高技术通讯 2022 年 第32 卷 第7 期:687-695

doi:10.3772/j.issn.1002-0470.2022.07.003

二进制张量分解法简化神经网络推理计算①

郝一帆②*** 杜子东③* 支 天*

(*中国科学院计算技术研究所智能处理器研究中心 北京 100190)

(**中国科学院大学 北京 100049)

摘 要 针对现有的简化神经网络推理计算方法面临模型精度下滑及重训练带来的额外开销问题,本文提出一种在比特级减少乘积累加运算(MAC)的乘加操作数的二进制张量分解法(IBTF)。该方法利用张量分解消除多个卷积核之间由于权值比特位重复导致的计算重复,并保持计算结果不变,即无需重训练。在比特级简化模型计算的 IBTF 算法与量化、稀疏等数据级简化方法正交,即可以协同使用,从而进一步减少 MAC 计算量。实验结果表明,在多个主流神经网络中,相较于量化与稀疏后的模型,IBTF 进一步使计算量减少了 3.32 倍,并且 IBTF 在不同卷积核大小、不同权值位宽及不同稀疏率的卷积运算中都发挥了显著的效果。

关键词 神经网络;二进制张量分解(IBTF);乘积累加运算(MAC)

0 引 言

近年来,智能驾驶、智慧医疗、大数据金融系统等多个领域逐步掀起了深度学习的热潮。作为深度学习中最具代表性的算法模型,人工神经网络是一种模仿动物神经网络行为特征,进行分布式并行信息处理的算法数学模型。人工神经网络中包含大量的参数以及乘积累加运算(multiply-accumulate,MAC),并且人工神经网络的计算量随参数量呈逐年显著上升趋势。1998 年,用于手写字符识别的神经网络规模小于 1×10^6 个参数[1]。2012 年,用于参加 ImageNet 竞赛的模型规模约 6×10^7 个参数[2]。2018 年,自然语言处理模型 BERT 含有约 3×10^8 个参数,以及数十亿的 MAC 运算。人工神经网络作为最主要的深度学习模型,其逐年增长的计算量为计算系统带来了巨大的负载,从而限制了深度学习在移动端设备的应用。所以,对神经网络模型进行简化是研究者们面临的一个重大挑战。

由于神经网络执行推理计算时涉及大量耗时耗能的 MAC 运算,所以减少 MAC 运算中的乘法和加法操作数是简化神经网络模型的核心思路。现有的方法根据对权值数据的操作可分为两大类,即微调法与重训练法。微调法是从 MAC 的计算流程入手,通过权值的低秩分解等手段配合对权值的重训练,以更少的乘加操作数完成原本的 MAC 计算并得到近似的结果,例如奇异值分解(singular-value decomposition,SVD)、正则多元分解(canonical polyadic,CP)和双聚类近似等[3]。重训练法是从模型的参数本身入手,通过稀疏、量化等手段,使权值在模型重训练过程中收敛到更多的零值和更低位宽的非零值,从而使参与计算的有效数据量降低,以达到减少操作数的目的。稀疏是减少神经网络模型中非零权值数量的方法,例如迭代剪枝法[4]将小于阈值的权值归零、权值聚类法[5]减少权值的非零取值、粗粒度稀疏法[6]减轻非零权值分布的不规则程度。量化是将模型中的非零权值用更低位宽的数据取代的

① 国家重点研发计划(2017YFB1003101,2018AAA0103300,2017YFA0700900)和国家自然科学基金(61532016,61732007)资助项目。

② 男,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:haoyifan@ict.ac.cn。

③ 通信作者,E-mail:duzidong@ict.ac.cn。

(收稿日期:2021-03-16)

方法。许多工作都已经表明,对于常用的神经网络模型,可以将权值数据量化为低位宽定点数,例如 8 比特,而不影响模型整体准确率[7-14]。此外,神经网络中不同层的权值数据可量化为不同位宽的定点数[15];对于某些特定的神经网络模型,甚至允许将权值量化为 2 比特或 1 比特[16-17]。

然而这些方法都涉及对权值的更改,无法避免执行神经网络的训练过程。所以在使用简化后的模型执行推理之前,系统需要承受模型反复训练带来的开销;在训练完毕之后,模型整体的准确度往往会有一定程度的下降。不仅如此,微调法往往面临超参难以选取的问题,例如当 SVD 等低秩分解方法的秩只能通过反复实验寻找最优解。对于重训练法,量化后的不同位宽的权值对中央处理器(central processing unit, CPU)或图形处理器(graphics processing unit, GPU)中固定位宽乘法器的硬件利用率不高,这也十分不利于神经网络模型推理计算的效率。

受 Stripes[18] 和 Laconic[19] 等工作中将权值中零比特位过滤的启发,不同于此前的数据级简化方法,即将单个权值数据作为最小优化单元,本文将权值中的每个比特看作独立的数据,发现多个卷积核之间由于权值比特位重复会导致一定的计算重复。基于此,本文提出一种在比特级降低神经网络中 MAC 计算的乘加操作数的二进制张量分解法(identical binary tensor factorization, IBTF)以消除上述计算重复。具体地,IBTF 将权值矩阵看作 0-1 二进制张量,并对该张量进行全等分解,而非近似。故 IBTF 不需要对权值的实际数值进行更改,也不会修改计算结果。进而 IBTF 既不会有训练模型带来的额外开销,也不会对模型整体准确度造成任何损失。此外,在比特级简化 MAC 计算是一种全新的优化维度,与权值低秩分解、稀疏、量化等数据级方法正交。所以 IBTF 可与之前的方法结合使用,从而进一步地减少神经网络模型中 MAC 运算的乘加操作数。实验结果表明,在多个主流神经网络中,相较于量化与稀疏后的模型,IBTF 进一步使计算量减少了 3.32 倍,并且 IBTF 在不同卷积核大小、不同权值位宽、不同稀疏率的卷积运算中都发挥了显著的效果。

1 神经网络中的 MAC 运算

本节主要介绍 MAC 运算在神经网络推理计算中的时间占比,并基于对 MAC 运算的观察,分析卷积中权值比特位重复带来的计算重复。

1.1 MAC 计算时间占比

在 NVIDIA V100 GPU 上,基于深度学习框架 Caffe[20],利用 time 功能测得多个常用神经网络模型推理计算中各阶段的执行时间,得到结果如图 1 所示。在 LeNet[1]、AlexNet[21] 和 VGG-16[22] 中,卷积层 CONV 和全连接层 FC 的执行时间占总推理时间的 80% 以上,其中在卷积规模较大的 VGG-16 中更是达到了 94.42%。在 ResNet-18[23] 和 ResNet-50[23] 中,卷积层和全连接层的时间占比略有下降,但仍超过 60%。下降的原因是这两个网络模型中的 BatchNorm 层和 Scale 层的逐元素乘加计算也占用了一定的时间。总之,由于卷积层与全连接层占神经网络推理计算时间的主要部分,并且在卷积层与全连接层中,除激活外全部是 MAC 运算,所以简化 MAC 运算有助于提升模型整体推理计算的性能。

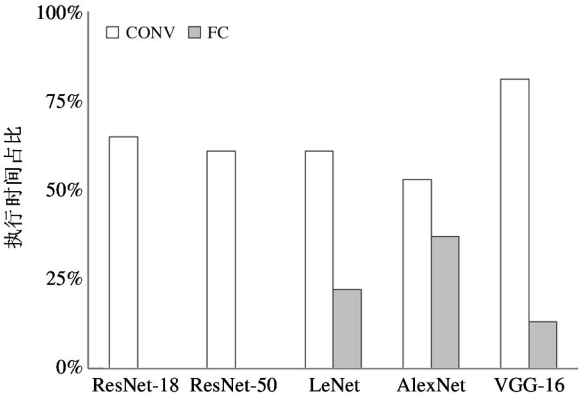

图 1 常用神经网络模型中卷积层 CONV 和全连接层 FC 的计算时间占比

1.2 MAC 中比特位重复带来的无效计算

在卷积层和全连接层的卷积运算中,单个卷积核在每个滑动窗口的运算是一个内积操作。如图 2 所示,以权值量化为 4 比特且大小为 2×2 的 1 个卷积核为例,4 个互不相等的非零权值分别为‘b1110,‘b1010,‘b1011,‘b0111,故不能通过稀疏(跳过零

权值的计算)或合并同类项(合并相等权值的乘加)的方法减少操作数,即该卷积运算在数据级已无任何可简化的空间。但是只要细化到比特级,即将权值数据的每个比特单独考虑,按照竖式加法将整个内积运算拆开后,可以观察到单个卷积核的 MAC 运算中含有两种无效计算:一种是由单个权值中的零比特位带来的含零加法,在图中由虚线框标出;另一种是由不同权值间的相同比特位带来的重复加法,在图中由实线框标出。于是,对于稀疏和合并同类项等数据级简化算法无效的该示例,比特级的简化可通过消除上述两种无效计算进一步减少 MAC 计算量。对于图中示例,原本需要 4 个乘法操作和 3 个加法操作的 MAC 运算,在去除上述两种无效计算后只需 8 个加法操作。这一简化效果十分显著,对于普通的 CPU 与 GPU,一般只含有 16/32/64 比特乘法器,故一个乘法的开销相当于数十个加法;即使对于含有 4 比特乘法器的专用硬件,由于 1 个 4 比特乘法由 4 个 4 比特数据移位累加完成,故 1 个乘法的开销约等于 3 个加法。所以在图 2 所示的例子中,在比特级去除无效计算后,该内积的计算开销(一个乘法的开销按上述方法根据数据位宽换算成多个加法)至少可节省为原计算开销的一半,甚至在普通 CPU 或 GPU 中节省数十倍。

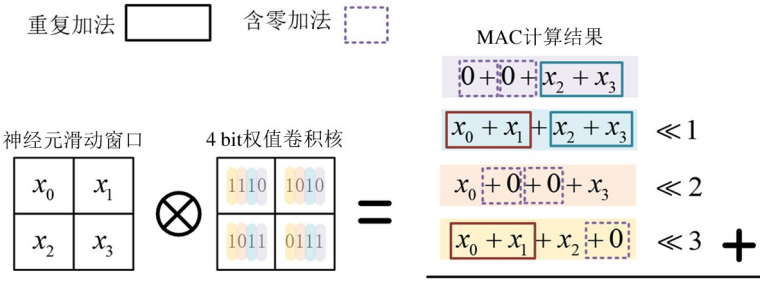

图 2　单个卷积核内的 MAC 运算中的 2 种无效计算示例

同理,多个卷积核在每个滑动窗口的运算是一个矩阵乘向量(matrix-vector multiply,MVM)操作。如图 3 所示,以权值量化为 2 比特且大小为 2×2 的 2 个卷积核 kernel0、kernel1 为例,分析其中的 MAC 计算。同样的,此时仍逐比特考虑权值数据,并且将全部卷积核参与的计算视作一个整体(即 MVM 操作),无效计算可由多个卷积核之间的权值比特位重复和零比特位带来。上述两种无效计算同样在图 3 中分别由实线框和虚线框标出。于是,比特级的简化可通过消除上述两种无效计算减少 MAC 计算量。对于图中示例,原本需要 8 个乘法操作和 6 个加法操作的 MAC 运算,在去除上述 2 种无效计算后只需 6 个加法操作。即使对于含有 2 比特乘法器的专用硬件,由于 1 个 2 比特乘法由 2 个 2 比特数据移位相加完成,故 1 个乘法的开销约等于 1 个加法。所以在图 3 所示的例子中,在比特级去除无效计算后,该内积的计算开销(乘法的开销按上述方法根据数据位宽换算成加法)至少可节省为原计算开销

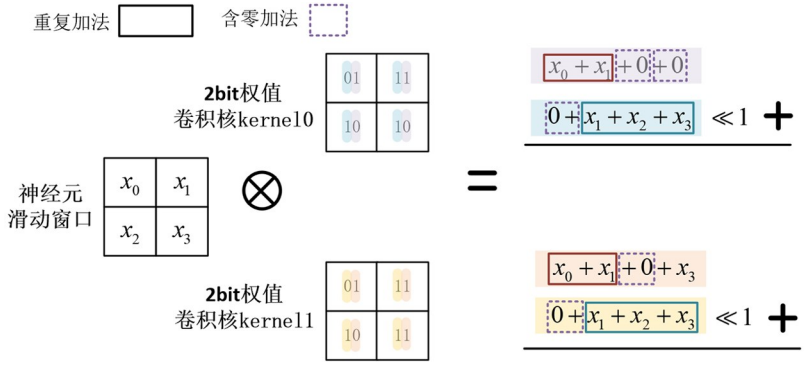

图 3　多个卷积核间的 MAC 运算中的 2 种无效计算示例

的 42.86%，甚至在普通 CPU 或 GPU 中节省数十倍。

在一般的神经网络模型中，一个卷积的计算规模远超图 2 和图 3 的示例。由于计算规模越大，分别由权值比特位重复和零比特位带来的 2 种无效计算越频繁出现，故比特级简化减少的 MAC 计算量越多，效果也越显著。于是，逐比特考虑权值数据并在比特级简化卷积计算是必要的。然而，虽然消除权值零比特位带来的无效计算可通过直接跳过零数据达成，但是提取权值比特位重复带来的无效计算的方法并不直观，原因在于权值比特位重复不仅发生在单个卷积核内，也发生在多个卷积核之间。这是本文提出 IBTF 算法的动机所在。

2　IBTF 算法

本节介绍一种在卷积运算中提取权值比特位重复的算法 IBTF，并说明其减少 MAC 运算中的乘加操作数的显著效果。

2.1　IBTF 算法步骤

以 4 比特神经网络中的某卷积层为例，设该层有 6 个尺寸为 16 ×4 ×4（16 个 channel，kernel 大小为 4 ×4）的卷积核。于是每个卷积核的单次滑动窗口内有 16 ×4 ×4 =256 个神经元。

第 1 步，将权值逐比特展开并等效为 0-1 二进制张量 $\boldsymbol{T}$。如 1.2 节所述，由于单个卷积核在每个滑动窗口位置的运算是一个内积，故可将神经元和单个卷积核看作长度为 256 的向量。进一步地，将 6 个 4 比特卷积核逐比特展开为 0-1 二进制权值张量 $\boldsymbol{T}$，张量 $\boldsymbol{T}$ 的维度为 256 ×6 ×4。

第 2 步，切分 0-1 二进制权值张量 $\boldsymbol{T}$。在此例中，以 3 比特为切分单元，挖掘权值张量 $\boldsymbol{T}$ 中非零比特重复带来的无效计算。如图 4 所示，首先将 $\boldsymbol{T}$ 划分为 8 个大小为 256 ×3 的 0-1 二进制矩阵 $\boldsymbol{T}_1$，$\boldsymbol{T}_2$，…，$\boldsymbol{T}_8$，图 4（a）显示的是拆分张量 $\boldsymbol{T}$ 的 3D 示意图，图 4（b）显示的是拆分俯视图。将长度为 256 的神经元向量记作 $\boldsymbol{x}$，于是整个卷积运算的结果满足式（1）。

$$\text{MAC result} = \boldsymbol{x}^{\mathrm{T}} \cdot \boldsymbol{T} = \sum_{i=1}^{6} \boldsymbol{x}^{\mathrm{T}} \cdot \boldsymbol{T}_i \cdot \begin{pmatrix} 2^2 \\ 2^1 \\ 2^0 \end{pmatrix} + \sum_{i=7}^{8} \boldsymbol{x}^{\mathrm{T}} \cdot \boldsymbol{T}_i \cdot 2^3 \tag{1}$$

$$= \sum_{i=1}^{6} \boldsymbol{x}^{\mathrm{T}} \cdot \boldsymbol{B}_i^{\mathrm{row}} \cdot \boldsymbol{K} \cdot \begin{pmatrix} 2^2 \\ 2^1 \\ 2^0 \end{pmatrix} + \sum_{i=7}^{8} \boldsymbol{x}^{\mathrm{T}} \cdot \boldsymbol{B}_i^{\mathrm{row}} \cdot \boldsymbol{K} \cdot 2^3 \tag{2}$$

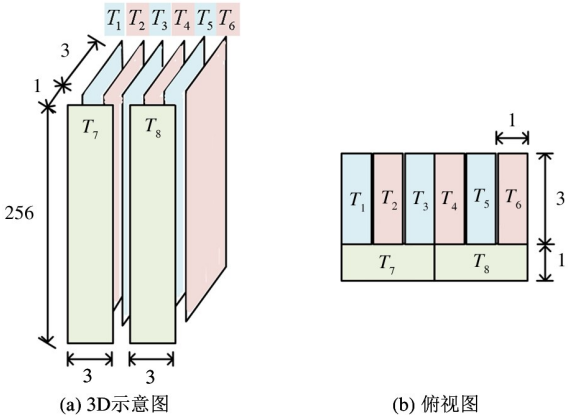

图 4　IBTF 算法中的二进制权值张量切分

第 3 步，对划分权值张量 $\boldsymbol{T}$ 后得到的二进制权值矩阵分别进行全等矩阵分解。将每个 $\boldsymbol{T}_i$（i = 1，2，…，8）分解成一个稀疏的二进制矩阵 $\boldsymbol{B}_i^{\mathrm{row}}$ 和一个非稀疏的二进制矩阵 $\boldsymbol{K}$。具体地，如图 5 所示，对于大小为 256 ×3 的二进制矩阵 $\boldsymbol{T}_i$，将其每行的 3 比特看作一个切分单元，则该切分单元共有 $2^3 - 1 = 7$ 种非零取值。矩阵 $\boldsymbol{K}$ 的每一行对应一种 3 比特切分单元的非零取值，则 $\boldsymbol{K}$ 的大小固定为 7 ×3。由于矩阵 $\boldsymbol{T}_i$ 的每一行的取值可从矩阵 $\boldsymbol{K}$ 的某一行中选出，或是取零向量，故可得到大小为 1024 ×7 的稀疏

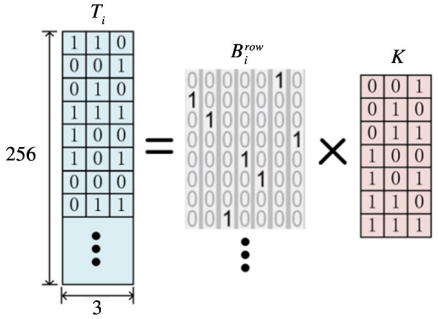

图 5　IBTF 算法中单个二进制权值矩阵的全等分解

矩阵 $\boldsymbol{B}_i^{\text{row}}$,使得 $\boldsymbol{T}_i = \boldsymbol{B}_i^{\text{row}} \cdot \boldsymbol{K}$。其中,$\boldsymbol{B}_i^{\text{row}}$ 的每一行是一个 one-hot 向量或零向量,对应 $\boldsymbol{T}_i$ 中一行的取值。分解完毕后,整个卷积运算的结果可写成式(2)。由于神经元与 2 的次幂相乘可以由移位操作代替,与 0/1 相乘可以由 AND/OR 操作代替,所以由式(2)计算的卷积结果不需要任何乘法。至此,IBTF 算法完成。

2.2 IBTF 效果分析

IBTF 处理后的卷积运算并不改变运算结果,且不需要任何乘法操作,所以只需分析加法操作数就能确定 IBTF 处理后的计算量。设某卷积运算中,卷积核的大小为 N,个数为 M,权值数据的位宽为 p。于是对于该卷积操作的一个滑动窗口中,神经元可展开成长度为 N 的向量 $\boldsymbol{x}$,卷积核可按权值逐比特展开成大小为 $N \times M \times p$ 的二进制张量 $\boldsymbol{T}$。设对 $\boldsymbol{T}$ 的切分单元大小为 a,于是 $\boldsymbol{T}$ 可被切分为 $\frac{Mp}{a}$ 个二进制权值矩阵 $\boldsymbol{T}_i$,$i = 1,2,\cdots,\frac{Mp}{a}$。而后每个 $\boldsymbol{T}_i$ 分解得到矩阵 $\boldsymbol{B}_i^{\text{row}}$ 和 $\boldsymbol{K}$,两个矩阵的大小分别为 $\dim(\boldsymbol{B}_i^{\text{row}}) = N \times (2^a - 1)$ 和 $\dim(\boldsymbol{K}) = (2^a - 1) \times \boldsymbol{K}$。假设该卷积运算中权值的稀疏率为 $\lambda(0 \leqslant \lambda < 1)$,于是 $\boldsymbol{B}_i^{\text{row}}$ 中含有 $(1 - \lambda)N$ 个非零行向量。此时对于式(2),神经元向量 $\boldsymbol{x}$ 与 $\boldsymbol{B}_i^{\text{row}}$ 相乘需要 $(1 - \lambda)N - 2^a$ 个加法,此后再与 $\boldsymbol{K}$ 相乘需要约 2^{a+1} 个加法。于是根据式(2)计算单次滑动窗口内的卷积结果所需的总加法操作数最多为

$$IBTF\_op = [(1 - \lambda)N + 2^a] \cdot \frac{Mp}{a} \tag{3}$$

对于 2.1 节中的示例,根据卷积规模设置可知 $N = 256$,$M = 6$,$p = 4$,$a = 3$。假设 $\lambda = 0$,即所有权值都参与 MAC 运算,代入式(3)可得,经 IBTF 处理后,加法操作数最多为 2112 个,且无任何乘法操作。而原本的卷积运算需要 $(1 - \lambda)NM = 1536$ 个乘法操作与 1536 个加法操作。为了更清晰地对比 IBTF 简化前后的运算量,可根据数据位宽将一个乘法等效为多个加法,并将等效后的总操作数记作等效 MAC 操作数 Eq_MAC_op。如 1.2 节所述,在实际硬件中,p 比特数据的乘法由多次移位操作与至少 $p - 1$ 次加法完成,故可将 1 个 p 比特乘法近似等效为 $p - 1$ 个加法。于是,原本卷积运算的等效 MAC 操作数为 $1536 + 1536 \times (4 - 1) = 6144$,是 IBTF 简化后的 2.91 倍。值得注意的是,在后续的实验中使用 Eq_MAC_op 度量运算量时,并不会将零权值参与的加法与乘法统计在内,只是在本示例中已经假设权值的稀疏率 λ 为 0。综上,IBTF 通过提取权值比特位重复带来的无效计算,大幅减少了 MAC 运算量,并且保持数值计算结果不变,从而在提升系统能效的同时不会对模型精度产生任何影响。不仅如此,由于 IBTF 简化后的卷积计算不含乘法,所以也不存在不同位宽的权值对固定位宽的乘法器利用率低下的问题。

此外,在实际使用 IBTF 执行神经网络推理计算时,由于权值已训练完毕并固定,所以 IBTF 的运算流也是固定的。于是可以通过直接存储矩阵 $\boldsymbol{B}_i^{\text{row}}$ 和 $\boldsymbol{K}$ 的方式代替存储权值。下面分析这种权值存储方式的开销。对于矩阵 $\boldsymbol{K}$,由于 $\boldsymbol{K}$ 的取值只与切分单元的大小 a 有关,所以只需存储 a 即可,即矩阵 $\boldsymbol{K}$ 几乎不占用任何存储。对于矩阵 $\boldsymbol{B}_i^{\text{row}}$,由于 $\boldsymbol{B}_i^{\text{row}}$ 是一个每行为全 0 或只有一个 1 的极度稀疏矩阵,所以可按如下方案存储:对于非零权值稠密的卷积,即 $\boldsymbol{B}_i^{\text{row}}$ 中全 0 的行向量较少时,只需存储记录 $\boldsymbol{B}_i^{\text{row}}$ 每行中 1 的位置将其按行存储;对于权值稀疏的卷积,即 $\boldsymbol{B}_i^{\text{row}}$ 中全 0 的行向量较多时,可通过游程编码按列存储 $\boldsymbol{B}_i^{\text{row}}$。综上,使用 IBTF 算法几乎不会带来额外的权值访存开销。

2.3 IBTF 最优切分

如式(3)所示,对于给定规模的卷积计算,经 IBTF 简化后所需的加法操作数至多为 $[(1 - \lambda)N + 2^a] \cdot \frac{Mp}{a}$,其中 N 为卷积核大小,M 为卷积核数量,p 为权值数据位宽,λ 为权值稀疏率,a 为切分单元大小。由此可知,对于固定权值规模与数值的卷积计算,上述操作数仅与切分单元大小 a 的选取相关。于是,求使 IBTF 操作数最少的切分单元问题可表达为式(4)。$IBTF\_op(a)$ 对 a 求导可得:当 a 满足式(5)时,$IBTF\_op(a = a^*)$ 取最小值。综上,在给定卷积运算规模和权值数据位宽时,可以根据式(5)直接计算得到最优 IBTF 拆分单元大小 a^*。

$$\min_{a\in Z^+}\left\{IBTF\_op(a)\mid IBTF\_op(a)=\left[(1-\lambda)N+2^a\right]\cdot\frac{Mp}{a},\ 2\leqslant a\leqslant Mp\right\} \tag{4}$$

$$\begin{cases}a^*=Mp & (Mp\cdot\ln 2-1)\cdot 2^{Mp}\leqslant(1-\lambda)N\\ (a^*\ln 2-1)\cdot 2^{a^*}\approx(1-\lambda)N & \text{其他}\end{cases} \tag{5}$$

对于 2.1 节的示例，根据式(5)求得最优切分单元的大小为 $a^*=6$，此时对二进制权值张量的划分如图 6 所示，并求得 IBTF 的加法操作数至多为 1280。相较于 2.1 节示例中 $a=3$ 的切分方式，最优切分 IBTF 的加法操作数进一步减少了 39.4%。更具体地，图 7 显示了该示例中取不同切分单元大小的操作数对比。从图中可以看出，切分单元大小的选取显著影响 IBTF 的效果，于是根据式(5)求 IBTF 最优切分单元大小是必要且正确的。

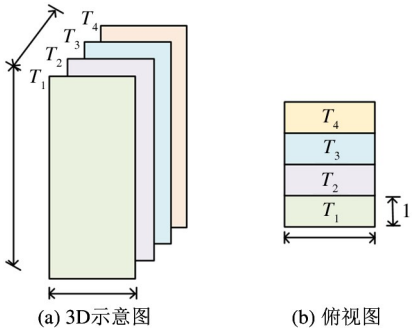

图 6　IBTF 算法示例二进制权值张量最优切分

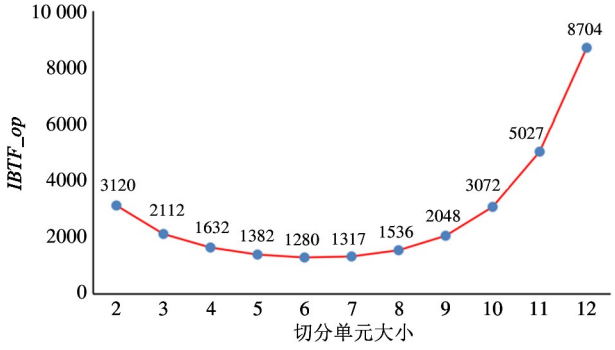

图 7　IBTF 算法示例操作数与切分单元大小

3　实　验

3.1　实验方法

在 NVIDIA V100 GPU 上，对于不同规模、不同位宽、不同权值稀疏度的单个卷积层以及常用神经网络模型，分别执行推理计算。其中在卷积运算中，IBTF 算法对 MAC 计算的简化处理类似一个即插即用(plug-and-play，PnP)模块。实验分别统计了 IBTF 简化后模型与原始模型的乘加操作数，以体现 IBTF 对 MAC 计算量的简化效果。

3.2　常用神经网络中的总操作数

实验使用了一组常用的神经网络模型为测试用例集，包括 AlexNet[21]、VGG-11[22]、VGG-16[22]、ResNet-18[23]、ResNet-50[23] 和 GoogleNet[24]，且它们执行的推理计算是在数据集 ILSVRC12[25] 上进行图像分类。在模型正确率下降 1% 以内的限制下，这些模型都可以将权值量化为 8 比特以内的定点数。具体地，量化后模型的对应权值位宽[9,26-27] 如表 1 所示。

如图 8 所示，实验比较了稀疏(*Spar.*)、量化 + 稀疏(*Quan.* + *Spar.*)和量化 + 稀疏 + IBTF(*IBTF*)处理后模型的相对计算量。其中模型的 MAC 计算量可由 2.2 节中所述的等效 MAC 操作数 *Eq_MAC_op* 统一度量，即将每个 p 比特乘法等效为 $p-1$ 个加法后的加法操作数。为了使图 8 更清晰地显示比较结果，将量化 + 稀疏 + IBTF 处理后每个模型的 *Eq_MAC_op* 都记为单位 1(实际上不同模型的计算量并不相同)，故图中纵坐标表示的是上述 3 种方法相较于 *IBTF* 的相对 *Eq_MAC_op*。实验结果显示，相较于 *Spar.*，*IBTF* 平均使计算量减少了 11.54 倍；相较于 *Quan.* + *Spar.*，IBTF 平均使计算量减少了 3.32 倍。尤其是对于权值位宽较大(7 比特)

表 1　常见神经网络模型及量化后权值位宽

网络模型	AlexNet	VGG-11	VGG-16	GoogleNet	ResNet-18	ResNet-50
权值位宽	4	7	6	3	4	2

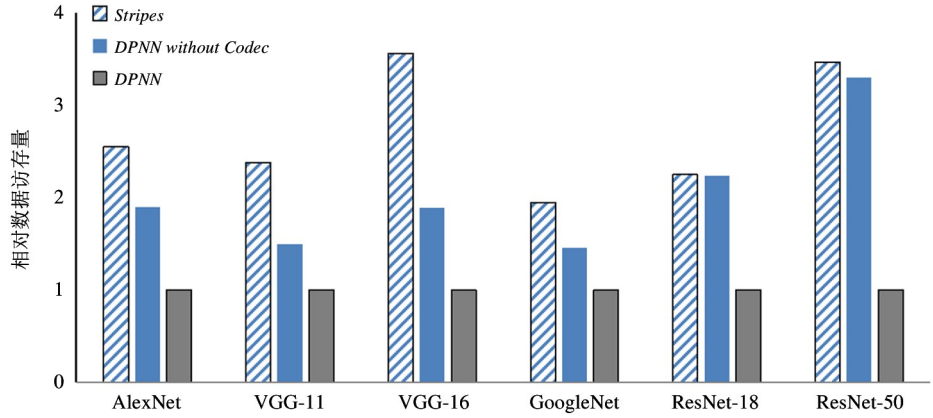

图 8　常见神经网络模型在不同简化方法下的相对计算量

的 VGG-11 网络,相较于 *Quan.* + *Spar.*, *IBTF* 平均使计算量减少了 5.22 倍。综上,基于量化与稀疏简化后的模型,保持计算结果不变的 IBTF 算法仍可进一步大幅减少 MAC 计算量。

3.3　单个卷积层中的操作数

在上一节使用整个神经网络模型的实验中,不同模型的卷积核大小、权值位宽与稀疏率各不相同,于是 IBTF 的简化效果也不同。为了更细致地验证 IBTF 的简化效果,下面的实验将在单个卷积层中分析卷积核大小、权值位宽与稀疏率对 IBTF 效果的影响。

$$w = \begin{cases} 0 & u = uniform(0,1) \leqslant \lambda \\ \text{random\_select}\{Z^{+} \cap [1, 2^{p}]\} & u = uniform(0,1) > \lambda \end{cases} \tag{6}$$

卷积层的构造规则为固定输出特征图的大小为 16,固定卷积核数量为 $M = 4$,卷积核大小分别取大卷积核 $N = 1024$ 和小卷积核 $N = 32$ 两种,权值数据位宽取值范围为 $1 \leqslant p \leqslant 8$,符合主流神经网络模型量化后的权值位宽,当稀疏率设为 $\lambda(0 \leqslant \lambda < 1)$ 时,p 比特权值 w 按式(6)随机生成,即以 λ 的概率取 0,以 $1 - \lambda$ 的概率在 $2^{p} - 1$ 个非零取值中等概率随机选取。

对于权值稠密的卷积计算,假设所有权值参与计算,即 $\lambda = 0$,实验得到不同卷积核大小、不同权值位宽的卷积操作数统计如表 2 所示。表中 *Eq_MAC_op* 为 2.2 节中所述的等效 MAC 操作数,表示 IBTF 简化前的模型计算量;*IBTF_op* 由式(5)得到最优切分后代入式(3)求得,表示 IBTF 简化后的模型计算量;*reduce rate* = *Eq_MAC_op* / *IBTF_op*,表示 IBTF 将计算量减少的倍数。从该表中数据可得到如下 2 个基本趋势:大卷积核情况下的 IBTF 算法效果优于小卷积核;高位宽权值情况下的 IBTF 算法效果优于低位宽权值。上述趋势的原因在于:当卷积核较大或权值位宽较高时,权值比特位重复更

表 2　不同卷积核大小、不同权值位宽的卷积操作数统计

λ	M	N	p	*Eq_MAC_op*	*IBTF_op*	*reduce rate*
0	4	1024	1	6.55E+04	1.66E+04	3.94
0	4	1024	2	1.31E+05	2.05E+04	6.40
0	4	1024	4	2.62E+05	4.10E+04	6.40
0	4	1024	8	5.24E+05	8.19E+04	6.40
0	4	64	1	4.10E+03	1.28E+03	3.20
0	4	64	2	8.19E+03	2.56E+03	3.20
0	4	64	4	1.64E+04	5.06E+03	3.24
0	4	64	8	3.28E+04	9.98E+03	3.28

多且重复粒度更大,于是 IBTF 消除的随之而来的无效计算更多且二进制权值张量的切分粒度更大,最终导致 IBTF 算法的效果更显著。

对于权值稀疏的卷积计算,固定卷积规模为输出特征图的大小为 16;卷积核数量 $M = 4$; 卷积核大小 $N = 1024$; 权值位宽 $p = 4$。实验得到不同权值稀疏率的卷积操作数统计如表 3 所示。从该表中数据可知,权值稀疏率越低,IBTF 效果越好。同理,该趋势原因为权值稀疏率低意味着非零比特位多,于是权值比特位重复更多且重复粒度更大。此外,实验结果表明,当权值稀疏率在 80% ~95% 时($0.8 \leqslant \lambda \leqslant 0.95$),即符合常见神经网络模型剪枝后的稀疏率取值,*reduce rate* 取值为 2.42 ~3.89。这一结果与 3.2 节的常见神经网络模型中 IBTF 平均使计算量进一步减少了 3.32 倍相符。

表 3 不同权值稀疏率的卷积操作数统计

λ	M	N	p	*Eq _ MAC _ op*	*IBTF _ op*	*reduce rate*
1	4	1024	4	2.62E +05	4.10E +04	6.40
0.9	4	1024	4	2.36E +05	4.06E +04	5.82
0.7	4	1024	4	1.84E +05	3.54E +04	5.19
0.5	4	1024	4	1.31E +05	2.74E +04	4.79
0.3	4	1024	4	7.86E +04	1.95E +04	4.03
0.2	4	1024	4	5.24E +04	1.35E +04	3.89
0.15	4	1024	4	3.93E +04	1.20E +04	3.29
0.1	4	1024	4	2.62E +04	8.11E +03	3.23
0.05	4	1024	4	1.31E +04	5.41E +03	2.42

4 结 论

本文从一种新的简化神经网络中卷积运算的角度出发,即消除权值比特位重复带来的计算重复,提出一种在比特级减少 MAC 运算的乘加操作数的方法 IBTF。它的主要优点有:(1)保持计算结果不变,不影响模型正确率;(2)不需要重训练;(3)与量化、稀疏等方法正交。效果上,在多个主流神经网络中,相较于量化与稀疏后的模型,IBTF 进一步使计算量减少了 3.32 倍;并且 IBTF 在不同规模、不同位宽、不同权值稀疏度的卷积运算中都发挥了显著的效果。未来将根据 IBTF 算法设计硬件中的卷积运算单元,最终形成一个应用 IBTF 的高能效神经网络加速器架构,从而将 IBTF 实际应用于深度学习计算系统的加速中。

参考文献

[1] LÉCUN Y, BOTTOU L, BENGIO Y, et al. Gradient-based learning applied to document recognition[J]. *Proceedings of the IEEE*, 1998, 86(11):2278-2324

[2] KRIZHEVSKY A, SUTSKEVER I, HINTON G, et al. ImageNet classification with deep convolutional neural networks[J]. *Advances in Neural Information Processing Systems*, 2012, 25: 1097-1105

[3] EMILY L D, WOJCIECH Z, JOAN B, et al. Exploiting linear structure within convolutional networks for efficient evaluation[C]// Proceedings of the 27th International Conference on Neural Information Processing Systems, Montréal, Canada, 2014: 1269-1277

[4] DENG W, YIN W, YIN Z. Group sparse optimization by alternating direction method[C]// International Society for Optics and Photonics, San Diego, USA, 2013, 8858-8867

[5] HAN S, MAO H, DALLY W J, et al. Deep compression: compressing deep neural networks with pruning, trained quantization and huffman coding[C]// International Conference on Learning Representations, San Juan, Puerto Rico, 2016: 1153-1162

[6] HE Y, ZHANG X, SUN J. Channel pruning for accelerating very deep neural networks[C]// Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 2017: 1389-1397

[7] 孙建辉, 方向忠. 卷积神经网络的混合精度量化技术研究[J]. 信息技术, 2020, 343(6):74-77

[8] 陈昀, 蔡晓东, 梁晓曦,等. 权重量化的深度神经网络模型压缩算法[J]. 西安电子科技大学学报, 2019, 46(2):132-138

[9] MIGACZ S. 8-比特 inference with TensorRT[C]// GPU Technology Conference, San Jose, USA, 2017: 5

[10] MISHRA A, NURVITADHI E, COOK J, et al. WRPN: wide reduced-precision networks[C]// International Con-

ference on Learning Representations, Vancouver, Canada, 2018: 1342-1351
[11] PARK E, AHN J, YOO S. Weighted-entropy-based quantization for deep neural networks[C]// Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Venice, Italy, 2017: 5456-5464
[12] PARK E, YOO S, VAJDA P. Value-aware quantization for training and inference of neural networks[C]// Proceedings of the European Conference on Computer Vision, Munich, Germany, 2018: 580-595
[13] 尹文枫, 梁玲燕, 彭慧民,等. 卷积神经网络压缩与加速技术研究进展[J]. 计算机系统应用, 2020, 29(9):16-25
[14] ZHOU S, WU Y, NI Z, et al. Dorefa-net: training low 比特 width convolutional neural networks with low bitwidth gradients[EB/OL]. https://arXiv.org/pdf/1606.06160.pdf:arXiv, (2018-02-02),[2021-03-10]
[15] JUDD P, ALBERICIO J, HETHERINGTON T, et al. Stripes: bit-serial deep neural network computing[C]// 2016 49th Annual IEEE/ACM International Symposium on Microarchitecture, Durgapur, India, 2016: 1-12
[16] RASTEGARI M, ORDONEZ V, REDMON J, et al. Xnor-Net: ImageNet classification using binary convolutional neural networks[C]// European Conference on Computer Vision, Amsterdam, Netherlands, 2016:525-542
[17] COURBARIAUX M, HUBARA I, SOUDRY D, et al. Binarized neural networks: training deep neural networks with weights and activations constrained to +1or -1[C]// Conference on Neural Information Processing Systems, Barcelona, Spain, 2016: 1432-1440
[18] ALBERICIO J, DELMÁS A, JUDD P, et al. Bit-pragmatic deep neural network computing[C]// Proceedings of the 50th Annual IEEE/ACM International Symposium on Microarchitecture, Cambridge, USA, 2017: 382-394
[19] SHARIFY S, LASCORZ A D, MAHMOUD M, et al. Laconic deep learning inference acceleration[C]// 2019 ACM/IEEE 46th Annual International Symposium on Computer Architecture, Phoenix, USA, 2019: 304-317
[20] JIA Y, SHELHAMER E, DONAHUE J, et al. Caffe: convolutional architecture for fast feature embedding[C]// Proceedings of the 22nd ACM International Conference on Multimedia, Orlando, USA, 2014: 632-641
[21] KRIZHEVSKY A, SUTSKEVER I, HINTON G. ImageNet classification with deep convolutional neural networks[C]// Conference on Neural Information Processing Systems, Lake Tahoe, USA, 2012: 876-884
[22] SIMONYAN K, ZISSERMAN A. Very deep convolutional networks for large-scale image recognition[C]// Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, USA, 2014: 987-996
[23] 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, 2016: 770-778
[24] SZEGEDY C, LIU W, JIA Y, et al. Going deeper with convolutions[C]// Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, USA, 2015: 1-9
[25] RUSSAKOVSKY O, DENG J, SU H, et al. ImageNet large scale visual recognition challenge[J]. *International Journal of Computer Vision*, 2015, 115(3): 211-252
[26] GYSEL P, PIMENTEL J, MOTAMEDI M, et al. Ristretto: a framework for empirical study of resource-efficient inference in convolutional neural networks[J]. *IEEE Transactions on Neural Networks and Learning Systems*, 2018, 29(11): 5784-5789
[27] YANG J, SHEN X, XING J, et al. Quantization networks[C]// Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Los Angeles, USA, 2019: 7308-7316

Simplifying inference computation of neural networks by identical-binary-tensor-factorization

HAO Yifan[*,**], DU Zidong[*], ZHI Tian[*]
([*]Intelligent Processor Research Center, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190)
([**]University of Chinese Academy of Sciences, Beijing 100049)

Abstract

Existing methods to simplify neural network inference often face the problem of model accuracy degradation and additional overhead caused by retraining. In this work, an identical binary tensor factorization (IBTF) algorithm is proposed for the further reduction of multiply-accumulate (MAC) operands under bit-level. IBTF uses tensor decomposition to extract the computation repetition between multiple convolution kernels due to the bit repetition of synapses, and keep computational results identical without retraining. Moreover, IBTF, which simplifies models under bit-level, is orthogonal to these data-level simplification methods such as quantization and sparsity, so they can be used synergistically to further reduce MAC operands. The experimental results show that, in several mainstream neural networks, compared with models after quantization and sparsity, IBTF further reduces 3.32 times MAC operands. In addition, IBTF plays a significant role in convolution layers with different sizes, bit-widths and sparsity rates.

Key words: neural network, identical binary tensor factorization (IBTF), multiply-accumulate (MAC)