doi:10.3772/j.issn.1002-0470.2024.06.002

一种基于自适应 PoT 量化的无乘法神经网络训练方法①

刘 畅②*** 张 蕊③***** 支 天***

(*中国科学院大学 北京 100049)

(**中国科学院计算技术研究所智能处理器研究中心 北京 100190)

(***中国科学院计算技术研究所处理器芯片全国重点实验室 北京 100190)

摘 要 当前的深度神经网络的训练过程中需要包含大量的全精度乘累加(MAC)操作,导致神经网络模型的线性层(包含卷积层和全连接层)的计算过程所需的能耗占整体能耗的绝大部分,达 90% 以上。本文提出了一种自适应逐层缩放的量化训练方法,可支持在神经网络计算全流程(前向传播和后向传播)将全部线性层中的全精度乘法替换为 4 位定点数加法计算和 1 位异或运算。实验结果表明,上述方法在能耗和准确率方面都优于现有方法,可支撑在训练过程中减少达 95.8% 的线性层能耗,在 ImageNet 数据集上的卷积神经网络和在 WMT En-De 任务上的 Transformer 网络得到小于 1% 的精度损失。

关键词 神经网络;量化;训练加速;低能耗

近年来,深度神经网络(deep neural network,DNN)在许多人工智能应用中取得了显著的效果。然而,训练 DNN 会消耗大量能量。例如,训练一个深度双向语言表征模型需要耗 948 kW·h 的能源,这超过了全球平均每年人均家庭用电量(731 kW·h)[1-2]。此外,DNN 训练的巨大能源消耗会显着增加碳排放,从而对全球气候产生负面影响[3-4]。因此,迫切需要降低 DNN 训练的能耗。

在 DNN 训练中,线性层(包括卷积层和全连接层)的能耗占总能耗的绝大部分,超过 90%[5-7],因为乘法累加(multiple accumulate,MAC)操作中大量使用耗能的全精度 32 位浮点数(32bit floating point,FP32)乘法。FP32 乘法运算非常耗能,部分原因在于它的高位宽。例如,FP32 乘法的能耗大约是 FP16 乘法的 4 倍。因此,为了获得低能耗的 DNN,有相关方法通过量化技术用低位宽乘法代替全精度乘法。其中一些量化方法使用全精度(FP32)预训练模型[8-9]进行微调得到低位宽模型,因此它们不能减少训练过程中的能耗。还有方法通过将权重(weight,W)、激活(activation,A)和激活梯度(gradient,G)量化为 16 位[10-11]或 8 位[12-15]从头开始训练模型。此外,乘法本身的能耗明显高于加法和移位等低功耗操作。例如,32 位定点数(32bit Integer,INT32)乘法的能耗大约比 INT32 加法高 22 倍。因此,有一些无乘法的方法可以直接用加法和移位等低功耗运算代替乘法[16-22]。然而,以上现有的量化方法和无乘法方法不能以低精度低能耗操作代替前向和后向传播过程中的所有全精度线性层乘法。因此,这些工作可以减少的训练能耗是有限的。

本工作提出了一种无乘法 MAC(multiplication-free multiple accumulate,MF-MAC),用 INT4 加法和 1 位异或(XOR)操作代替 FP32 乘法,比现有方法的能耗更低。为了支持 MF-MAC,本工作提出了一种自适应分层缩放 PoT 量化(adaptive layer-wise scal-

① 国家重点研发计划(2018AAA0103300),国家自然科学基金(62102399,U22A2028,U20A20227)和中国科学院稳定支持基础研究领域青年团队计划(YSBR-029)资助项目。

② 女,1996 年生,博士生;研究方向:计算机系统结构,深度学习算法;E-mail:liuchang18s@ict.ac.cn。

③ 通信作者,E-mail:zhangrui@ict.ac.cn。

(收稿日期:2023-02-03)

ing power-of-two quantization, ALS-PoTQ) 方法。首先,该方法使用可变的缩放系数来适应 W、A 和 G 的所有数据范围,以将它们转换为统一的 PoT 数据格式。此外,为了保持训练稳定并提高准确性,本工作提出了一种权重偏差校正(weight bias correction, WBC)技术来校正 W 的偏差,以及一种参数化比率裁剪(parameterized ratio clipping, PRC)技术来增加 A 的长尾区域分辨率。最后,结合上述技术,完整的 ALS-PoTQ 方法没有引入额外的乘法,而现有方法[9,12-15]在量化数据时引入了乘法。因此,所提出的 MF-MAC 和 ALS-PoTQ 中的所有操作都是低能耗的。为了验证 ALS-PoTQ 方法是否对网络准确率造成影响,本文在 ImageNet [23]上的 AlexNet[24]、ResNet18[25]、ResNet50[25]模型和 WMT En-De 任务上的 Transformer 模型进行了实验,它们的准确率和双语评估替补(bilingual evaluation understudy, BLEU)下降均小于 1%。同时,与全精度训练相比,该方法在训练期间减少了达 95.8% 的线性层能耗。总而言之,该方法在能耗和准确性方面都优于现有方法。

1 相关工作

近年来,为了避免深度学习中乘法的高能耗,许多低精度的工作在乘法之前将 W、A 或 G 量化为低精度(位宽)[9-15,26-34]。在这些工作中,一些量化方法[8-9]从全精度预训练模型开始微调,因此它们不能减少训练过程的能耗。其他方法通过将权重 W、激活 A 和激活梯度 G 量化为 16 位定点数/浮点数[10-11]、8 位定点数/浮点数[12-15]或其他格式,如 radix-4 数[35]。此外,还有一些无乘法工作直接用低能耗操作代替乘法,例如加法[16]、按位移位[18-21]或它们的组合[22]。AdderNet[16]将过滤器和输入特征之间的 L1 范数距离作为输出响应,并用加法代替线性层中的乘法。该方法仍然使用 FP32 数字进行计算和存储,在基于 ImageNet 数据集的 ResNet 模型上准确率降低了大约 3%。此外,用移位代替乘法的方法是基于对数量化方法。对数量化方法将全精度数据量化为基数的 0 和幂。例如,Ultra-Low[35]使用 radix-4 的对数数据格式来表示梯度,需要专门的 radix-4 硬件支持。当对数函数的基数为 2 时,称为 PoT(power-of-two)量化,乘法可以用按位位移代替。INQ[17]将预训练的权重分成 2 组,一组是 PoT 量化的,而另一组是全精度重新训练的,以补偿精度下降。ShiftCNN [18]将全精度预训练模型的 W 量化为 PoT 格式,而 LogNN[19]将全精度预训练的 W 和 A 都量化为 PoT 数。以上所有方法都需要全精度的预训练模型,而不是从头开始训练。因此,它们不能用于减少训练的能耗。同时,还有一些工作可以降低训练的能耗:Deepshift[20]使用 DeepShift-Q 和 DeepShift-PS 2 种训练方法将所有 W 转换为 PoT 数,给网络带来了 4.42% 到 5.59% 的准确率下降。对数无偏量化(logrithmic unbiased quantization, LUQ)[21]在训练期间将具有简直操作的对数无偏量化应用于量化梯度,实现了小于 2.00% 的精度下降。这些工作用按位移位替换了训练中的一部分乘法。但是,它们在前向或反向传播过程中至少保留了线性层中多达 1/3 的乘法。

2 自适应逐层缩放 PoT 量化与无乘法

本节介绍一种自适应逐层缩放 PoT 量化方法和后续的无乘法 MAC。

2.1 PoT 量化与 PoT 数乘法

本节给出 PoT 量化的定义[17-18,20,35]以及 PoT 量化后数字相乘的计算方式。PoT 数 p 的值是 2 的幂或是 0:

$$\{0, \pm 2^{-2^{b-2}+1}, \pm 2^{-2^{b-2}+2}, \cdots, \pm 2^{2^{b-2}-1}\} \tag{1}$$

其中,b 是表示数字的位宽。位宽 b 的 PoT 数包含 1 个符号位和 $b-1$ 个指数位。基本的 PoT 量化方法如下:

$$e = Round(\log_2(|f|)) \tag{2}$$

$$p = \begin{cases} 0 & e < -2^{b-2}+1 \\ \mathrm{sign}(f) \cdot 2^{2^{b-2}-1} & e < 2^{b-2}-1 \\ \mathrm{sign}(f) \cdot 2^{e} & \text{其他} \end{cases} \tag{3}$$

其中,$Round(\,)$ 为对数据进行四舍五入操作,f 为未量化前浮点数。将 FP32 数转换为 b 位 PoT 数后,2 个 b 位 PoT 数之间的乘法可以用对数域中的 (b_1)

位定点数加法和符号翻转(1 位异或操作)替代:

$$2^k \cdot 2^m = 2^{k+m} \tag{4}$$

$$flip(s_1, s_2) = s_1 \oplus s_2 \tag{5}$$

其中,k 和 m 为整数,组成 PoT 数字的对数,$flip$ 表示数字符号翻转,s_1 和 s_2 为数字的正负符号,⊕表示异或操作。

2.2　ALS-PoTQ 算法

首先观察 W、A 和 G 在不同训练代数和不同网络以及不同层中的分布。如图 1(a) ~ (c)所示,DNN 中 W、A 和 G 的分布都是尖峰和长尾的近对数正态分布。换句话说,大多数数据集中在 0 附近,少量数据的绝对值相对较大。与此一致的是,PoT 数的分辨率在接近 0 的区域密集,在远离 0 的区域稀疏。这种一致性为将 PoT 量化应用于 DNN 提供了基础。

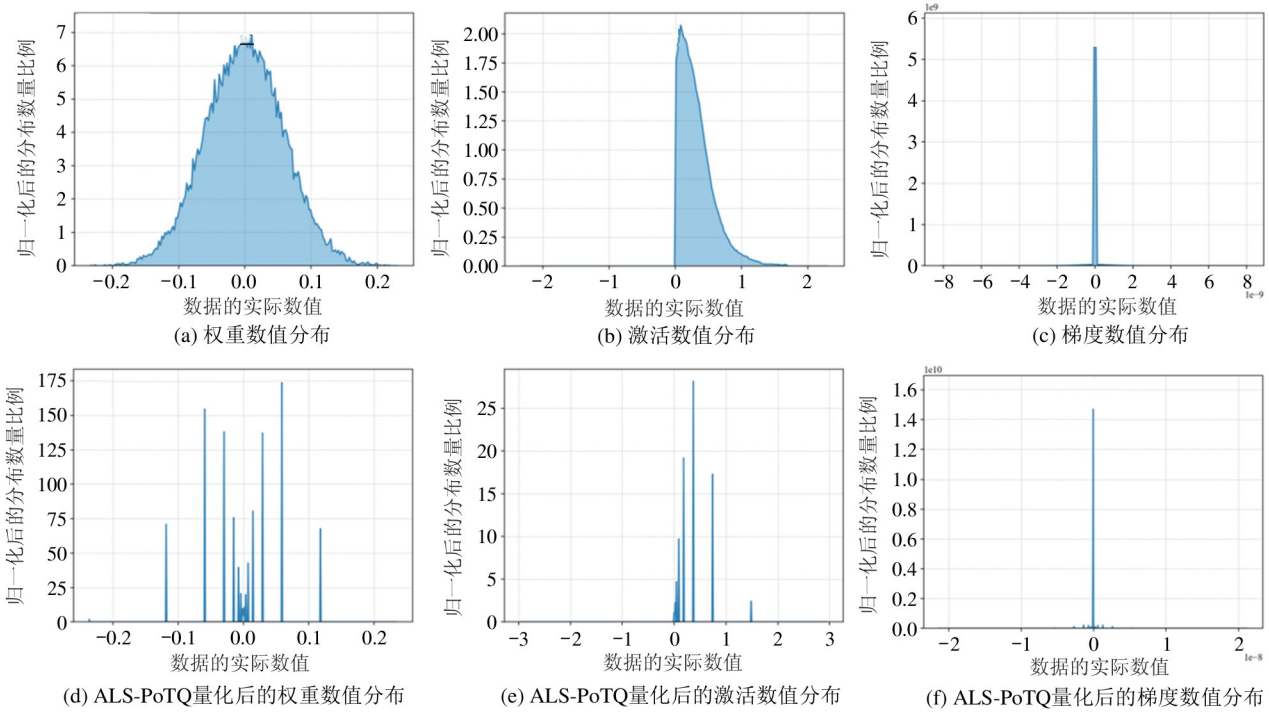


图 1　全精度数据与 ALS-PoTQ 量化后数据分布

然而,W、A 和 G 之间的数值范围相差较大,不同层的数据范围也不相同,且随着训练代数的变化而变化。其中,W 的数据范围变化较小,而 A 和 G 的数值范围随着训练过程剧烈变化,甚至可以达到几个数量级的差别,因此相关工作使用的基础 PoT 量化方法并不适用于量化所有数据。b 位 PoT 数的表示范围是 $[-2^{2^{b-2}-2}, 2^{2^{b-2}-1}]$,其中位宽 b 对于实际的硬件实现来说应该是固定的,所以该 PoT 数能表示的数据范围是固定的。而以上观察表明,量化方法需要更灵活的表示范围来拟合不同层、不同训练代数的 W、A 和 G 数据。

因此,为了自适应地改变固定位宽 PoT 数的表示范围,在 PoT 量化之前应用自适应缩放因子 α 来缩放 W、A 和 G。

假设一组 FP32 数据 $F = \{f_1, f_2, \cdots, f_n\}$($F$ 可以是线性层中的 W、A 或 G)。将 F 的数据范围限制到 b 位 PoT 数的表示范围 $[-2^{2^{b-2}-1}, 2^{2^{b-2}-1}]$,那么逐层缩放因子 α 为

$$\alpha = \frac{\max(|F|)}{2^{2^{b-2}-1}} \tag{6}$$

缩放后的数据为

$$F_{\text{scaled}} = \frac{F}{\alpha} \tag{7}$$

然后,如 2.1 节所述,将 F_{scaled} 量化到 PoT 数据 P,P 为数据块(数组),由 PoT 数 p_i 组成:$P = \{p_1, p_2, \cdots, p_k\}$。因此,$F$ 量化后的实际数值 $\hat{F}$ 为

$$\hat{F} = \alpha \cdot P \tag{8}$$

以上缩放操作引入了额外的乘法,如式(7)所

示，这与节约能耗的目的是相悖的。因此，进一步将 α 舍入到最接近的 PoT 数字：

$$\beta = Round(\log_2(\alpha)) \tag{9}$$

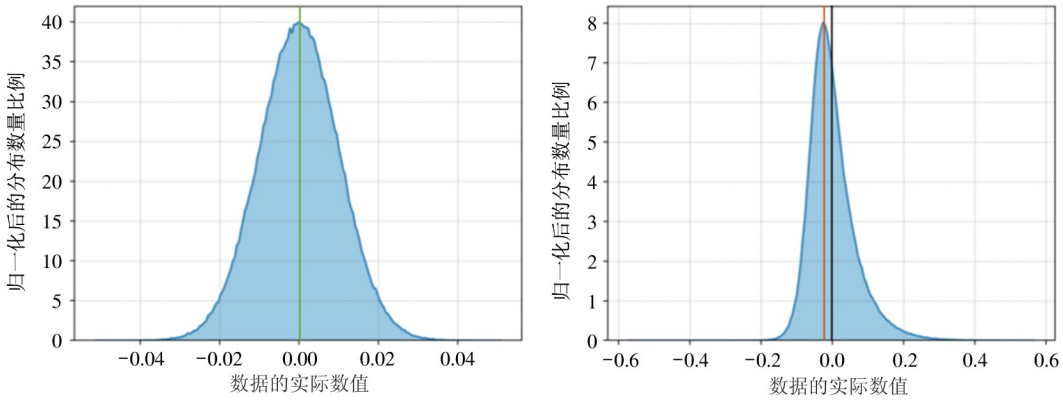
其中，β 是一个整数，并且对于各层和训练每一代中的 W、A 和 G 是不同的。根据实验经验，W 和 A 的值在[-5, -2]的范围内，G 的值在[-20, -10]的范围内。然后，式(7)中的乘法可以替换为 β 与 F 的指数部分之间的加法。此外，由于每层上万的数据（W、A 或 G）共享同一个标量 α，其存储空间和式(8)中的标量乘法可以忽略不计。本工作选择 b = 5，因此 MAC 中的每个 FP32 乘法都可以替换为 INT4 加法和 1 位异或操作。如图 1(d) ~ (f)所示，虽然 W、A 和 G 的分布差异很大，但用上述方法量化后的数据可以很好地拟合它们中的每一个。

此外，本文发现 W 和 A 的 PoT 量化仍会偶尔导致训练不稳定和准确率下降的问题，并提出了相应的解决方法。

对于 W，在实际实验中，其数据分布在训练过程中经常变化，其均值时常出现偏移，如图 2 所示。而有偏移的权重与 PoT 量化的对称性不一致。因此，量化后的 W 和原始的 W 之间的均方误差变大了。此外，W 的偏移可以在反向传播中累积到 G 上，而 G 上的偏差在理论上和经验上都被证明会影响训练收敛性[36]。因此，本工作提出了一种有效消除偏差的权重偏差校正技术。给定一组 FP32 权重 $W = \{w_1, w_2, \cdots, w_n\}$，使用权重偏差校正(WBC)来获得无偏差权重：

$$\widetilde{W} = W - mean(W) \tag{10}$$

其中，$mean(W)$ 为数据块(数组)中每个数字的平均值。应用 WBC 技术，ALS-PoTQ 训练可以正常收敛。与之前工作中使用的权重归一化技术[23]不同，WBC 技术不引入任何额外的乘法。



(a) 第1代权重数值分布　(b) 第5 000代权重数值分布

图 2　W 的数据偏移

A 在不同模型上对其进行自适应 PoT 量化会导致准确率下降大约 2% ~3%。这主要是因为 PoT 量化的刚性分辨率问题[37]。刚性分辨率问题是指无论 PoT 量化位宽多大，长尾区域的分辨率都是稀疏且固定的。如图 3 所示，4 位 PoT 量化仅在接近于 0 的小区域具有更高的分辨率。由于 PoT 数的格式不能改变，可以限制量化范围来提高长尾区域的分辨率，这就是量化工作中的“裁剪”技术。

因此，受参数化裁剪激活(parameterized clipping activation，PACT)[38]的启发，本文通过裁剪更改 A 的数据范围。因为数据分布是不同的，裁剪阈值应该适应不同的层。为了将提出的技术应用于所有线性层的 A，本工作提出了一种参数化比率裁剪(PRC)技术，其裁剪比率因子为 γ。给定一组 FP32 激活 $A = \{a_1, a_2, \cdots, a_n\}$ 和裁剪比率因子 γ，裁剪后的激活 $\bar{A} = \{\bar{a}_1, \bar{a}_2, \cdots, \bar{a}_n\}$ 定义为

$$\bar{a}_i = \begin{cases} -\max(|A|)\cdot\gamma & a_i < -\max(|A|)\cdot\gamma \\ \max(|A|)\cdot\gamma & a_i < \max(|A|)\cdot\gamma \\ a_i & \text{其他} \end{cases} \tag{11}$$

然后，在训练期间通过联合优化获得 γ，并应用直通估计器(straight-through estimator，STE)[39]对量化中的舍入操作进行反向传播。在自适应 PoT 量化

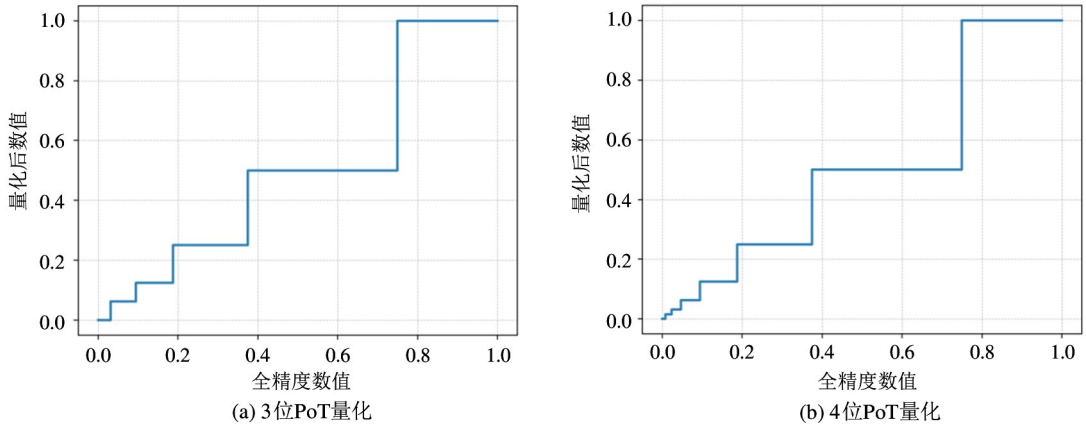


图 3　PoT 量化的刚性分辨率

之前将 PRC 技术应用于 A，并在损失函数中加入了 γ 的 L2 正则化器。PRC 技术同样也不会向线性层引入额外的乘法。

综合上述方法即为完整的 ALS-PoTQ 方法，并应用于所有的线性层 W、A 和 G。

2.2　MF-MAC 单元

2.1 节提出了 ALS-PoTQ 方法来获得 5 位 PoT 数。本节将进一步描述如何实现无乘法 MAC（MF-MAC）和 ALS-PoTQ。如图 4 所示，将 FP32 MAC（由 FP32 乘法和 FP32 累加组成）替换为 ALS-PoTQ 和 MF-MAC。MAC 的输入是 2 组 FP32 数 $\{x_{11},x_{12},\cdots,x_{ij},\cdots,x_{mn}\}$（缩写为 $\{x_{ij}\}$，称之为数据块）和 $\{x'_{ij}\}$。在本文中，首先使用 ALS-PoTQ 转换 2 个 FP32 输入。在 ALS-PoTQ 中，数据块 $\{x_{ij}\}$（或 $\{x'_{ij}\}$）首先通过逐层缩放因子 $\alpha = 2^{\beta}$，实际计算为整数 β 和 FP32 数的指数部分之间的 8bit Integer（INT8）加法，以得到缩放后数据 $\{y_{ij}\}$。然后，将缩放后的数字四舍五入到最接近的 PoT 数字 $p = 2^{e}$ 以获得带有 1 位符号块 $\{s_{ij}\}$ 的 INT4 数据块 $\{e_{ij}\}$。在 ALS-PoTQ 之后，FP32 数据块 $\{x_{ij}\}$ 和 $\{x'_{ij}\}$ 被转换为 2 个 INT4 数据块 $\{e_{ij}\}$ 和 $\{e'_{ij}\}$，2 个 1 位符号数据块 $\{s_{ij}\}$ 和

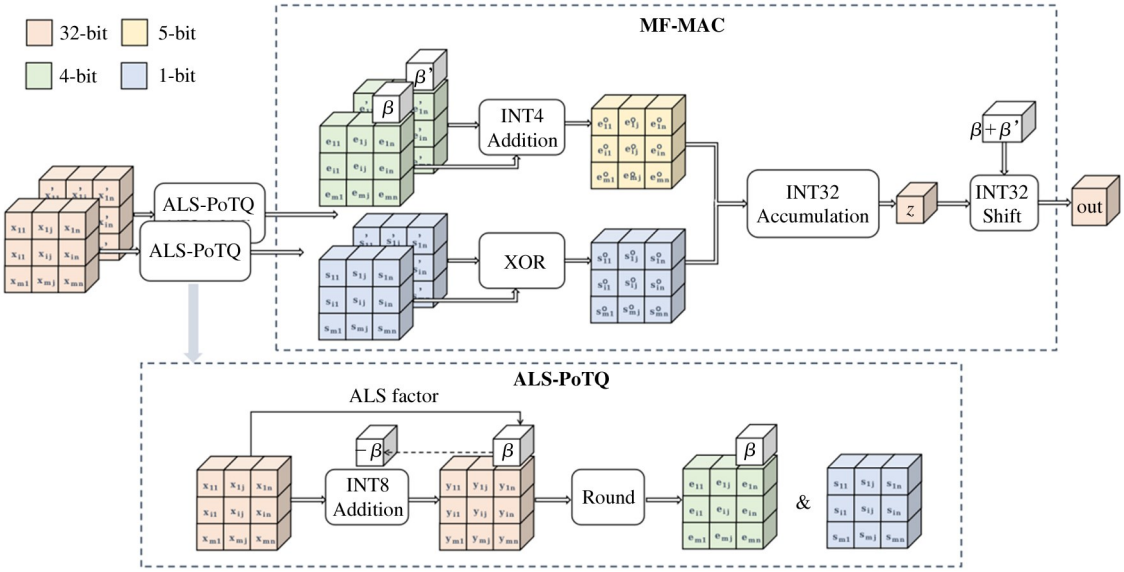


图 4　MF-MAC 与 ALS-PoTQ 计算流程示意图

$\{s'_{ij}\}$，以及 2 个整数 β 和 β'，它们是 MF-MAC 的输入。在 MF-MAC 中，首先应用一个 INT4 加法器来计算 $\{p_{ij}\}$ 和 $\{p'_{ij}\}$ 之间的 INT4 加法。同时，应用异或门来处理式(5)中的符号翻转操作；然后，将 INT4

加法器的输出与 XOR 门的输出相结合,并将带符号的数字累加到 INT32 数据 z; 最后,将 INT32 数据 z 移动 $\beta+\beta'$ 位以获得 MF-MAC 的输出。

3 实 验

3.1 训练能耗分析

本节分析了本文提出方法的能耗并将其与相关工作进行比较。首先,表 1 展示了在 45 nm 互补金属氧化物半导体技术(complementary metal oxide semiconductor techniques, CMOS Techniques)中实现的不同计算操作的单位能耗值,这与已有工作[22,40]的做法一致。在本工作中,将每个线性层 FP32 MAC 替换为由 INT4 加法和 XOR 门组成的 MAC。如表 1 所示,INT4 加法的能耗大约仅为 FP32 乘法的 0.4%,而异或门的能耗小于0.01 pJ[40]。此外,本工作将 MAC 中的 FP32 累加替换为 INT32 累加,这样可以减少约 84.0% 的能量。因此,与 FP32 MAC 相比,MF-MAC 可以减少大约 96.6% 的能量。此外,本工作考虑了 ALS-PoTQ 量化所引入的额外能耗:在 ALS-PoTQ 中引入了 INT8 加法、舍入运算,以及在 MF-MAC 中累加后的标量 INT32 移位。这些操作平均到每个数字的单位能耗大约为0.04 pJ。总而言之,与 FP32 MAC 相比,带有 ALS-PoTQ 量化器的 MF-MAC 可以减少大约 95.8% 的线性层计算能耗。

表 1 不同计算操作的功耗

	单位能耗/pJ				
乘法	FP32	INT32	FP8	INT8	INT4
	3.70	3.10	0.23	0.19	0.04
加法	FP32	INT32	INT16	INT8	INT4
	0.90	0.14	0.05	0.03	0.01
移位	INT32-4		INT32-3	INT4-3	
	0.96		0.72	0.08	

将本工作与相关工作进行全面比较,包括先进的低精度量化训练方法[12]以及无乘法网络[16-22]。表 2 列出了每种方法在前向和反向传播过程中用于替换 MAC 中的 FP32 乘法的操作。根据表 1 中这些操作的能耗,计算得到每个方法在ImageNet上训练

表 2 训练过程中的计算操作及能耗分析

方法	W	A	G	训练				
				从头训练	大数据集	线性层乘法替代操作 前传	线性层乘法替代操作 反传	能耗/J 前传+反传
原版训练	FP32	FP32	FP32	-	-	FP32 Mul	FP32 Mul	14.53
INQ	PoT5	FP32	FP32	×	√	FP32 Mul (INT32-4 Shift)	FP32 Mul	14.53
LogNN	PoT4	PoT4	FP32	×	×	FP32 Mul (INT3Add)	FP32 Mul (INT32-3 Shift)	14.53
ShiftCNN	PoT4	FP32	FP32	×	√	FP32 Mul (INT32-4 Shift)	FP32 Mul	14.53
ShiftAddNet	PoT5	INT32	INT32	√	×	INT32-4 Shift INT32 Add	INT32 Mul INT32-4 Shift	9.08
AdderNet	FP32	FP32	FP32	√	√	FP32 Add	FP32 Add	5.70
DeepShift-Q	PoT5	INT32	FP32	√	√	INT32-4 Shift	FP32 Mul INT8 Add	7.81
DeepShift-PS	PoT5	INT32	FP32	√	√	INT32-4 Shift	FP32 Mul INT8 Add	7.81
S2FP8	FP8	FP8	FP8	√	√	FP8 Mul	FP8 Mul	3.57*
LUQ	INT4	INT4	PoT5	√	√	INT4 Mul	Shift4-3	3.07*
ALS-PoTQ	PoT5	PoT5	PoT5	√	√	INT4 Add	INT4 Add	0.49

注:"*"表示该方法引入了额外计算,但为了公平比较,功耗值不包含这部分额外计算。

ResNet50 于一次迭代中的能耗。表 2 中比较结果表明,本文提出方法在训练功耗方面优于现有方法。此外,表 2 还列出了现有方法存在的其他一些缺陷:INQ[17]、LogNN[19]和 ShiftCNN[18]使用 FP32 预训练模型而不是从头开始训练,因此它们不能减少训练的能量消耗;LogNN[19]和 ShiftAddNet[22]没有在 ImageNet 等大规模数据集上进行实验;S2FP8[12]和 LUQ[21]在量化过程中引入了额外的乘法运算,这会增加训练能耗。

3.2　训练准确率

本工作在 TensorFlow[41]和 TensorPack 框架提供的官方模型上进行了训练实验。需要注意的是,权重的初始化器应该被设定为无截断正态分布(untruncated normal distribution)而不是有截断正态分布(truncated normal distribution)。为了进行全面的比较,本工作评估了图像分类任务的方法,因为大多数相关工作都选择该方法来评估性能。在 ILSVRC12 ImageNet 分类数据集[23]上使用 AlexNet[24]、ResNet18[25]、ResNet50[25]进行实验。表 3 给出了与相关工作的准确率比较,其中包括 INQ[17]、ShiftCNN[18]、AdderNet[16]、DeepShift[20]、Ultra-low[35]和 LUQ[21]。在这里,本方法不与 LogNN[19]和 ShiftAddNet[22]进行比较,这是因为它们没有将其方法应用于 ImageNet 等大规模数据集的训练,因此缺少相关数据。同样地,本文比较了 DeepShift 和 LUQ 工作中从头开始训练的结果,而不是它们的微调结果。然而,INQ 和 ShiftCNN 只有从预训练的 FP32 模型开始微调的结果,而不是从头开始训练,所以在这里展示了它们的推理准确率结果(用 * 号表示)。

表 3　ImageNet 数据集上的 CNN 模型准确率

网络	方法	位宽 *W/A/G*	准确率/%	准确率变化量/%
AlexNet	原版训练	32/32/32	58.00	-
	INQ	5/32/32	56.13*	-1.87
	Ultra-low	4/4/4	56.38	-1.62
	ALS-PoTQ	5/5/5	**57.22**	-0.78
ResNet18	原版训练	32/32/32	70.10	-
	INQ	5/32/32	68.98*	-1.12
	ShiftCNN	4/32/32	64.24*	-5.86
	AdderNet	32/32/32	67.00	-3.10
	DeepShift-Q	5/32/32	65.32	-4.78
	DeepShift-PS	5/32/32	65.34	-4.76
	S2FP8	8/8/8	**69.60**	-0.50
	Ultra-low	4/4/4	68.27	-1.83
	LUQ	4/4/4	69.00	-1.10
	ALS-PoTQ	5/5/5	69.52	-0.58
ResNet50	原版训练	32/32/32	76.32	-
	INQ	5/32/32	74.81*	-1.51
	ShiftCNN	4/32/32	72.58*	-3.74
	AdderNet	32/32/32	74.90	-1.42
	DeepShift-Q	5/32/32	70.73	-5.59
	DeepShift-PS	5/32/32	71.90	-4.42
	S2FP8	8/8/8	75.20	-1.12
	Ultra-low	4/4/4	74.01	-2.31
	LUQ	4/4/4	75.32	-1.00
	ALS-PoTQ	5/5/5	**75.36**	-0.96

注:“*”表示该方法只有从预训练模型微调的准确率结果,表中仅展示该结果,则不是从头训练的准确率结果。

除了图像分类任务之外，本工作还将 ALS-PoTQ 方法应用于 WMT En-De 数据集上的 Transformer-base 模型[42]以进行机器翻译。在训练中，本文所提方法不改变任何 TensorFlow 官方提供的超参数。与 Ultra-low[35]和 LUQ[21]对比，ALS-PoTQ 方法得到了最高的 BLEU 分数。相比全精度训练，ALS-PoTQ 方法的 BLEU 分数下降不到 0.3%，如表 4 所示。

综上所述，如图5所示，本工作提出方法在能耗和准确性方面都优于现有方法。

表 4　WMT En-De 任务上的 Tranformer 模型 BLEU 结果

网络	方法	位宽 *W/A/G*	*BLEU*/%	*BLEU* 变化量/%
Transformer	原版训练	32/32/32	27.5	-
	Ultra-low	4/4/4	25.4	−2.1
	LUQ	4/4/4	**27.2**	−0.3
	ALS-PoTQ	5/5/5	**27.2**	−0.3

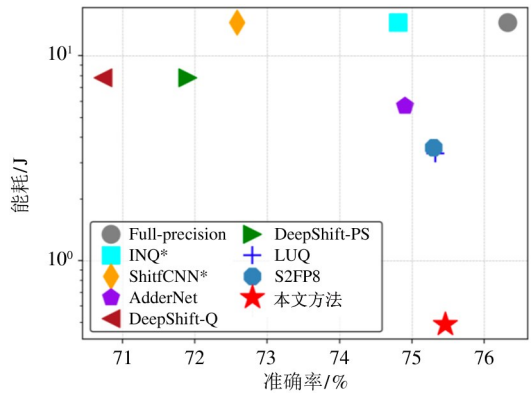


图 5　训练能耗-准确率联合比较

3.3　消融实验

本节对所提出的方法进行消融实验研究，包括自适应逐层缩放（ALS）、权重偏差校正（WBC）和参数化比率裁剪（PRC）。表 5 中 ResNet18 的训练精度结果证明了这些技术的效果。如果没有自适应逐层缩放，训练就会崩溃，准确率会下降到 0.0%。这是因为 PoT 量化的表示范围无法容纳数据范围，尤其是对于梯度来说。如果不进行权重偏差校正，则训练不稳定，有时会造成准确率的严重下降，这与 } 节中对 W 的分析是一致的。此外，参数化比率裁剪技术将 ResNet18 的精度提高了 1.3%。这些实验结果证明了本文方法中每一个技术都对整体有贡献。

表 5　算法各部分对准确率结果的影响

ALS	×	√	√	√	√
WBC	×	×	√	×	√
PRC	×	×	×	√	√
准确率/%	0.0	18.0/61.2	66.8	6.0	69.5

4　结　论

本工作研究了深度神经网络的低功耗训练问题。通过观察 W、A 和 G 的数据分布，提出了一种包括权重偏差校正（WBC）和参数化比率裁剪（PRC）技术的自适应逐层缩放 PoT 量化（ALS-PoTQ）方法和一种无乘法 MAC（MF-MAC）操作。在网络训练过程中，用 INT4 加法和 XOR 运算代替线性层中的所有 FP32 乘法，以减少训练能耗。在处理图像分类任务的 CNN 和处理机器翻译任务的 Transformer 网络上，该方法在准确率和 BLEU 下降不到 1.0% 的情况下，减少了训练过程中达 95.8% 的线性层能耗。

参考文献

[1] STRUBELL E, GANESH A, MCCALLUM A. Energy and policy considerations for deep learning in NLP[C]//Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. Florence, Italy: ACL, 2019: 3645-3650.

[2] WILSON L. Average household electricity consumption [EB/OL]. (2023-01-01) [2023-01-20]. https://shrinkthatfootprint.com/average-household-electricity-consumption: ShrinkThatFootprint.

[3] ANTHONY L F W, KANDING B, SELVAN R. Carbontracker: tracking and predicting the carbon footprint of training deep learning models[EB/OL]. (2020-07-06)[2023-01-20]. https://arxiv.org/pdf/2007.03051.pdf.

[4] THOMPSON N C, GREENEWALD K, LEE K, et al. The computational limits of deep learning[EB/OL]. (2022-07-27)[2023-01-20]. https://arxiv.org/pdf/2007.05558.pdf.

[5] JUSTUS D, BRENNAN J, BONNER S, et al. Predicting the computational cost of deep learning models[C]// 2018 IEEE International Conference on Big Data. Seattle, USA: IEEE, 2018:3873-3882.

[6] HAN S, MAO H, DALLY W J. Deep compression: compressing deep neural networks with pruning, trained quantization and Huffman coding[EB/OL]. (2016-02-15)[2023-01-20]. https://arxiv.org/pdf/1510.00149.pdf.

[7] 郭朝鹏. 王馨昕, 仲昭晋,等. 能耗优化的神经网络轻量化方法研究进展[J]. 计算机学报, 2023,46(1):85-102.

[8] FANG J, SHAFIEE A, ABDEL-AZIZ H, et al. Post-training piecewise linear quantization for deep neural networks[C]// Computer Vision-ECCV 2020: 16th European Conference. Glasgow, UK: ECCV, 2020:69-86.

[9] BANNER R, NAHSHAN Y, SOUDRY D. Post training 4-bit quantization of convolutional networks for rapid-deployment[C]// The 33rd Conference on Neural Information Processing Systems. Vancouver, Canada: NIPS, 2019:1-9.

[10] DAS D, MELLEMPUDI N, MUDIGERE D, et al. Mixed precision training of convolutional neural networks using integer operations[EB/OL]. (2018-02-23)[2023-01-20]. https://arxiv.org/pdf/1802.00930.pdf.

[11] KÖSTER U, WEBB T, WANG X, et al. Flexpoint: an adaptive numerical format for efficient training of deep neural networks[C]// Proceedings of the 31st International Conference on Neural Information Processing Systems. Long Beach, USA: ACM, 2017:1740-1750.

[12] CAMBIER L, BHIWANDIWALLA A, GONG T, et al. Shifted and squeezed 8-bit floating point format for low-precision training of deep neural networks[EB/OL]. (2020-01-16)[2023-01-20]. https://arxiv.org/pdf/2001.05674.pdf.

[13] SUN X, CHOI J, CHEN C Y, et al. Hybrid 8-bit floating point (HFP8) training and inference for deep neural networks[C]// The 33rd Conference on Neural Information Processing Systems. Vancouver, Canada: ACM, 2019:1-10.

[14] WANG N, CHOI J, BRAND D, et al. Training deep neural networks with 8-bit floating point numbers[C]// Proceedings of the 32nd International Conference on Neural Information Processing Systems. Montréal, Canada: ACM, 2018:7686-7695.

[15] ZHU F, GONG R, YU F, et al. Towards unified int8 training for convolutional neural network[C]// Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Seattle, USA: IEEE, 2020:1969-1979.

[16] CHEN H, WANG Y, XU C, et al. AdderNet: do we really need multiplications in deep learning?[C]// Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Seattle, USA: IEEE, 2020:1468-1477.

[17] ZHOU A, YAO A, GUO Y, et al. Incremental network quantization: towards lossless CNNs with low-precision weights[EB/OL]. (2017-08-25)[2023-01-20]. https://arxiv.org/pdf/1702.03044.pdf.

[18] GUDOVSKIY D A, RIGAZIO L. ShiftCNN: generalized low-precision architecture for inference of convolutional neural networks[EB/OL]. (2017-06-07)[2023-01-20]. https://arxiv.org/pdf/1706.02393.pdf.

[19] MIYASHITA D, LEE E H, MURMANN B. Convolutional neural networks using logarithmic data representation[EB/OL]. (2016-03-17)[2023-01-20]. https://arxiv.org/pdf/1603.01025.pdf.

[20] ELHOUSHI M, CHEN Z, SHAFIQ F, et al. Deepshift: towards multiplication-less neural networks[C]// Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Nashville, USA: IEEE, 2021:2359-2368.

[21] CHMIEL B, BANNER R, HOFFER E, et al. Logarithmic unbiased quantization: practical 4-bit training in deep learning[EB/OL]. (2022-06-06)[2023-01-20]. https://arxiv.org/pdf/2112.10769.pdf.

[22] YOU H, CHEN X, ZHANG Y, et al. ShiftAddNet: a hardware-inspired deep network[J]. Advances in Neural

Information Processing Systems, 2020,33:2771-2783.

[23] DENG J, DONG W, SOCHER R, et al. ImageNet: a large-scale hierarchical image database[C]//2009 IEEE Conference on Computer Vision and Pattern Recognition. Miami, USA: IEEE, 2009:248-255.

[24] KRIZHEVSKY A, SUTSKEVER I, HINTON G E. ImageNet classification with deep convolutional neural networks[J]. Communications of the ACM, 2017,60(6): 84-90.

[25] HE K, ZHANG X, REN S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas, USA: IEEE, 2016:770-778.

[26] VAN BAALEN M, LOUIZOS C, NAGEL M, et al. Bayesian bits: unifying quantization and pruning[J]. Advances in Neural Information Processing Systems, 2020, 33:5741-5752.

[27] CAI Z, VASCONCELOS N. Rethinking differentiable search for mixed-precision neural networks[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. Seattle, USA: IEEE, 2020: 2349-2358.

[28] COURBARIAUX M, BENGIO Y, DAVID J P. Binaryconnect: training deep neural networks with binary weights during propagations[C]// Proceedings of the 28th International Conference on Neural Information Processing Systems. Montreal, Canada: ACM, 2015:3123-3131.

[29] HUBARA I, COURBARIAUX M, SOUDRY D, et al. Binarized neural networks[C]//Proceedings of the 30th International Conference on Neural Information Processing Systems. Barcelona, Spain: ACM, 2016:4114-4122.

[30] KIM J, YOO K Y, KWAK N. Position-based scaled gradient for model quantization and pruning[J]. Advances in Neural Information Processing Systems, 2020, 33: 20415-20426.

[31] POURANSARI H, TU Z, TUZEL O. Least squares binary quantization of neural networks[C]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. Seattle, USA: IEEE, 2020:698-699.

[32] QIN H, GONG R, LIU X, et al. Binary neural networks: a survey[J]. Pattern Recognition, 2020,105:10728.

[33] TAO C, LIN R, CHEN Q, et al. Fat: learning low-bitwidth parametric representation via frequency-aware transformation[EB/OL]. (2021-02-20)[2023-01-20]. https://arxiv.org/pdf/2102.07444.pdf.

[34] ZHOU S, WU Y, NI Z, et al. Dorefa-Net: training low bitwidth convolutional neural networks with low bitwidth gradients[EB/OL]. (2018-02-02)[2023-01-20]. https://arxiv.org/pdf/1606.06160.pdf.

[35] SUN X, WANG N, CHEN C Y, et al. Ultra-low precision 4-bit training of deep neural networks[J]. Advances in Neural Information Processing Systems, 2020, 33: 1796-1807.

[36] LIU C, ZHANG X, ZHANG R, et al. Rethinking the importance of quantization bias, toward full low-bit training [J]. IEEE Transactions on Image Processing, 2022,31: 7006-7019.

[37] LI Y, DONG X, WANG W. Additive powers-of-two quantization: an efficient non-uniform discretization for neural networks[EB/OL]. (2020-02-02)[2023-01-20]. https://arxiv.org/pdf/1909.13144.pdf.

[38] CHOI J, WANG Z, VENKATARAMANI S, et al. Pact: parameterized clipping activation for quantized neural networks[EB/OL]. (2018-07-17)[2023-01-20]. https://arxiv.org/pdf/1805.06085.pdf.

[39] BENGIO Y, LÉONARD N, COURVILLE A. estimating or propagating gradients through stochastic neurons for conditional computation [EB/OL]. (2013-08-15)[2023-01-20]. https://arxiv.org/pdf/1308.3432.pdf.

[40] WANG Y, HUANG M, HAN K, et al. AdderNet and its minimalist hardware design for energy-efficient artificial intelligence[EB/OL]. (2021-02-03)[2023-01-20]. https://arxiv.org/pdf/2101.10015.pdf.

[41] ABADI M, BARHAM P, CHEN J, et al. TensorFlow: a system for large-scale machine learning[C]//Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation. Savannah, USA: USENIX, 2016:265-283.

[42] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is all you need[C]//Proceedings of the 31st International Conference on Neural Information Processing Systems. Long Beach, USA: ACM, 2017:6000-6010.

Multiplication-free neural network training based on adaptive PoT quantization

LIU Chang[*,**], ZHANG Rui[**,***], ZHI Tian[**,***]

([*]University of Chinese Academy of Sciences, Beijing 100049)

([**]Intelligent Processor Research Center, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190)

([***]State Key Lab of Processors, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190)

Abstract

The current deep neural network training process needs a large number of full-precision multiply-accumulate (MAC) operations, resulting in a situation that the energy consumption of the linear layers (including the convolutional layer and the fully connected layer) accounts for the vast majority of the overall energy consumption, reaching more than 90%. This work proposes an adaptive layer-wise scaling quantization training method, which can support the replacement of full-precision multiplication in all linear layers with 4-bit fixed-point addition and 1-bit XOR operation. The experimental results show that the above method is superior to the existing methods in terms of energy consumption and accuracy, and can reduce the energy consumption of linear layers by 95.8% in the training process. The convolutional neural networks on ImageNet and the Transformer networks on WMT En-De achieve less than 1% accuracy loss.

Key words: neural network, quantization, training acceleration, low energy consumption