

引用格式: 沈林耀, 王琴, 蒋剑飞, 等. 一种面向忆阻器加速器的神经网络模型压缩框架[J]. 微电子学与计算机, 2021, 38(8): 20-27. [SHEN L Y, WANG Q, JIANG J F, et al. Towards ReRAM-based Accelerator: An Energy-efficient NN Model Compression Framework[J]. Microelectronics & Computer, 2021, 38(8): 20-27.]

一种面向忆阻器加速器的神经网络模型压缩框架

沈林耀, 王 琴, 蒋剑飞, 景乃锋

(上海交通大学 电子信息与电气工程学院, 上海 200240)

摘 要: 当前基于忆阻器的神经网络加速器存在的资源需求高、系统功耗大等问题, 提出了一种包含剪枝及量化算法在内的神经网络模型压缩框架. 根据忆阻器阵列紧密耦合的特点, 设计了一种忆阻器阵列感知的规则化增量剪枝算法, 在保证模型准确度的条件下实现了硬件资源的节省; 针对忆阻器加速器系统中 ADC 单元和忆阻器阵列功耗占比过大等问题, 设计了一种二的幂次量化算法以降低加速器系统中 ADC 的精度需求以及计算阵列中低阻值忆阻器器件个数, 实现系统功耗的降低. 实验结果表明: 提出的神经网络模型压缩框架在忆阻器加速器部署网络时可取得 17.2~30.7 倍的能效提升以及 4.3~9.3 倍的加速比, 模型的精度损失维持在 1% 左右.

关键词: 忆阻器加速器; 神经网络; 量化; 剪枝

中图分类号: TP389.1

文献标识码: A

文章编号: 1000-7180(2021)08-0020-08

Towards ReRAM-based Accelerator: An Energy-efficient NN Model Compression Framework

SHEN Linyao, WANG Qin, JIANG Jianfei, JING Naifeng

(School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China)

Abstract: The current ReRAM-based NN accelerator have many problems such as high hardware resource demand and high power consumption. An energy-efficient model compression framework consisting of pruning and quantization algorithms is proposed. According to the tightly coupled crossbar structure and unstructured sparsity, a crossbar-aware incremental structured pruning algorithm is designed to achieve higher sparsity and accuracy. A power of two quantization method is designed to reduce ADC resolution requirements and the number of low resistance states (LRS) ReRAM cells in crossbars to improve the energy efficiency. Experimental results show that the proposed model compression framework can achieve 17.2-30.7x energy efficiency and 4.3-9.3x speedup, compared with ReRAM-based accelerators for dense NN with about 1% accuracy loss.

Key words: ReRAM-based Accelerator; Neural networks; Quantization; Pruning

1 引言

近年来, 神经网络算法在包括图像分类、目标识别及自然语言处理在内的多个研究领域取得了巨大的成就. 然而随着网络规模的不断增大, 神经网络模型对计算平台的算力和存储提出了极高的要求, 因此人们开始探究基于存内计算的新型计算架构.

忆阻器由蔡邵棠于 1971 年提出^[1], 并于 2008 年由惠普实验室成功研制^[2]. 忆阻器的阻变特性和记忆特性打破了传统冯诺依曼架构下存算分离的桎梏, 为存算一体架构提供了新的解决思路. 研究表明, 以忆阻器为基本单元构建的交叉阵列 (Crossbar) 结构可自然高效的执行乘累加运算进而加速神经网络推理^[3]. 因此基于忆阻器的神经网络加速器

引发了学界及工业界的研究热潮。

尽管忆阻器加速器在实现神经网络推理时拥有巨大优势,但该类加速器仍存在着硬件资源需求大、系统功耗高等问题。经典的 ISAAC 忆阻器加速器在部署 VGG-16 模型时,仅最大层网络就需要 523 个 Tile 级别的计算单元^[4]。同时高精度 ADC 单元以及计算阵列带来的功耗消耗也占据系统总功耗的 70% 以上。因此针对忆阻器加速器的特点进行网络裁剪,同时设法降低 ADC 单元精度需求以及阵列功耗是必要的。

本文针对当前忆阻器加速器存在的上述两类缺点,提出了一种基于阵列感知剪枝算法和二的幂次量化算法的神经网络模型压缩框架,实现了硬件资源的节省和加速器功耗的降低。主要研究工作概括如下:

1) 在硬件资源节省层面,提出阵列感知(cross-bar-aware)的规则增量剪枝算法以提高模型稀疏度,在同样算力情况下网络模型推理得以大大加速。该算法考虑到忆阻器阵列列紧密耦合的特点,摒弃了已有的随机稀疏的裁剪算法,同时针对当前已有的面向忆阻器加速器的剪枝算法粒度过粗,模型稀疏度和准确度过低等问题,实现了细粒度规则剪枝。

2) 在功耗降低层面,本文通过二的幂次量化算法,将模型非 0 权重约束成 2 的幂次形式,使 ADC 精度需求降低,同时减少阵列中低阻值忆阻器器件的比例进而降低阵列功耗。

2 忆阻器加速器架构

2.1 忆阻器基本概念

如图 1 所示,忆阻器是一种两端器件,整个结构由上下电极及夹在中间的阻变材料层构成。与传统的数字电路用电平高低表示逻辑值“0”与“1”不同,忆阻器用不同阻值状态表示不同逻辑值,例如以高阻值状态(HRS)代表逻辑值“0”,低阻值状态(LRS)代表逻辑值“1”。通过改变其两端电压方向和大小,忆阻器器件完成阻值状态的改变,实现特定布尔运算或点积运算。

2.2 基于忆阻器的神经网络计算原理

利用忆阻器实现点积运算是整个忆阻器加速器的基本原理。忆阻器加速器可自然实现乘累加运算从而加速神经网络推理,其经典计算架构如图 2 所示。网络层的每个 3D 卷积核会转成一维向量映射到同一列忆阻器上。输入数据通过 DAC 单元转成一组电压序列施加于阵列的行上,每个忆阻器器件

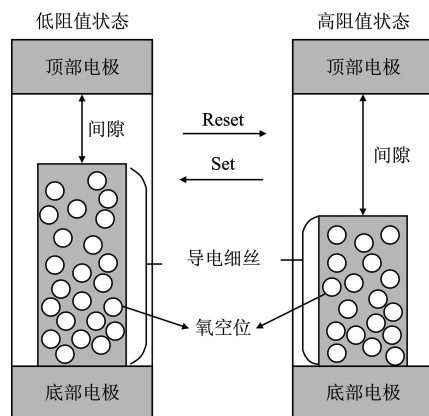


图 1 忆阻器模型

会产生相应的电流值,这样的—个电流值正好是电压与器件电导值的乘积。而位于同一列上的忆阻器器件产生的电流通过列方向的金属线进行汇聚实现加法的效果,完成一次乘累加的运算。列方向会聚的电流会通过一个采样保持单元以及共享的 ADC 单元实现模数转换最终获得数字域的计算结果。

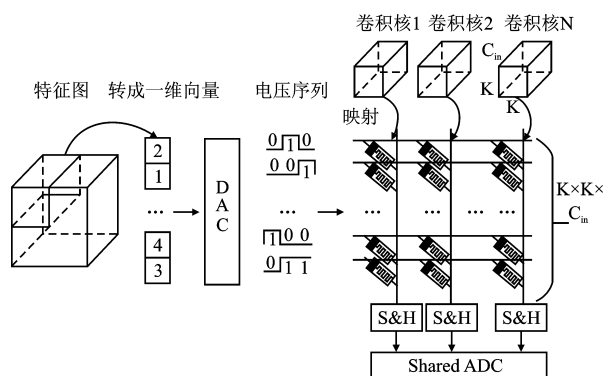


图 2 基于忆阻器的神经网络计算原理

3 神经网络模型压缩框架

3.1 框架总览

图 3 是本文提出的面向忆阻器加速器的神经网络模型压缩框架示意图。整个框架的输入是一个原始的密集型神经网络模型,输出是一个适用于忆阻器神经网络加速器的规则裁剪后的量化模型。原始的神经网络模型通过本文提出的忆阻器阵列感知的规则化增量剪枝算法,可以获得一个稀疏模型。该稀疏模型在硬件部署时对忆阻器阵列极为友好。图 3 中白色方块为 0 值权重,灰色方块为非 0 权重。每个忆阻器阵列在行方向存在数量不等的全 0 行,该 0 行对应的忆阻器器件可被节省下来用于映射其他非 0 行的权重。经过权重重排,稀疏性模型重新组合成规模更小的稠密型模型。之后该裁剪后的模型通过

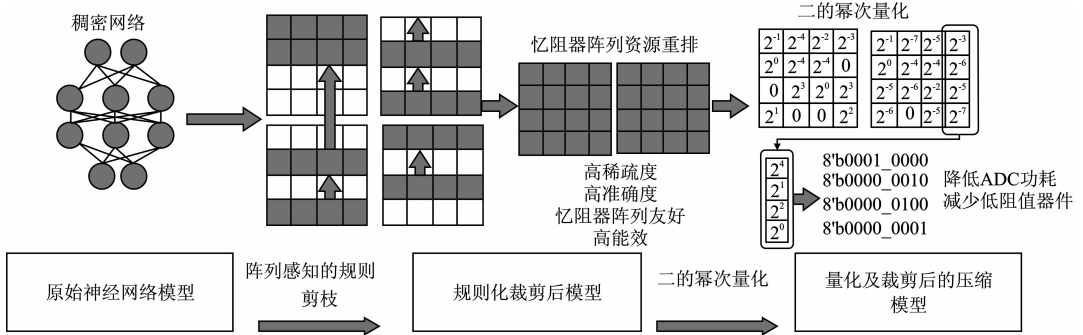


图3 神经网络模型压缩框架

本文提出的量化算法将所有的非 0 权重约束成二的幂次形式。最终我们只需将这个裁剪后的量化模型映射到 2.2 节所叙述的忆阻器神经网络加速器系统中即可。

3.2 忆阻器阵列感知的规则化增量剪枝算法

3.2.1 剪枝粒度的影响

为探讨剪枝粒度对忆阻器加速器的影响,本文首先说明神经网络模型在实际忆阻器阵列中的映射方式。如图 2 所示,网络层中的每个 3D 卷积核映射到同一列忆阻器上。考虑到忆阻器受非理想性因素影响,实际可制造的忆阻器阵列列方向器件个数远小于卷积核元素个数,因此需要在垂直方向上安排多个忆阻器阵列对卷积核的切分后进行映射。同时神经网络模型的输出通道数量决定了忆阻器阵列行方向器件个数。当模型输出通道数(3D 卷积核个数)超过忆阻器阵列列数时,水平方向上也需要多个忆阻器阵列。

图 4 展示了不同剪枝粒度在忆阻器阵列下的表现形式,其中白色方块表示 0 值权重,黑色方块表示非 0 权重。假设原始网络层为 8 个大小为 $2 \times 2 \times 2$ 的 3D 卷积核,忆阻器阵列大小为 4×4 。那么映射这样的一层网络需要 2×2 , 共 4 个忆阻器阵列。图 4(a)为随机稀疏算法,通常经过该算法处理的模型准确度和稀疏度最高。但由于 0 值分布随机,硬件上无法删去 0 值对应的忆阻器器件,因此无法节省忆阻器硬件资源。图 4(b)为当前已有的忆阻器加速器采用的剪枝算法,即裁剪掉整个 3D 卷积核或者整个输入通道^[5-7]。从忆阻器阵列尺度去看,则为水平或垂直方向跨越多个忆阻器阵列删去整行或整列。其剪枝粒度最高,也因此容易删去关键权重导致其模型准确度和稀疏度最低。图 4(c)为本文提出的阵列感知的规则化剪枝算法对应的忆阻器阵列映射形式。剪枝算法考虑到忆阻器阵列规模,可按实际阵列大小进行剪枝粒度的调整,因此具有一定的灵活性

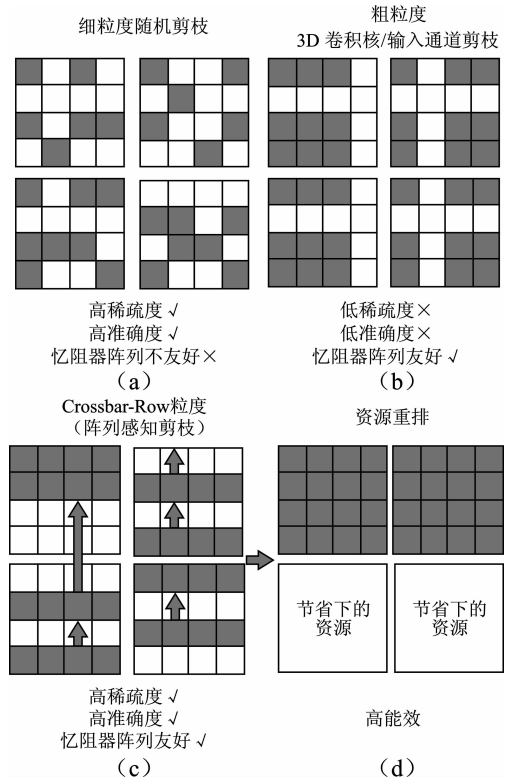


图4 不同剪枝粒度在忆阻器阵列下的映射形式

和通用性。同时因其在单个忆阻器阵列内部仍为规则化的剪枝,因此在计算时可以节省下整行 0 值对应的忆阻器器件,将其进行权重映射的重分配。如图 4(d)中所示,经过权重重排可以节省下大量硬件资源。

3.2.2 阵列感知的规则化增量剪枝算法实现细节

图 4(c)展示了本文提出的剪枝算法在忆阻器阵列上的映射形式和硬件资源节省的原理。算法 1 为本文提出的阵列感知的规则化增量剪枝算法。

算法 1 CrossbarAwarePrune

输入: 原始模型 W , 忆阻器阵列尺寸 L , 目标准确度 α , 训练次数 β , 剪枝率 φ , 剪枝率增量 $\Delta rate$ 。

输出: 裁剪后模型 W_{pruned}

1. 令 $\gamma = 0$, $\text{Try} = 0$, 根据剪枝率 φ 设定分层稀疏配置 $\text{sparse_cfg} = \{\theta_{l_1}, \theta_{l_2}, \dots, \theta_m\}$
2. While $\gamma \leq \alpha$ or $\text{Try} \leq \beta$ do
3. foreach layer $(l_i) \in \text{Model}$ do
4. 将当前网络层 l_i 转成 GEMM 形式 $W_{H \times W}$
5. 计算 L 尺寸下每行权重的 L2 范数 $\text{set}(\|W\|_{l_2})$
6. 根据当前层稀疏配置 θ_{l_i} 去除 L2 范数小的行
 $W_{\text{pruned}} = \text{PruneRow}(\text{set}(\|W\|_{l_2}), \theta_{l_i})$
7. 重训练模型 $W_{\text{pruned}} = \text{ReTrain}(W_{\text{pruned}})$
8. 测试准确度 $\gamma = \text{Test}(W_{\text{pruned}})$
9. if $\gamma \geq \alpha$
10. 提高剪枝率获取新的分层稀疏配置
11. $\text{sparse_cfg}, \varphi = \text{IncreaseRate}(\varphi, \Delta_{\text{rate}})$
12. else
13. 降低剪枝率获取新的分层稀疏配置
14. $\text{Try} = \text{Try} + 1$
15. $\Delta_{\text{rate}} = \Delta_{\text{rate}} / 2$
16. $\text{sparse_cfg}, \varphi = \text{DecreaseRate}(\varphi, \Delta_{\text{rate}})$

整个算法的输入包括一个用于裁剪的原始神经网络模型 W , 忆阻器阵列尺寸 L , 模型目标准确度 α , 网络训练次数阈值 β , 初始剪枝率 φ 以及剪枝率阶梯增量 Δ_{rate} . 为了解决由剪枝引起的模型准确度下降的问题, 本文采用增量剪枝的策略. 首先将剪枝率 φ 设成较低的初始值, 使其在首次剪枝中只裁剪掉少量以忆阻器阵列行为粒度的权重, 然后通过重训练恢复之前的模型准确度, 之后会根据剪枝率阶梯增量 Δ_{rate} 提高剪枝率 φ 以进一步裁剪模型, 再通过重训练恢复精度. 整个提高剪枝率再恢复精度的过程将一直执行直到整个模型的准确度达到设计的阈值目标 α 同时网络训练次数达到 β .

值得注意的是, 本文提出的剪枝算法中给模型的每层网络设定了差异化的剪枝率以最大程度提升模型性能, 图 5 的示意图也说明了此点. 在模型中浅层网络以及靠近输出层的网络对模型性能的影响较大, 而中间层网络冗余性最高对模型性能影响也最低. 因此剪枝率 φ 为模型整体剪枝率参数, 而各层剪枝率按网络层次设定相关不同的剪枝率参数配置.

本文采取的剪枝原则为裁剪掉 L2 范数较小的行. 首先我们需要将当前层网络转成通用矩阵乘的形式, 然后以忆阻器阵列行的规模计算每一行权重的 L2 范数值. L2 范数的定义如公式(1)所示.

$$\|x\|_2 = \sqrt{|x_1|^2 + |x_2|^2 + \dots + |x_n|^2} \quad (1)$$

根据每层网络设定的剪枝率得出每层网络中 L2 范数裁剪的阈值, 再将低于该阈值的权重参数设

置为 0. 在统计 L2 范数时是以整层网络为尺度, 因此对应到每个忆阻器阵列时, 每个忆阻器阵列删去的行数不固定.

图 5 (a)~(j) 为 VGG-16 网络经过阵列感知的规则剪枝算法压缩处理后, 部分网络层的权重可视化示意图. 图中白色部分为 0 值权重, 黑色部分为非 0 权重. (a)~(j) 每张图代表一层网络转成通用矩阵乘形式后的二维示意图. 由图可知, 每层网络的稀疏度各不相同, (a)~(f) 图黑色非 0 权重占比高于 (g)~(j) 图. (j) 图仅存少量非 0 权重. 该实验结果与算法预期设定一致.

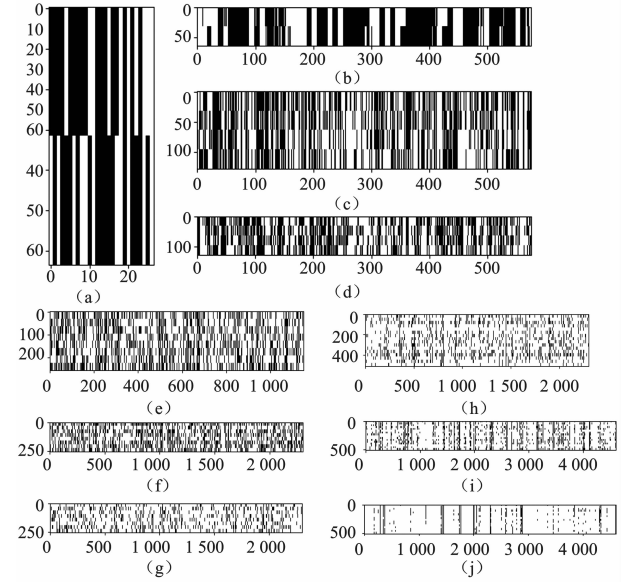


图 5 VGG-16 剪枝后部分网络层可视化示意图

图 6 展示了 VGG-16 网络经过阵列感知的剪枝后网络输入数据通道数的变化情况. 随着阵列尺寸 L 的增加, 输入通道数也随之降低. 而通道数的降低在加速器实现网络前向推理时意味着数据搬运读取次数的降低, 数据频繁读取带来的功耗也得以降低. 深层网络中输入通道数的降低尤为明显.

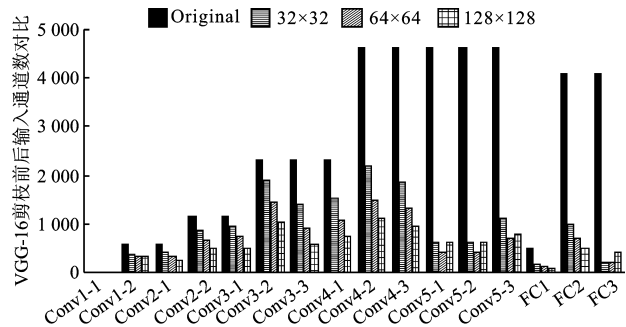


图 6 不同尺寸阵列感知的剪枝前后输入通道数对比

经过上述阵列感知规则化增量剪枝算法裁剪后,我们将得到一个细粒度规则剪枝的神经网络模型,以及一个记录着重排后的权重映射索引表. 这样的—个剪枝后的神经网络模型后续将通过二的幂次量化算法进一步降低系统功耗. 同时由于裁剪带来的权重重排等问题,输入数据次序也需要经过重新调整. 因此权重映射索引表将被存储在相关缓存模块用来从上级缓存单元取出恰当的输入数据.

3.3 二的幂次量化算法

3.3.1 传统的均匀量化算法

均匀量化是一种常用的将浮点权重转为定点权重的方法. 整个量化过程如下所述:

首先确定量化的缩放比 $scale$. 假定每个权重的绝对值将被量化成 k 比特. 而这个量化的缩放比 $scale$,则由当前网络层的绝对值最大的权重 $|x|_{\max}$ 所决定,公式为

$$S = \frac{2^k}{2^{\lceil \log_2 |x|_{\max} \rceil}} \quad (2)$$

那么被量化后的权重则如公式(3)所示:

$$x_{\text{quan}} = \text{round}(S \times |x|) \quad (3)$$

假定参数 k 为 8,那么整个量化权重范围则被约束成 $[0, 255]$ 内的整数. 这些整数所对应的二进制编码拥有不同数量的比特. 由 2.1 节内容可知,比特 1 通常通过低阻值忆阻器器件进行表示,因此均匀量化在降低低阻值忆阻器器件比例方面无法起到作用.

3.3.2 二的幂次量化算法实现细节

ADC 功耗是整个系统功耗中占比最大的部分. 降低 ADC 的精度能够有效降低 ADC 的功耗. 而 ADC 的精度值由流过忆阻器阵列方向上的电流值的个数所决定. 如果能够降低流过忆阻器阵列方向的电流种类数,那么低精度的 ADC 即可成功区分各个电流值. 本文提出的二的幂次量化算法有效的降低电流种类数,同时能够降低忆阻器阵列中低阻值器件的比例.

二的幂次量化算法的含义即是将所有非 0 浮点权重量化成最靠近其数值的 2^n . 如果直接将所有权重一次全部量化,那么模型准确度将大幅下降. 因此量化部分采用增量量化的手段,即将权重分为两组,量化其中—组为 2 的幂次或 0,重训练更新另一组的权重以弥补精度损失,不断迭代直至所有权重被量化. 算法细节如算法 2 所示:

算法 2 PowerOfTwoQuan

输入: 剪枝后模型 W_{pruned} , 初始分组率 α , 量化精度 k

比特.

输出: 量化后模型 W_{quan}

1. 根据初始分组率 α 将每层参数划分成 2 组
2. 求出每层权重的绝对值最大值 $|x|_{\max}$
3. 根据 $|x|_{\max}$ 确定量化范围 $\{\pm 2^{n_1}, \dots, \pm 2^{n_2}, 0\}$ 的上界 $n_1 = \lceil \log_2(4 |x|_{\max})/3 \rceil$
4. 根据 k 和 n_1 确定下界 $n_2 = n_1 + 1 - 2^{k-1}/2$
5. 将步骤 1 中的待量化组的非 0 浮点权重量化成离它最近的 2^n , 并重训练另一组未量化的权重
6. 提高分组率 α , 并重复步骤 2 ~ 5, 直到所有权重被量化

为应用二的幂次算法,忆阻器加速器需要进行一定修改以满足算法要求. 图 7 展示了该量化算法到忆阻器阵列上的映射形式. 介于该量化算法只限制了每层网络指数的差值为固定常数,因此每层网络权重的实际分布范围并不完全一致. 图 7 (a) 展示了二的幂次权重实际映射到忆阻器阵列时的权值. 为了将各层网络的权重限制在统一的范围之内,实际所映射的权重是原始权重除以当前层绝对值最小的权重得出的缩放后权重. 如图 7 (a) 所示,经过缩放后,每层网络的权重均被限制在集合 $\{2^0, 2^1, \dots, 2^7\}$ 中. 图 7 (b) 表示的是二的幂次权重二进制编码在忆阻器阵列上的映射方式. 假设幂指数范围限制在 $[0, 7]$. 那么每个权重的二进制编码将被限制在 8 比特,这 8 比特编码中有且仅有一个 1 比特数值. 因此映射每个权重的每一位二进制编码共需要 8 个相关的忆阻器阵列分别映射不同位置的二进制编码值. 由图 7 (b) 可以发现,每个忆阻器阵列绝大部分用于映射 0 比特数值,即大部分器件为高阻值器件,因此阵列功耗得以降低.

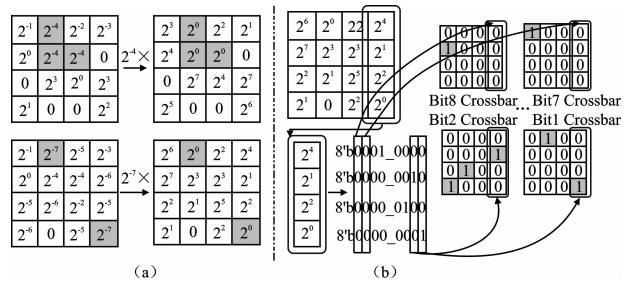


图 7 量化算法的忆阻器阵列映射形式

图 8 显示了三种不同的网络在经过二的幂次量化算法后,忆阻器阵列功耗的变化情况. 与均匀量化相比,经过二的幂次量化之后,忆阻器计算阵列功耗可以降低 43.95%~50.91%. 这主要是源于低阻值器件的减少带来的功耗降低.

由图 7 (b) 可观察到每个忆阻器阵列的列方向上仅有少量的 1 比特数值映射到低阻值器件. 因此

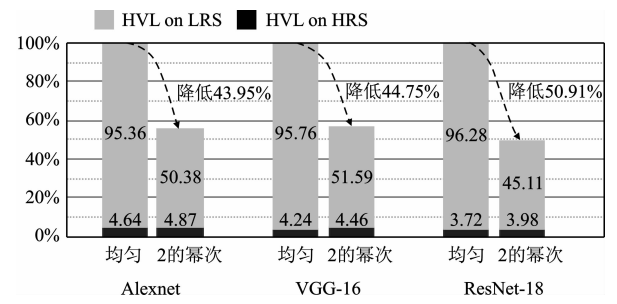


图 8 不同量化算法对应的忆阻器阵列功耗

当行方向上施加电压时,整个列方向上汇聚而成的电流值分布范围将远远小于均匀量化产生的电流值分布范围,从而使使得 ADC 的精度需求得以降低. 表 1 为两种量化算法对应的忆阻器阵列汇聚产生的电流值种类数. 本文在三种网络上用实际图片模拟 DAC 电压测试了不同阵列大小的电流种数. 结果表明当采用二的幂次算法时,实际电流种类数大大降低. 根据表 1 数据 ADC 分辨率可降低 2 比特.

表 1 不同量化算法对应的列方向电流值最大种类数			
网络	阵列规模	均匀量化 电流种数	二的幂次量 化电流种数
Alexnet	32	45	12
	64	104	27
	128	238	44
VGG-16	32	53	14
	64	113	30
	128	203	39
ResNet-18	32	36	15
	64	96	25
	128	197	40

4 实验结果与分析

4.1 实验环境

为评估上述模型压缩框架,本文选取了当前主流的三种网络模型 AlexNet、VGG-16 和 ResNet-18 在 Cifar10 数据集上进行实验. 本文搭建了一个基于文献[8]的忆阻器神经网络加速器,并进行一定修改. 整个忆阻器加速器中忆阻器阵列参数来源于 NVsim 仿真结果^[9]. 本文利用 CACTI^[10] 对所涉及到的缓存单元模块进行建模并获取相关功耗和延时信息. 本文选取了和 ISAAC 中相同的 ADC 单元并根据实际精度需求将其功耗和面积等参数缩放到 6bit 精度下的情况. DAC 单元参数也来源于ISAAC 中参数. 数字电路部分用 Verilog 实现相关电路并用 Design Compiler 在 32 nm 工艺下进行综合并获

取相关参数. 加速器计算单元关键模块参数如表 2 所示.

表 2 加速器计算单元关键模块参数		
模块	配置	功耗
ADC	数量: 8;精度: 6 bit;采样率: 1.2 GSps	5.14 mW
DAC	数量: 8×64;精度: 1 bit	2 mW
采样保持模块	数量: 8×64	5 μW
忆阻器阵列	数量: 8;尺寸: 64;SLC 器件	0.336 mW
移位加模块	数量: 8	0.4 mW
地址译码单元	数量: 1	0.62 mW
输入缓存	大小: 0.5 KB	0.392 mW
输出缓存	大小: 64 B	0.042 mW

4.2 实验结果

4.2.1 准确度与稀疏度

本文提出的面向忆阻器加速器的神经网络模型压缩框架在三种主流网络上实验得出的准确度和稀疏度结果如表 3 所示. 为了证明该模型压缩框架的通用性和有效性,本文在 32×32、64×64 和 128×128 这三种忆阻器阵列规模上测试了该框架并将结果进行对比. 实验结果表明经过剪枝和量化之后,主流网络模型相比原始网络模型精度损失大部分小于 1%,同时整个模型稀疏度最高可达 92.74%,充分证明本文提出的压缩框架在取得极高的模型压缩率情况下模型性能仍保持稳定.

通过比较同一网络不同忆阻器阵列大小的实验结果可以发现忆阻器阵列越大,即规则剪枝粒度越大,整个模型的准确度越低,同时所能达到的稀疏度也越低. 这一实验结果也与 3.2.1 节讨论的剪枝粒度的影响相一致.

表 3 不同网络经模型压缩后的准确度与稀疏度结果					
网络	阵列 规模	原始准 确度(%)	剪枝准 确度(%)	剪枝+量化 准确度(%)	稀疏度 (%)
Alexnet	32	77.20	77.04	76.53	91.89
	64	77.20	76.51	75.88	91.87
	128	77.20	76.37	75.80	91.87
VGG-16	32	93.80	93.06	93.00	92.76
	64	93.80	92.80	92.77	92.74
	128	93.80	92.53	92.38	91.71
ResNet-18	32	94.80	94.65	94.58	87.25
	64	94.80	94.59	94.41	85.45
	128	94.80	94.21	94.00	83.60

4.2.2 性能与功耗

表 4 为本文与同类研究能耗情况对比结果. 结果显示本文压缩框架相比文献[8]和[4]能耗可降低

一个数量级. 相比传统均匀量化,由二的幂次带来的能耗可降低 64%. 图 9 展示了三种网络模型在原始网络以及不同压缩策略下的归一化能效情况. 测试基准为文献[8] 能效情况,其余为文献[4] 能效情况、本文剪枝算法与传统的均匀量化算法压缩后的能效,以及本文提出的网络模型压缩框架处理后的能效.

表 4 一个计算周期内的本文能耗与同类研究能耗比较
单位:nJ

网络	本文	剪枝+均匀量化	文献[8]	文献[4]
Alexnet	1 194.23	3 395.67	33 638.02	17 727.91
VGG-16	1 804.89	5 128.80	55 386.13	26 374.35
ResNet-18	1 068.08	3 017.19	18 324.28	13 088.78

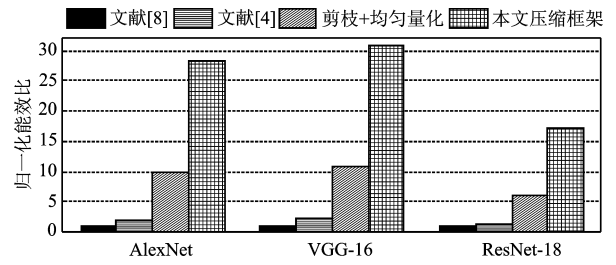


图 9 不同网络归一化能效比

通过将文献[4]、文献[8]与第一种压缩策略的能效情况进行对比,可以发现本文提出的剪枝算法可以使得忆阻器加速器系统获得 6.1~10.8 倍的能效提升. 原因在于通过裁剪掉冗余的模型参数,大量的硬件资源例如忆阻器阵列、ADC 以及 SRAM 缓存单元被节省了下来. 比较第一种压缩策略和本文提出的压缩策略的实验结果可以发现,由于 ADC 精度的降低和低阻值忆阻器数量的减少,整个系统功耗在剪枝的基础上进一步得到了改善. 最终整个系统能效的提升可以达到 17.2~30.7 倍. 同时对比不同网络的能效提升情况可以发现,本文提出的模型压缩框架在大型网络上的实验效果优于小型网络.

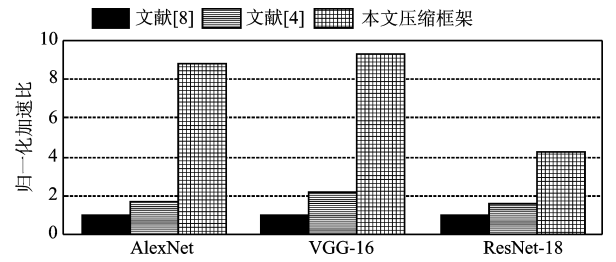


图 10 不同网络归一化加速比

图 10 展示了三种不同网络经过本文提出的模

型压缩框架处理之后,忆阻器加速器能够取得的加速比情况. 基准为文献[8]推理速度. 由于量化算法对忆阻器加速器的计算速度不产生影响,本实验仅测试本文压缩框架. 实验结果显示,本文提出的压缩框架可实现 4.3~9.3 倍的速度提升. 这主要来源于剪枝算法带来的资源节省可用于提高计算单元的并行性加快计算速度.

5 结束语

本文针对当前忆阻器神经网络加速器存在的资源需求过大、功耗过高等问题,提出了一种高能效的模型压缩框架. 该框架包含了一种阵列感知的规则化增量剪枝算法和二的幂次量化算法. 与已有的粗粒度规则剪枝算法相比,本文的剪枝算法对应的模型准确度和稀疏度更高,能够节省更多硬件资源. 本文的量化算法可同时降低 ADC 精度需求以及阵列中低阻值忆阻器器件比例进而降低 ADC 功耗和忆阻器阵列功耗. 经过仿真,本文提出的模型压缩框架相比原始模型可取得 17.2~30.7 倍的能效提升以及 4.3~9.3 倍的加速,同时模型的精度损失较小.

参考文献:

[1] CHUA L. Memristor-the missing circuit element[J]. IEEE Transactions on circuit theory, 1971, 18(5): 507-519. DOI: 10.1109/TCT.1971.1083337.

[2] STRUKOV D B, SNIDER G S, STEWART D R, et al. The missing memristorfound[J]. Nature, 2008, 453(7191): 80-83. DOI: 10.1038/nature06932.

[3] HU M, STRACHAN J P, LI Z Y, et al. Dot-product engine for neuromorphic computing: Programming 1T1M crossbar to accelerate matrix-vector multiplication[C]//2016 53nd ACM/EDAC/IEEE Design Automation Conference (DAC). Austin, TX, USA: IEEE, 2016: 1-6. DOI: 10.1145/2897937.2898010.

[4] SHAFIEE A, NAG A, MURALIMANO HAR N, et al. ISAAC: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars [J]. ACM SIGARCH Computer Architecture News, 2016, 44(3): 14-26. DOI: 10.1145/3007787.3001139.

[5] JI H X, SONG L H, JIANG L, et al. ReCom: An efficient resistive accelerator for compressed deep neural networks[C]//2018 Design, Automation & Test in Europe Conference & Exhibition (DATE). Dresden, Germany: IEEE, 2018: 237-240. DOI: 10.23919/DATE.2018.8342009.

[6] WANG P Q, JI Y, HONG C, et al. SNrram: an effi-

- cient sparse neural network computation architecture based on resistive random-access memory[C]//2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC). San Francisco, CA, USA: IEEE, 2018: 1-6. DOI: 10.1109/DAC.2018.8465793.
- [7] LIN J L, ZHU Z H, WANG Y, et al. Learning the sparsity for ReRAM: mapping and pruning sparse neural network for ReRAM based accelerator[C]//Proceedings of the 24th Asia and South Pacific Design Automation Conference. Tokyo, Japan: ACM, 2019: 639-644. DOI: 10.1145/3287624.3287715.
- [8] ZHU Z H, SUN H B, LIN Y J, et al. A configurable multi-precision CNN computing framework based on single bit RRAM[C]//2019 56th ACM/IEEE Design Automation Conference (DAC). Las Vegas, NV, USA: IEEE, 2019: 1-6.
- [9] DONG X Y, XU C, XIE Y, et al. NVSim: A circuit-level performance, energy, and area model for emerging nonvolatile memory[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2012, 31(7): 994-1007. DOI: 10.1109/TCAD.2012.2185930.
- [10] MURALIMANO HAR N, BALASUBRAMONIAN R, JOUPPI N P. CACTI 6.0: A tool to model large caches[R]. HP Laboratories, 2009.

作者简介:

沈林耀 男,(1995-),硕士研究生.研究方向为忆阻器神经网络加速器设计、大规模集成电路设计.

E-mail:shenlinyao@sjtu.edu.cn

王 琴 女,(1975-),研究员.研究方向为大规模集成电路设计、先进集成电路设计方法.

蒋剑飞 男,(1979-),助理研究员.研究方向为高速数字集成电路设计.

景乃锋 男,(1982-),博士,副研究员.研究方向为计算机体系结构.