compressing large-scale transformer-based models: a case study on BERT[J]. Transactions of the Association for Computational Linguistics, 2021, 9: 1061 - 1080.

[21] TAY Y, DEHGHANI M, BAHRI D, et al. Efficient transformers: a survey[J]. ACM Computing Surveys, 2020, 55(6): 109.

[22] LYU Y S, GUO H, HUANG L B, et al. GraphPEG: accelerating graph processing on GPUs [J]. ACM Transactions on Architecture and Code Optimization, 2021, 18(3): 30.

[23] DOU Y, VASSILIADIS S, KUZMANOV G K, et al. 64-bit floating-point FPGA matrix multiplication [C]//Proceedings of the ACM/SIGDA 13th International Symposium on Field-Programmable Gate Arrays, 2005.

[24] HINTON G, VINYALS O, DEAN J, et al. Distilling the knowledge in a neural network[EB/OL]. (2015 - 03 - 09) [2022 - 10 - 11]. https://arxiv.org/abs/1503.02531.

[25] KIM S, GHOLAMI A, YAO Z W, et al. I-BERT: integer-only BERT quantization [C]// Proceedings of the 38th International Conference on Machine Learning , 2021.

[26] ZHU M, GUPTA S. To prune, or not to prune: exploring the efficacy of pruning for model compression[EB/OL]. (2017 - 09 - 13) [2022 - 10 - 11]. https://arxiv.org/abs/1710.01878.

[27] WANG A, SINGH A, MICHAEL J, et al. GLUE: a multi-task benchmark and analysis platform for natural language understanding [C]//Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, 2018.

[28] RAJPURKAR P, ZHANG J, LOPYREV K, et al. SQuAD: 100,000 + questions for machine comprehension of text[C]//Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, 2016.

[29] WOLF T, DEBUT L, SANH V, et al. HuggingFace′s transformers: state-of-the-art natural language processing[EB/OL]. (2020 - 07 - 14) [2022 - 10 - 11]. https://arxiv.org/abs/1910.03771.

(编辑: 王颖娟, 杨琴)