RAELLA：重构高效、低分辨率、低损耗模拟存内处理的算术运算：无需重新训练！

Tanner Andrulis 麻省理工学院 美国马萨诸塞州剑桥 andrulis@mit.edu

Joel S. Emer 麻省理工学院、英伟达 美国马萨诸塞州剑桥 jsemer@mit.edu

Vivienne Sze 麻省理工学院 美国马萨诸塞州剑桥 sze@mit.edu

arXiv:2304.07935v1 [cs.AR] 17 Apr 2023

摘要

存内处理（PIM）加速器有望通过减少高昂的数据搬移，并利用阻变存储器（ReRAM）实现高效模拟计算，从而高效运行深度神经网络（DNN）推理。遗憾的是，存内处理加速器的整体效率受限于能耗高昂的模数转换器（ADC）。此外，现有降低模数转换器成本的加速器，要么通过改变深度神经网络权重来实现，要么采用会降低输出保真度的低分辨率模数转换器。这些策略会损害深度神经网络准确率，和/或需要耗费高昂成本进行深度神经网络重新训练以弥补损失。

为解决这些问题，我们提出了 RAELLA 架构。RAELLA 针对每个深度神经网络调整架构；它通过编码权重以产生接近零的模拟值、针对每个深度神经网络层自适应地切片权重，以及通过推测和恢复动态切片输入，降低计算所得模拟值的分辨率。低分辨率模拟值使 RAELLA 既能使用高能效的低分辨率模数转换器，又能在无需重新训练的情况下保持精度，同时减少 ADC 转换次数。

与其他精度损失较低的存内计算加速器相比，RAELLA 的能效最高提升 4.9×，吞吐量最高提升 3.3×。与会造成精度损失并通过重新训练深度神经网络来恢复精度的存内计算加速器相比，RAELLA 无需进行代价高昂的深度神经网络重新训练，即可达到相近的能效和吞吐量。

CCS 概念

• **计算机系统组织** → **模拟计算机**；**神经网络**；• **硬件** → 新兴架构。

关键词

存内处理、存内计算、模拟、神经网络、加速器、架构、切片、模数转换器、阻变存储器

ACM 参考格式：

Tanner Andrulis、Joel S. Emer 和 Vivienne Sze。2023 年。RAELLA：重构高效、低分辨率、低损耗模拟存内处理的算术运算：无需重新训练！载于《第 *50* 届年度国际计算机体系结构研讨会论文集》（*ISCA '23*），*2023*年*6*月*17*日至*21*日，美国佛罗里达州奥兰多。ACM，美国纽约州纽约，16 页。

https://doi.org/10.1145/3579371.3589062

允许免费制作本作品的部分或全部内容的数字或纸质副本，供个人或课堂使用，但前提是不得为盈利或商业利益制作或分发副本，且副本须注明本声明及首页的完整引文。必须尊重本作品第三方组件的版权。其他用途请联系版权所有者/作者。*ISCA '23*，*2023*年*6*月*17*日至*21*日，美国佛罗里达州奥兰多。© 2023 版权所有者/作者保留版权。ACM ISBN 979-8-4007-0095-8/23/06。https://doi.org/10.1145/3579371.3589062

1 引言

存内处理（PIM）是解决深度神经网络（DNN）推理所需高计算量和高能耗的一个有前景的方案。通过在存储器中进行计算，[34], 存内计算加速器可避免深度神经网络权重的高昂片外数据移动开销 [59]。此外，存内计算加速器通常采用阻变存储器（ReRAM）器件和ReRAM交叉阵列 [5, 54, 56]，以实现密集、高效的模拟计算 [34]。

遗憾的是，尽管忆阻器交叉阵列能够高效且高密度地进行计算，但存内处理加速器的总体能耗往往主要受读取交叉阵列中计算所得模拟值的模数转换器（ADC）支配。由于模数转换器开销，一些存内处理加速器 [47, 54] 即使拥有存内处理带来的机遇，其能耗也未能显著优于非存内处理加速器 [4] 。

一些先前的研究尝试通过降低模数转换器的分辨率来减少这种模数转换器开销，从而以指数级方式降低模数转换器能耗 [65]。架构通常会将深度神经网络输入和权重中的比特划分，或切片为多个较低分辨率的切片，并在多个步骤中使用不同切片进行计算 [54]。尽管切片算术可以使用低分辨率模数转换器，但模数转换器必须处理每个切片的结果，因此这些策略会用多次低分辨率模数转换，替代每次高分辨率模数转换，所以模数转换器仍主导着整体能耗。

其他存内计算加速器虽然降低了模数转换器能耗，但代价是深度神经网络精度下降。有些设计通过剪枝深度神经网络 [8, 26, 48, 75, 80]来减少深度神经网络权重计数，因此我们将这些设计称为权重计数受限。它们减少了所需的计算次数和模数转换次数，但也会造成精度损失。另一些设计则使用高效的低分辨率模数转换器，处理来自交叉阵列 [5, 7, 24]的高分辨率模拟值。我们将这些设计称为求和保真度受限，因为分辨率差异会降低输出保真度并引入误差。这些架构会重新量化深度神经网络，使其能够容忍模数转换器的分辨率限制，但这同样会导致精度损失。

为了降低这种精度损失，权重数量受限和求和保真度受限架构都会重新训练深度神经网络。这是一个问题；深度神经网络训练的计算成本极高 [43], ，可能需要繁琐的超参数调优才能达到高精度 [19],，而且如果训练数据是私有的，重新训练可能无法进行 [50, 62]。此外，前沿深度神经网络通常需要特定的训练方案 [6], ，而这些方案可能与某种架构所要求的重新训练方案不兼容。

为避免精度损失且无需重新训练，我们考察保真度限制。我们将保真度 定义为模数转换器表示所计算模拟值完整分辨率的能力。

RAELLA 的模型可在 https://github.com/mit-emze/raella 获取

当所计算模拟值的分辨率高于模数转换器分辨率时，架构会损失保真度并产生误差。每个深度神经网络都会生成多种模拟值分布，而以往受总保真度限制的方法会修改深度神经网络，重塑这些模拟值分布，使其适应分辨率受限的模数转换器范围。相比之下，我们发现，可以通过可调架构重塑模拟值分布，而无需更改深度神经网络。

基于这一关键洞见，我们提出 RAELLA 架构，实现无需重新训练的高效存内处理推理。RAELLA 通过调整算术运算和切片方式，塑造计算所得的数值分布，从而生成低分辨率模拟结果。这样，RAELLA 就能使用高效的低分辨率模数转换器，同时保持高保真度并降低深度神经网络精度损失。RAELLA 的主要贡献如下：

- 中心值+偏移编码可在模拟域中累加更多数值，同时保持总和较小且分辨率较低。具体而言，RAELLA 对深度神经网络权重进行偏移，使每个交叉阵列列中正、负权重切片的平均幅值相等。随着模拟域中的计算结果不断累加，正负结果相互抵消，产生接近零的总和，从而能够以高保真度进行转换。

- 在编译时对深度神经网络权重进行自适应切片，以平衡密度、效率和保真度。每个忆阻式随机存取存储器器件中存储更多比特，能够提高密度和效率，但会产生分辨率更高的模拟值。对于每个深度神经网络层，RAELLA 会调整每个权重所使用的忆阻式随机存取存储器器件数量，以及每个器件中的比特数。这样，RAELLA 就能采用尽可能高密度、高效率的策略，同时保持计算所得模拟值的低分辨率。

- 在运行时对深度神经网络输入激活值进行动态切片，以实现高效且高保真的计算。RAELLA 采用一种高效策略进行推测执行，在每个输入切片中处理更多位。RAELLA 使用一种效率较低但保真度更高的策略，通过将输入切分为更多切片并减少每个切片的位数，来检测并恢复错误结果。这样，RAELLA 无需降低保真度，便能进一步减少模数转换次数。

与其他低精度损失的存内计算加速器相比， [54],RAELLA 既能降低模数转换器分辨率，又能在不牺牲保真度的情况下运行深度神经网络，减少多达 14× 的模数转换次数。

我们使用七种具有代表性的深度神经网络，将 RAELLA 与三种最先进的存内计算加速器进行了比较评估。与其他低精度损失的存内计算加速器相比，RAELLA 的能效最高提升 4.9×（几何平均值为 3.9×），吞吐量最高提升 3.3×（几何平均值为 2.0×）。与需要通过 DNN 重新训练来恢复精度的权重数量受限和求和保真度受限加速器相比，RAELLA 在避免代价高昂的 DNN 重新训练的同时，仍能提供相近的效率和吞吐量。

2 背景与动机

我们首先简要概述深度神经网络推理、存内处理（PIM），以及用于降低模拟操作数分辨率的切片。随后，我们探讨模数转换器如何限制存内处理、如何降低模数转换器能耗，以及以往方法的局限性。

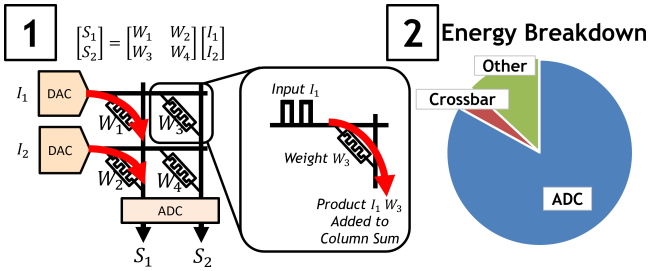

图 1：基本存内处理交叉阵列。 1 2×2矩阵-向量乘法。每个操作数都是一个切片。2 基于 ISAAC 的设计的能耗分解。

2.1 深度神经网络推理

现代深度神经网络的主要计算是卷积层和全连接层中的矩阵-向量运算 [59]。对于推理，8 位（8b）逐通道量化的深度神经网络已广泛应用，其输入和权重为 8b，部分和为 16b，并且能够实现高准确率 [22, 25, 37, 51, 82]。RAELLA 支持这种量化方式。

深度神经网络可能包含数十亿次乘累加运算（MAC），涉及数百万个权重 [16]。因此，存内处理是加速深度神经网络推理的理想选择。存内处理可直接在权重存储器中运算，以减少数据搬移 [5]，并使用阻变存储器实现高密度、高效的模拟计算。

2.2 阻变存储器特性

存内处理系统的功能单元是忆阻器交叉阵列。ReRAM交叉阵列通过执行密集的存内矩阵-向量乘法来加速深度神经网络层。此外，电阻式随机存取存储器体积小，存储密度高 [45]。这种高密度使基于阻变存储器的系统能够存储并运行片上流水线，依次计算深度神经网络层 [54, 56]，无需高成本地访问片外存储器 [59]。阻变存储器的一项缺点是写入能耗高 [34]。由于阻变存储器具有非易失性，写入的权重可在多次推理中重复使用，因此推理过程中可摊销写入成本 [54]。

图1展示了在 2×2 忆阻器交叉阵列上执行的基本 2×2 矩阵-向量乘法。权重矩阵 W 被写入电阻式随机存取存储器中。输入向量 I 的各个元素被送入数模转换器（DAC），转换为模拟值。每个忆阻式随机存取存储器器件将所在行的输入与其已写入的权重相乘。各列中的乘积累加，得到模拟列和，再由模数转换器（ADC）转换为数字结果 S。

已有研究表明，电阻式随机存取存储器可编程至最高5b [1]，而 512×512 交叉阵列在模拟噪声限制 [17] 下可计算最高8b的列和。这些分辨率限制使得必须通过切片来计算更高分辨率的深度神经网络层。

2.3 使用切片进行算术运算

为了使用低分辨率器件执行8b深度神经网络推理，存内处理架构会将输入位和权重位划分为输入切片和权重切片，即进行切片。切片是操作数中的一组位，两个切片相乘会得到一个切片乘积。

切片有两种类型。时间切片在不同周期处理各个切片（例如，位串行是极端情况，每个切片仅包含 1 位），空间切片则在并行交叉阵列列中的不同电阻式随机存取存储器中处理各个切片。在大多数存内计算加速器中，

点积： 2b 输入 $i_h i_l \cdot$ 2b 权重 $w_h w_l$

切片输入			✓		✓
切片权重				✓	✓
周期	列				
1	1	$i_h i_l \cdot w_h w_l$	$i_h \cdot w_h w_l$	$i_h i_l \cdot w_h$	$i_h \cdot w_h$
1	2	-	-	$i_h i_l \cdot w_l$	$i_h \cdot w_l$
2	1	-	$i_l \cdot w_h w_l$	-	$i_l \cdot w_h$
2	2	-	-	-	$i_l \cdot w_l$
每次乘加的位数		4	2	2	1
每次乘加运算的转换次数		1	2	2	4

表1：切片的工作原理与权衡。一个 2 位输入/权重相乘，且两者都可切分为两个 1 位切片。高位和低位比特为 i_h，w_h和 i_l，w_l。每列/周期都会计算所示的切片乘积。更多切片可降低每片位数和每次乘加的位数，从而能够使用成本更低、分辨率更低的模数转换器。然而，处理每个切片都需要周期、列和 ADC 转换次数。切片越多，每次乘加运算的转换次数就越多。

切片，输入采用时间切片，权重采用空间切片。如果一个权重向量及其切片映射到同一交叉阵列中的同一组列，并且它们共同为深度神经网络层计算一个点积，我们就将其称为权重滤波器 。

表1展示了切片算术的一个示例。每个权重切片在空间上映射到一个交叉阵列列，而每个输入切片则在一个周期内按时间处理。对于每一列和每个周期，模数转换器都会转换列和。转换结果经过移位后进行数字累加，使存内处理架构即使面对低分辨率带来的模拟限制，也能计算出完整的16位部分和 [14, 54]。

表1展示了与切片相关的权衡。随着切片数量增加，许多开销也会增加：每增加一个输入切片，*Cycles/Input* 就会增加；每增加一个权重切片，*Columns/Weight*就会增加。 *ADC Converts* 则随输入切片数量与权重切片数量的乘积而增长。增加切片数量的好处是，每个切片所需的位数可以减少，从而降低乘加分辨率和所需的模数转换器分辨率。我们还可以通过使用更大的交叉阵列，在更多行上累积更多模拟值，以降低 *ADC Converts*；但这也会提高所需的模数转换器分辨率。

2.4 模数转换器限制存内计算加速器

Fig. 图1展示了基于奠基性工作 ISAAC [54]的8位存内计算架构的功耗分解。存内处理交叉阵列密集且高效，但受模数转换器成本限制。交叉阵列可以用 < 100fJ 执行8位乘加运算，但整体能耗主要由模数转换器决定。交叉阵列面积密集，但架构中模数转换器的面积可能是交叉阵列的 5 [24] 到 50 [54] 倍。交叉阵列可以通过高并行度进行计算，扩展至1024行 [32], , 但模数转换器的面积和能耗会随分辨率呈指数级增长 [65]。以往的工作将激活行数限制在最低仅16行 [75] ，以降低列和及模数转换器分辨率要求。

由于阻变存储器密度高、功耗低，RAELLA 以使用更多阻变存储器为代价，换取分辨率更低的模数转换器。此外，通过降低分辨率，我们可以使用更多交叉阵列行/列，同时减少模数转换器面积/能耗的扩展。更高的并行度使整个 RAELLA 加速器获得更高的吞吐量和效率。

2.5 降低模数转换器成本

为了实现高效的 PIM 推理，我们必须降低模数转换器的面积和能耗。为此，我们提出了模数转换器能耗的钛定律。[1][2]表2展示了模数转换器能耗的钛定律公式，并分解了其中的各项因素。模数转换器能耗是以下四个项的乘积：

- 每次转换能耗 由模数转换器效率决定，并随模数转换器分辨率呈指数级增长 [65].[3]
- 每次乘加运算的转换次数 由交叉阵列行数、输入切片和权重切片的数量决定。• 每个深度神经网络的乘加运算数 由深度神经网络工作负载决定。
- *1*/利用率 对应于深度神经网络使用的交叉阵列行数。利用率为 1 表示使用了所有行。

综合考虑这些因素，表2展示了如何通过改变硬件属性来降低模数转换器能耗。首先，请注意表格第一行和第二行中每次转换能耗 与每次乘加运算的转换次数 之间的权衡。虽然切分并调整交叉阵列的尺寸似乎可以直接降低模数转换器能耗，但这种方法的收益有限。这是因为，要降低每次乘加运算的转换次数，我们必须采取以下措施之一：(1) 增加交叉阵列行数，使每次模数转换能处理更多切片后的乘积；(2) 增加每个权重切片的位数，从而减少存储每个权重所需的列数，并减少处理每列所需的模数转换次数；或者 (3) 增加每个输入切片的位数，从而减少处理所有周期中的列和所需的周期数和模数转换次数。然而，这些方法都有一个局限：在所有情况下，我们都会累积数值更大、分辨率更高的列和。为了保持保真度，需要使用更高分辨率的模数转换器，这会增加每次转换能耗 ，抵消我们获得的收益。反过来，若要降低每次转换能耗，就必须提高保真度，因此需要增加每次乘加运算的转换次数。

表2的最后一列展示了减少钛定律各项所带来的后果。六种后果中，有三种无法降低模数转换器能耗。每次乘加运算的转换次数 与能耗/转换次数 此消彼长。*1*/利用率 无法降至1以下。

能够降低模数转换器能耗的架构会选择最终导致**精度损失或重新训练**的方案。图2展示了这些架构如何改变深度神经网络操作数并造成精度损失。表2中标有 W◯ 的单元格对应权重数量受限架构，它们通过剪枝或重塑深度神经网络权重来降低乘加运算/深度神经网络。遗憾的是，除非重新训练深度神经网络，否则改变权重会导致深度神经网络精度损失。另一方面，标有 S◯ 的单元格对应求和保真度受限架构，它们使用更多行、每个输入/权重切片采用更多位，并使用低分辨率模数转换器，以同时降低每次乘加运算的转换次数 和乘加运算/深度神经网络。但由于它们会生成幅值大、分辨率高的列和，却使用低分辨率模数转换器，因此列和保真度会降低。这会导致输出出现误差，并造成精度损失，除非对深度神经网络进行重新量化和重新训练。

反直觉的是，在求和保真度受限架构中，8 位模数转换器并不总是足以处理 8 位量化的深度神经网络。8 位输入和权重经过乘加运算后，部分和为 16 位，而高精度线性

[1]受铁律 [68] 和钛基阻变式存储器器件的启发 [13]。[2]此处重点关注模数转换器能耗，但也可以通过将每次乘加运算的转换次数 替换为模数转换器数量与吞吐量之比，对面积进行类似分析。[3]每次转换能耗 也可以通过巧妙设计新型模数转换器来降低，但受模拟噪声影响，效率存在上限 [35] 。这需要在模数转换器和架构两方面进行创新。

钛定律： $\frac{ADCEnergy}{DNN} = \frac{Energy}{Convert} \times \frac{Converts}{MAC} \times \frac{MACs}{DNN} \times \frac{1}{Utilization}$

Term	硬件属性	如何降低	权衡	后果
每次转换能耗	ADC 分辨率	降低模数转换器分辨率	更少的交叉阵列行数或每片位数	高 每次乘加运算的转换次数
			S◯保真度损失、部分和误差	S◯**精度损失或重新训练**
每次乘加运算的转换次数	交叉阵列 Rows	增加交叉阵列行数行数或每片位数	高分辨率模数转换器	高能耗/每次转换
			S◯保真度损失、部分和误差	S◯**精度损失或重新训练**
乘加运算／深度神经网络	# 权重	剪枝／重塑权重	W◯消除／更改的权重	W◯**精度损失或重新训练**
1／利用率	映射	改进映射	灵活性成本、利用率 ≤ 1	有限的收益

表2：模数转换器能耗的钛定律，以及如何降低模数转换器能耗组成部分。在可能的后果中，三种方案无效，另外三种会导致精度损失或需要DNN重新训练。求和保真度受限架构选择标记为 S◯ 的单元格，权重数量受限架构选择标记为 W◯ 的单元格。

架构	高成本 ADC	限制权重计数	保真度 Loss	需要深度神经网络重新训练
ISAAC [54]	**Yes**	-	-	**No**
AtomLayer [47]	**Yes**	-	-	**No**
FORMS [80]	**No**	**Yes**	-	**Yes**
SRE [75]	**No**	**Yes**	-	**Yes**
ASBP [48]	**No**	**Yes**	-	**Yes**
TIMELY [24]	**No**	-	**High**	**Yes**
PRIME [5]	**No**	-	**High**	**Yes**
RAELLA	**No**	-	**Low**	**No**

表3：与先前工作的比较。以往方法要么付出高昂的模数转换器成本，要么采用会导致深度神经网络精度损失的策略，因此需要重新训练来恢复精度。

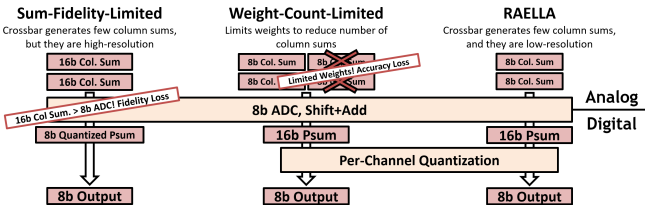

图 2：造成损失的架构与 RAELLA 并列展示。尽管它们降低了每次乘加运算的转换次数，求和保真度受限架构会在模数转换器处损失保真度，并强制采用受硬件限制的量化。权重数量受限架构会限制深度神经网络权重。RAELLA 的算术和切片策略能够以较低的每次乘加运算的转换次数保持高保真度。

量化策略需要完整的 16b 部分和范围 [51, 69, 82], , 因此我们希望全部 16b 都具有高保真度。求和保真度受限架构可能会生成 $> 8b$ 个列和，并使用 8b 模数转换器进行采集。当这些架构中的模数转换器丢失列和中的位时，整体部分和也会丢失相应的位。这会限制高精度量化策略，使其只能使用硬件计算出的位子集，而不是完整的 16b。这种硬件强制限制可能导致精度损失。

2.6 动机

以往工作通过重新训练深度神经网络来应对精度损失。FORMS [80], 是一种权重数量受限架构，通过剪枝和重新训练，在 ResNet18 上实现了 2.0× *MACs/DNN*的减少。TIMELY [24],是一种求和保真度受限架构，通过使用大型交叉阵列以及每个输入/权重切片采用较多位数，在 [54] 上实现了最高 512× 转换/*MAC* 的减少。然而，TIMELY 也会使每个列和损失 16b 的保真度，并通过深度神经网络再量化和重新训练来恢复精度。表3展示了近期存内处理研究中的一个不足：一些存内计算架构效率低下，无法降低高昂的模数转换器成本；另一些虽然降低了模数转换器成本，却会造成深度神经网络精度损失，并通过重新训练来弥补。

重新训练深度神经网络可能面临挑战，原因包括计算成本高昂 [43], 超参数调优繁琐 [19], 以及可能无法获得训练数据集 [50, 62]。此外，高效率的深度神经网络（例如高度降低精度的模型） [6, 9, 12] 往往依赖各自的训练/量化流程。如果某种架构需要不同的训练/量化流程，就可能难以甚至无法运行这些最先进的深度神经网络。RAELLA 的设计初衷是提供高效推理，并在不重新训练或修改深度神经网络的情况下避免精度损失。

3 RAELLA：低分辨率，高保真度

为了提高效率，我们希望降低模数转换器分辨率。但如果列和分辨率高于模数转换器分辨率，就会损失保真度和深度神经网络精度。我们确定了三种会产生高分辨率列和的架构权衡：

- 每个模数转换器转换更多切片乘积 → 模数转换器转换次数更少，列和分辨率更高。
- 每个权重切片的位数越多→ 权重列越少，ADC 转换次数越少，列和的分辨率越高。
- 每个输入切片的位数越多 → 输入周期越少，ADC 转换次数越少，列和的分辨率越高。

这里，我们概述 RAELLA 针对这三种权衡采取的策略。首先，我们以采用 512×512交叉阵列和 4 位输入/权重切片的方案作为基线。如图 3 所示，这种设置会产生非常宽的列和分布，范围从零到数万。表示这些列和需要 17 位。RAELLA 的策略会收窄列和分布，直到其可用有符号 7 位范围 $[-64, 64$表示。

为在不损失保真度的情况下覆盖这一 7 位范围，我们将 RAELLA 的模数转换器设置为始终读取列和的七个最低有效位（LSB）。也就是说，如果只有一个交叉阵列行处于开启状态，并产生值为 1 的切片乘积，模数转换器将读取列和并输出

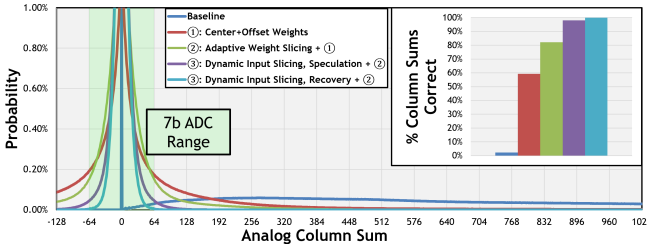

图 3：在 ImageNet 上运行 ResNet18 时，RAELLA 采用各项策略后得到的列和分布。RAELLA 将列和分辨率从 17 位降至 7 位，并将 ADC饱和率从 98% 降至 0.1%。

数值 1。这样的小步长可在范围内的列和保持完整保真度。[4] 缺点是，小步长意味着范围较小；如果列和超出 $[-64, 64)$，模数转换器就会饱和并损失保真度。使用 4 位输入/权重切片时，即使只有一个交叉阵列行，也能产生高达 $(2^4 - 1)^2 = 225$ 的切片乘积，这会在 63 处饱和。RAELLA 必须在一次对 512 行求和时避免饱和。

通过重塑列和分布，RAELLA 的策略将饱和概率降至近乎为零。这正是 RAELLA 能够凭借低分辨率模数转换器实现高保真度的方式：只有当列和较大时，RAELLA 的模数转换器才会损失保真度，而以下策略能够减小列和。对于每项策略，我们报告了在 ImageNet 上运行 ResNet18 时的模数转换器饱和率。

3.1 中心值+偏移权重

3.1.1 问题。标准阻变存储器交叉阵列会计算无符号切片乘积。如果每个切片乘积为 ≥ 0，那么累加大量切片乘积将产生数值较大、分辨率较高的列和。

3.1.2 解决方案。我们将权重平移一个中心值，使得大约一半权重高于中心值，另一半低于中心值。因此，当我们对权重进行切片并在交叉阵列中进行计算时，大约 50% 的切片乘积来自正权重，另有 50% 来自负权重。随后，我们在交叉阵列中对有符号切片乘积求和。正负切片乘积会相互抵消，即使累加了许多切片乘积，列和仍然很小。为了最大限度地利用这种抵消，Center+Offset 选择中心值，使每个交叉阵列列中正负切片的幅值达到平衡。

3.1.3 权衡。RAELLA 以增加交叉阵列面积为代价，在交叉阵列中实现有符号算术。交叉阵列十分密集，因此这种面积权衡值得采用，因为它能降低模数转换器成本。RAELLA 还使用额外的存储空间和低成本数字电路来存储和处理中心值。

3.1.4 结果。采用中心值+偏移权重时，图 3 中标记为 ① 的列和分布是有符号的，并以零为中心。列和分辨率为 $\leq$ 7b 的概率为 59.2%。

[4]这种策略与许多求和保真度受限架构采用的方法形成对比；后者会在计算中舍弃最低有效位 [5, 7, 24] 虽然舍弃最低有效位可以降低饱和概率，但也必然会使每个部分和的保真度下降。

3.2 自适应权重切片

3.2.1 问题。每个权重切片中存储数值，从而提高列和值及其分辨率。

3.2.2 解决方案。如图 7 所示，RAELLA 在编译时自适应地对权重进行切片。通过减少每个权重切片的位数，我们可以降低权重切片中存储数值的平均值，并降低列和分辨率。然而，增加权重切片会增加列数，从而增加存储占用和模数转换器转换次数，因此我们希望尽量减少所用切片的数量。在编译期间，我们会测量保真度损失造成的误差，并通过选择每个权重切片的位数来控制误差，同时尽量减少切片数量。RAELLA 可以为每个切片使用不同的位数，但同一层中的所有权重均采用相同的切片方式。

3.2.3 权衡。RAELLA 在存储密度、模数转换器转换次数和编译时预处理之间进行权衡。所需的电阻式随机存取存储器和模数转换器转换次数会随权重切片数量增加。RAELLA 采用简单的预处理策略，并重复使用深度神经网络权重进行多次推理，以尽量降低预处理成本。

3.2.4 结果。采用自适应权重切片后，图 3 中标记为 ② 的列和分布更加集中。列和分辨率为 $\leq$ 7b 的情况占 82.1%。

3.3 动态输入切片

3.3.1 问题。每个输入切片的位数越多，输入切片的值就越大，从而导致列和的值和分辨率升高。

3.3.2 解决方案。如图 9 所示，RAELLA 会在运行时动态切分输入。RAELLA 可以减少每个输入切片的位数，以降低列和。但这需要更多周期和更多 ADC 转换次数。RAELLA 采用动态策略，通过推测使用每个输入切片位数较多的高效方案。发生大列和饱和错误时，RAELLA 会通过减少每个输入切片的位数来恢复。该方法兼具推测带来的高效率和恢复带来的高保真度。

3.3.3 权衡。RAELLA 在吞吐量与交叉开关能耗之间进行权衡。推测通常用于提升速度，而 RAELLA 的推测则牺牲速度以换取效率。执行推测与恢复都需要额外周期。此外，RAELLA 的交叉阵列在推测与恢复过程中都会消耗能耗。由于交叉阵列具有高吞吐量和高效率，为降低模数转换器开销，这些成本是值得的。

3.3.4 结果。采用动态输入切片后，图 3 中标记为 ③ 的推测与恢复列和分布进一步收窄。在推测周期和恢复周期中，列和分辨率为 $\leq$ 7b 的情况分别占 98.0% 和 99.9%。

3.4 接受保真度损失

3.4.1 问题。尽管采用了 RAELLA 的所有优化，列和分辨率仍可能高于模数转换器分辨率。我们使用 7b 模数转换器，但有 0.1% 的时间会产生 >7b 列和。这会导致模数转换器输出饱和在其最小值/最大值 -64/63，并将错误值传播到部分和。

3.4.2 解决方案。深度神经网络本身具有抗噪性 [21, 46]，因此较高的错误率也是可以接受的。表 4 显示，RAELLA 的保真度误差对各种深度神经网络造成的损失都很低。

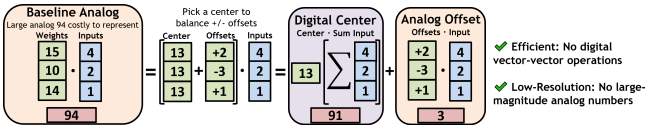

图 4：中心值+偏移量权重。标准点积会产生高分辨率数值，难以在模拟域中表示。中心值+偏移量方法通过数字方式从权重中减去中心值，并在交叉阵列中使用平均值接近零的偏移量进行计算。

RAELLA 使用 512 行交叉阵列和 7 位模数转换器。即使输入切片和权重切片均采用最小的 1 位宽度，列和分辨率也可能达到 9 位，因此无法保证完美保真度。通过采用最小的权重切片尺寸，RAELLA 将模数转换器相关的保真度误差率降至每千万个部分和约出现一次错误，也就是 ResNet50 [16] 每次推理出现一个错误的部分和。

4 实现 RAELLA 的策略 4.1 实现中心值+偏移量权重

如图所示 4，我们将深度神经网络权重表示为中心值加上或减去一个较小的偏移量。我们选择中心值，使每列的正、负权重切片大小大致相同。RAELLA 执行有符号模拟运算，幅值平衡的正、负权重切片产生的切片乘积相互抵消，从而使列和接近于零。这使 RAELLA 能够在累加来自多个交叉阵列行的切片乘积时，保持较小的列和。同时，RAELLA 在数字域中高效处理高分辨率中心值。

我们首先讨论为什么中心值+偏移对于平衡正/负权重切片很重要，接着展示 RAELLA 如何使用中心值+偏移编码进行算术运算，并说明如何计算最优中心值。最后，我们展示使用中心值+偏移编码权重计算点积的硬件。

4.1.1 为什么要平衡切片。虽然深度神经网络权重分布通常被认为是零均值的 [14, 73]，但整个深度神经网络中所有权重值的均值为零，并不意味着任何给定的权重滤波器都是零均值的，也不意味着列和是零均值的。为此，我们需要每个交叉阵列列中的权重切片都具有零均值。实际情况往往并非如此，因为各个权重滤波器以及由切片构成的列会随机收敛到不同的分布。[5]

越来越多的研究正在探索差分编码；与中心值 + [3, 偏移类似，14, 28, 32, 67, 73, 81] 它能够进行有符号模拟运算。差分编码使用正切片表示正权重，使用负切片表示负权重；它可以借助中心值+偏移来平衡正/负权重切片，并降低列和所需的分辨率。

Center+Offset 对于权重切片分布具有明显非零均值的滤波器尤其有益。这种情况可能出现在负权重数量多于正权重的滤波器中，例如图 5 所示的滤波器。差分编码会将这些以负权重为主的权重表示为以负权重为主的切片，从而使每列中的切片均值为负。与数百个切片进行点积后，即使是很小的负均值也会累积，形成较大的负列和。这种效应会显著增加模数转换器饱和，并导致深度神经网络精度损失，如表 4 所示。通过平衡正权重/负权重切片，Center+Offset 可减少每列的偏差，避免精度损失。

4.1.2 中心值+偏移算术。给定一个权重 w 和中心值 ϕ，我们计算正偏移量 $w_+ = max(w - \phi, 0)$和负偏移量 $w_- = max(\phi - w, 0)$。对于高于中心值的权重，w_+是权重与中心值之差，而 w_-为零。对于低于中心值的权重，情况则相反。给定编程为正/负偏移向量 W_+、W_-的权重滤波器W，RAELLA 使用输入向量 I 计算点积，如下所示：

$$W \cdot I = \left(\phi \sum I\right) + (W_+ - W_-)I \quad (1)$$

RAELLA 在运行时计算公式 1，其中 $\phi \sum I$ 以数字方式计算，$(W_+ - W_-)I$ 则以模拟方式计算。

4.1.3 计算最优中心值。我们在编程 RAELLA 之前，通过一次性预处理来计算中心值/偏移量。我们针对每个权重滤波器分别计算中心值，因为不同权重滤波器的权重分布和最优中心值各不相同。

我们定义一个优化问题，以求解中心值 ϕ。首先，我们将切片 S 定义为一系列连续的位索引 $[h \ldots l]$，从最高有效位索引到最低有效位索引，即从 h 到 l（例如，切片 $[7 \ldots 4]$ 包含最高的四位）。然后，我们定义切片函数 $D(h, l, x)$，该函数对有符号数 x 进行裁剪，使其仅包含索引从 h 到 l 的位（移位后，位 l 位于最低有效位位置），同时保留符号。给定权重滤波器 W 和切片 $S_{i \in \{1, 2, \ldots N\}} = [h_i \ldots l_i]$，我们通过以下方式求解 W 的中心值 ϕ：

$$\underset{\phi \in \{1,2,\ldots,255\}}{\arg\min} \sum_{i=1}^{N} 2^{l_i} \left(\sum_{w \in W} D(h_i, l_i, w - \phi) \right)^4, \quad (2)$$

其中，N是切片总数，w是W中的一个权重。式（2）平衡权重切片中每一列的正值与负值，并对非零和较大的列赋予更高代价。内层求和 $(\sum_{w \in W} D(h_i, l_i, w - \phi))^4$ 计算单列的代价，即将该列中权重切片的和提升至四次方。四次方是通过经验选定的；我们发现，幂次过低无法充分惩罚较大的和，而幂次过高则会过分看重和最大的列，无法兼顾所有列。外层求和 $\sum_{i=1}^{N} 2^{l_i}$（.....）根据位位置对代价加权（例如，在一个 8b 数中，最高有效位的位值为 2^7，仅包含这一位的 1b 切片的代价也会乘以 2^7），并对所有列的代价求和。由于高位切片中的饱和对部分和的影响更大，因此代价按位位置加权。

我们针对交叉阵列中的每个权重滤波器（即，单个点积）分别计算中心值。采用更粗的粒度，例如为整个交叉阵列（即，100+个不同的滤波器）仅设置一个中心值，效果就不会那么好，因为不同的深度神经网络滤波器可能具有不同的权重分布，因此需要不同的中心值。

RAELLA 的逐滤波器中心值有一个缺点：每个中心值都需要为一个滤波器中的多列进行平衡，因此可能不适用于其中任何一列。理想情况下，逐列中心值可以精确地将每列权重切片值的平均值归零

[5]我们所说的"滤波器"是指来自一个点积的一组权重，这些权重可容纳在一个交叉阵列中。如果深度神经网络层的某个输出通道（或传统意义上的"滤波器"）的权重无法容纳在一个交叉阵列中，则可将其划分到多个交叉阵列上。重要的是，无论整体深度神经网络的特性如何，每个交叉阵列列都会产生独特的列和分布。为此，中心值+偏移量方法尝试平衡每列中的正/负权重切片。

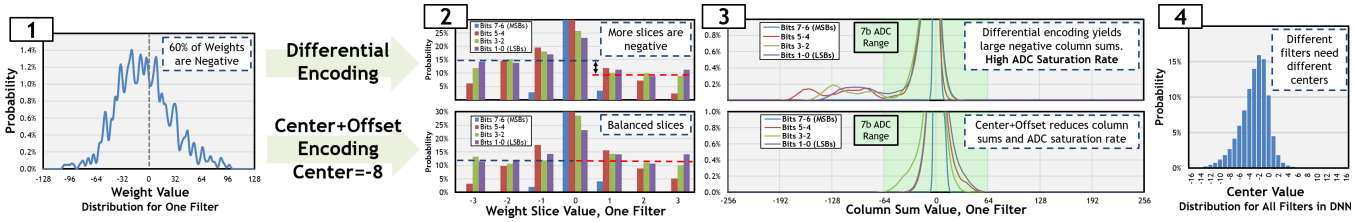

图 5：差分编码与中心值+偏移编码。为说明问题，图中展示了 InceptionV3 [61] 滤波器中负平均权重切片的分布。8 位权重/输入被切分为四个 2 位切片/八个 1 位切片。1 滤波器中的大多数权重为负。2 差分编码用负切片表示负权重，因此该滤波器的切片大多为负。中心值+偏移编码使正负切片趋于平衡。3 由多数为负的切片计算出的点积会产生较大的负列和，从而导致模数转换器饱和。中心值+偏移编码可减小列和。4 每个深度神经网络滤波器都需要不同的中心值。

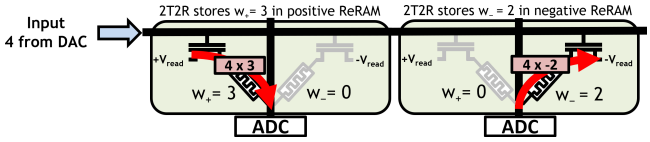

图 6：2T2R 器件在交叉阵列中执行有符号算术。红色表示电流流动的大小和方向。灰显的器件处于关闭状态（设为高电阻状态）。

。然而，这种方法受到整数精度中心值的限制。考虑一列切片的平均值为 0.4 的情况。我们可以将该列中的所有权重切片平移 −1，但这会使平均值变差，将其移至 −0.6。相反，我们在切片之前平移全精度权重值，这样可以重塑（而非平移）每个切片的数值分布。这种分布重塑可以对每列权重切片值的平均值进行更小幅度的调整。

4.1.4 中心值+偏移量的硬件实现。根据公式 (1) 计算部分和需要两个项。第一项，即输入和 $\phi \sum I$，通过数字方式计算。交叉阵列列共享输入向量，因此输入和的计算开销可分摊到各列。

第二项是与偏移量进行向量与向量乘法。$(W_+ - W_-)I$ 由交叉阵列以模拟方式计算。RAELLA 使用图 6 所示的 2T2R器件，在交叉阵列中实现模拟减法。6 每个权重对应两个阻变存储器（2R），并通过两个访问晶体管（2T）进行访问；这种 2T2R 结构已被探索用于表示有符号权重 [3, 27, 28, 67, 74]。其中一个阻变存储器器件连接到正电源，另一个连接到负电源，使 2T2R单元能够向列和中加上或减去数值。对于每个权重，我们将正偏移量/负偏移量 w_+/w_- 写入两个阻变存储器中。对于任意给定的权重，其中一个偏移量为零，因此每对中只使用一个阻变存储器器件。新增的阻变存储器和访问晶体管会增大 RAELLA 的交叉阵列尺寸，但交叉阵列规模较小，因此系统面积仅增加 ~ 10%。

4.2 实现自适应权重切片

自适应权重切片会尽量减少每个深度神经网络层所用的权重切片数。它会在不造成过多保真度损失的前提下，尽可能增加每个切片所用的位数。图 7 展示了 RAELLA 可用的各种切片方式。每个切片的位数越多，每个权重所需的切片就越少，存储密度越高，

[6]也可以使用电路进行模拟减法 [20] 1T2R [81], 和 SRAM [40, 78]。

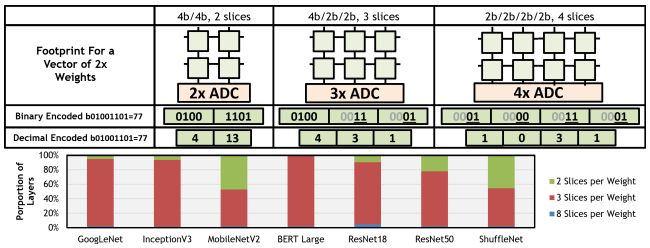

图 7：（上）权重切片交叉阵列占用。（下）DNN逐层权重切片方案。增加切片数会降低列和及饱和概率，但会增加转换/*MAC*。大多数层使用每个权重三片切片。

模数转换器转换次数也越少；但位数增加也会使每个权重切片中存储的值变大，并提高列和需要高分辨率的概率。RAELLA 可以为不同切片使用不同的位数，但同一层中的所有权重都采用相同的切片方式。

比特密度，即某个位为 1 的概率，会影响权重切片中的数值。图 8 展示了典型深度神经网络层中深度神经网络输入和权重的逐比特密度。输入值通常呈右偏分布，因此高位较为稀疏。权重值通常近似呈钟形曲线。使用中心值+偏移编码表示时，高位也会因此变得稀疏。由于高位权重比特具有稀疏性，在大多数层中，4b 权重切片可以存储权重中数值较低、列和较低的最高 4b。低位权重比特密度更高，通常需要每个权重切片采用较低的 2b。图 7 展示了深度神经网络各层的切片方案。大多数层采用 4b-2b-2b 配置，共使用三个权重切片：一个权重切片存储最高位的四个比特，另外两个权重切片各存储两个低位比特。

4.2.1 误差预算。 我们可以选择权重切片来尽量减少饱和，但这种做法过于保守。深度神经网络可以容忍一定的误差，因此我们希望允许少量饱和。不幸的是，很难预测切片会如何影响饱和，以及饱和会如何影响深度神经网络层输出的误差。列和过大会导致饱和，而列和受输入/权重分布、输入/权重切片分布、分布之间的相关性、每个滤波器的权重数以及随机方差影响。此外，16b 部分和会经过数字量化，转换为 8b 输出 [82], , 这可能会放大或缩小误差。

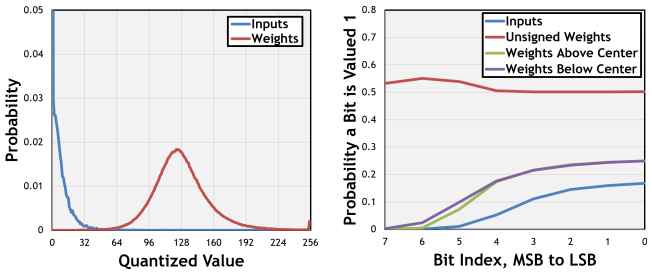

图 8：（左）未进行切片时深度神经网络输入/权重的数值分布；（右）逐位密度。图中展示了 ResNet50 的倒数第二层，代表典型的深度神经网络层。呈钟形分布的权重可以围绕中心拆分为两个相似的分布，其高位较为稀疏。无符号输入的高位天然稀疏。

为综合考虑所有这些因素，我们采用经验方法。我们将误差预算 定义为一层的输出完全计算完毕并量化为 8b 后，非零输出所允许的平均幅值误差。计算平均幅值误差时只统计非零输出，从而使具有不同输出稀疏度的层之间的计算更加一致。[7]

为计算误差，我们从验证集中随机选取十个输入。使用如此少量的输入便已足够，因为改变切片可能使误差变化一个数量级甚至更多，而且很容易检测到这些差异。[8] 数量级上的差异源于列和分布的形状（图 3）。分布尾部呈指数级收缩，因此，分布宽度的变化（即由切片导致的变化）会对饱和率产生指数级影响。

在我们的测试中，误差预算设为 0.09，这相当于平均每 11 个 8 位输出中就有 1 个偏差 1。量化后，模数转换器饱和造成的误差通常较小，因此精度损失也较低，如表 4 所示。

4.2.2 选择权重切片。权重切片通过算法1所示的预处理过程计算得出。将深度神经网络编译为 RAELLA 时，预处理只需执行一次；在 Nvidia RTX2060 GPU 上，每层耗时 10–1000 毫秒。预处理后，经过切片编码的+权重会被写入交叉阵列，可用于任意次数的推理。

对于 M 位权重和每个阻变存储器最多 N 位的情况，我们将切片定义为一个整数元组 $1 \le s_0..s_j \le N$，使得 $\sum s_i = M$。对于 8 位权重，若每个阻变存储器为 ≤ 4 位，切片方案包括 (4 位,4 位)、(2 位,1 位,1 位,4 位) 和 (1 位,2 位,2 位,3 位)。总共有 108 种切片方案。

为找到深度神经网络层的最佳切片方式，我们遍历全部 108 种切片方案。对于每种方案，我们按照第4.1节对权重进行 Center+Offset 编码，使用十个测试输入模拟交叉阵列，并记录误差。我们选择使用切片数最少且误差低于预算的方案。对于权重切片数相同的方案，则选择误差较低的方案。

比较权重切片方式时，我们使用 1b 输入切片。对于最后一个权重切片，始终使用最保守的每权重 1b 方式

[7]当 ReLU 被折叠到量化中时，这一点很重要。如果 ReLU 将某个输出置零，那么它很可能也会将与该输出相关的误差置零。如果 ReLU 将许多输出置零，那么平均误差会降低，而每个非零输出的误差则保持不变。[8]事实上，仅用一个输入进行测试时，该算法通常会选出相同的切片方案。若以高斯噪声作为输入进行测试，则有 > 90% 的层会得到相同的切片方案。

层。最后一层对深度神经网络准确率的影响格外显著 [6]，而最后一层采用效率较低的切片方式，对整体能耗/吞吐量的影响很小，因为深度神经网络的性能主要受中间层支配（图 7）。

算法1：权重切片与中心值预处理

```
1  函数 切片编码权重(层、测试输入、误差预算)
       /* 深度神经网络层预处理。需要一个层（形状、
          量化、 权重）、 测试输入（激活值
          来自本文中的十张图像/令牌），以及一个
          标量误差预算（本文中为 0.09）。 */
       切片 = FindBestSlicing(层, 测试输入, 误差预算)
2      中心值 = FindOptimalCenters(层, 切片)
3      返回 切片, 中心值
4
5  函数 FindBestSlicing(层, 测试输入, 误差预算)
       /* 第 4.2 节中自适应权重切片的实现
          每层耗时 10–1000 毫秒。 */
6      预期输出 = 层.Run(测试输入)
7      可能的切片方案 =GetAllPossibleSlicings()
8      最佳切片方案 = 可能的切片方案[0]
9      最佳切片数 = CountSlices(最佳切片方案)
10     最佳误差 = ∞
11     对于 slicing ∈ possibleSlicings 执行
12         中心值 = FindOptimalCenters(层, 切片)
13         输出 = 层。模拟交叉阵列(测试输入, 切片,
             中心值)
14         误差 = |预期输出 - 输出|
15         平均误差 = 平均值(误差[预期输出！= 0])
16         切片数 =CountSlices(切片)
17         更好的切片 = 切片数 < 最佳切片数 ||
             (切片数==最佳切片数 && 平均误差 < 最佳误差)
18         如果 平均误差 <误差预算 && 更好的切片 那么
19             最佳切片 = 切片
20             最佳切片数 = 切片数
21             最佳误差 = 平均误差
22     返回 最佳切片
23
24 函数 FindOptimalCenters(层，切片)
       /* 求解中心值+偏移量方程 (2)。 <1毫秒/层。
          为每个权重滤波器返回一个中心值。 */
25     中心值 =SolveOptimizationProblem(层，切片)
26     return 中心值
```

4.2.3 硬件中的自适应切片。给定 4b 电阻式随机存取存储器，每个器件都可以编程为 $2^4 - 1$ 个模拟电平。为了编程 3b 或 2b 切片，我们使用最低的 $2^3 - 1$ 或 $2^2 - 1$ 个电平。给定一个 3b 权重切片 XXX，这相当于将器件编程为 0XXX。这只是对可用范围的限制，因此不需要更改电阻式随机存取存储器。交叉阵列本来就需要移位+加法电路，以跨权重切片和输入切片累加列和；自适应切片只需更改移位+加法模式。

主要开销取决于权重切片的数量。每增加一个权重切片，就需要更多阻变存储器和 ADC 转换次数。RAELLA 可使用两个权重切片（每个切片 4 位，效率最高）到八个权重切片（每个切片 1 位，效率最低）。大多数层使用三个权重切片。

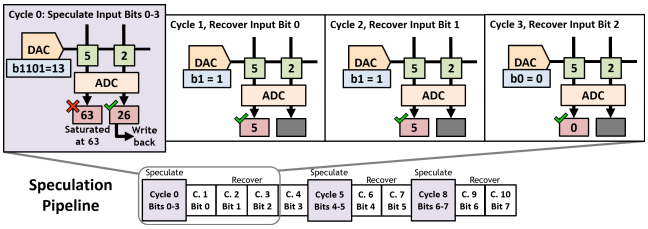

图 9：推测计算。推测周期中，每个输入切片使用 2–4 位，每个输入的模数转换次数较少，但饱和概率较高。恢复周期中，每个输入切片使用 1 位，并且模数转换器仅处理推测失败的列。

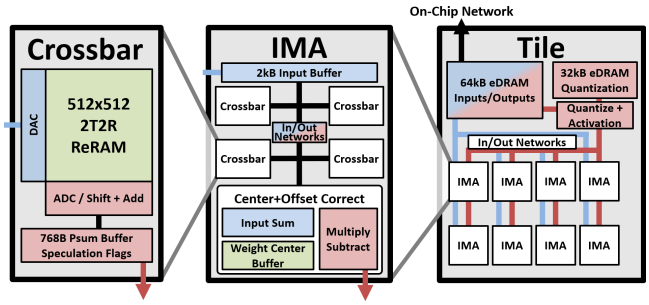

图10：RAELLA 架构。(1) 基本单元是一个交叉阵列。(2) 四个交叉阵列组成一个 IMA。(3) 八个 IMA 组成一个瓦片。输入存储/处理组件标为蓝色，权重组件标为绿色，输出组件标为红色。

4.3 实现动态输入切片

动态输入切片在高效率的多位输入切片与高保真的少位输入切片之间取得平衡。我们希望尽量减少输入切片的数量，从而减少 ADC 转换次数，同时避免因高分辨率列和而造成的保真度损失。与权重不同，输入切片可以在运行时更改。这使我们能够先采用高效且激进的切片方案进行推测，再采用保守的切片方案进行恢复。在推测阶段，RAELLA 使用三个输入切片，效率较高，但产生较大、高分辨率列和的可能性也较高。在恢复阶段，RAELLA 使用最保守的方案，即八个 1 位输入切片。

推测与恢复的流程如图 9 所示。首先，将一个 4 位高位切片推测性地送入交叉阵列，并通过模数转换器转换列和。如果列和过大，就会在模数转换器边界 $[-64, 64)$ 处饱和。如果模数转换器输出等于这两个边界中的任意一个，就会检测到错误，并将其标记为推测失败。接下来，所有列处理完毕后，将 4 位输入切片重新切分为 1 位切片，并在四个恢复周期中再次处理。为节省恢复过程中的能耗，对推测成功的列关闭模数转换器的电源。在极少数情况下，如果模数转换器在恢复期间发生饱和，我们会接受保真度损失，并传递饱和值。处理完四个恢复切片后，流程会针对后续的推测与恢复周期重复执行。

4.3.1 硬件中的动态输入切片。 给定一个 4 位数模转换器，模拟输入切片可以采用 $2^4 - 1$ 个模拟电平中的任意一个。对于 N 位输入切片，我们可以使用最低的 $2^N - 1$ 个电平。给定一个 3 位输入切片 XXX，这相当于转换 0XXX。由于这只是对可用范围的限制，因此不需要更改数模转换器硬件。

为了跟踪成功/失败的推测，RAELLA 会为每个交叉阵列在缓冲区中存储推测成功标志。在恢复过程中，模数转换器只转换推测失败的列和。

整个阻变存储器交叉阵列是一个单元，因此所有列会一起进行推测和恢复。由于至少有一列推测失败的可能性很高，交叉阵列总是会执行恢复。因此，RAELLA 的推测以速度为代价节省能耗（不同于推测通常用于提升速度的情况，例如 CPU分支预测）。

由于所有列和 ReRAM 都会运行推测与恢复周期，推测也会增加交叉阵列能耗。恢复周期的能耗低于推测周期，因为忆阻式随机存取存储器器件在输入值较小时能耗更低 [29]，而且在恢复周期中，模数转换器只处理少数列。

4.3.2 动态输入切片系统影响。RAELLA 可以在不进行推测的情况下运行，仅处理八个恢复切片。采用这种方法时，每列需要进行八次 ADC 转换，以处理全部八个输入切片。采用推测时，则只需进行三次 ADC 转换，以处理三个 2-4b 推测输入切片。在我们的各个基线中，推测约有 2% 的情况会失败，根据失败的推测切片不同，需要进行 2-4 次恢复转换。总体而言，推测的成功率为 ~ 98%，与仅恢复的方法相比，可将 ADC 转换次数减少 ~ 60%。处理每列平均需要三次推测转换 + 0和 3 次恢复转换。

RAELLA 通过推测减少 ADC 转换次数，但代价是吞吐量和交叉开关能耗。RAELLA 的交叉阵列需要 11 个周期，才能运行全部三个推测切片和八个恢复切片。另一种方案是不进行推测，只运行八个恢复切片，这样可以提高吞吐量，但也会增加所需的 ADC 转换次数。

5 RAELLA架构与流水线

RAELLA 的高层架构如图1所示。RAELLA 的组织方式大体沿袭 ISAAC [54]。我们将自底向上介绍 RAELLA 架构，展示 RAELLA 的数据流，随后说明 RAELLA 如何减少模拟非理想效应。

5.1 交叉阵列

交叉阵列由 512×512 2T2R 构成。每个交叉阵列都编程写入来自一个深度神经网络层的权重，并且根据深度神经网络层切片（第4.1节），每个权重滤波器使用 2–8 列交叉阵列。

为处理输入，我们使用 4 位脉冲序列数模转换器，因其硬件简单 [32] 且线性度优越 [55]。脉冲序列数模转换器通过最多 $2^N - 1$ 个脉冲来编码 N 位输入切片。该数模转换器由一个用于施加输入脉冲的简单行驱动器、一个用于存储当前输入位的 1 位触发器，以及一个用作使能信号的与门组成 [32]。要输出一个 4 位值，首先将最高有效位载入触发器，然后由全局时钟生成八个 1 ns 脉冲。数模转换器输出时钟与其存储值进行与运算后的结果：若该位为 1，则输出八个脉冲；若为 0，则输出零个脉冲。随后依次载入后续位，分别运行四个、两个和一个脉冲。

数模转换器激活 2T2R 的访问晶体管，每个双晶体管双电阻器件都会计算一个切片乘积，并将其加到列和中或从列和中减去

。列和以电流的形式出现在列上，经电流缓冲器缓冲并缩放 [24]，随后被捕获为电容器电压，并由采样+保持电路 [38]保持。

接下来，四个 7b 模数转换器 [23] 在 100ns 内将 512 个列和转换为数字信号 [54]。7b 有符号模数转换器的结果由移位加法电路 +求和，并累加到 16b 部分和缓冲器 [82]中。

采用每个权重两个切片的最高密度切片方式时，一个交叉阵列最多可产生 256 个部分和，这些部分和存储在一个包含 256 个条目的部分和缓冲器中。每个条目存储一个 16b 部分和 + 8 个成功标志，因而部分和缓冲器的总容量为 768B。

在推测/恢复周期中，输入会针对每个周期分别流入交叉阵列。在推测阶段，模数转换器处理所有列。如果某个模数转换器发生饱和，则不更新部分和，并标记成功标志。在恢复阶段，会检查所有成功标志。模数转换器仅处理并写入推测失败的列的结果。

交叉阵列周期采用两级流水线 [54]。第一级中，数模转换器提供输入脉冲，交叉阵列计算模拟列和，并将结果锁存到采样+保持电路中。采用 1 ns/1 ns 开/关脉冲宽度的 4 位脉冲串数模转换器，可在 30 ns 内发送最多 15 个输入脉冲。交叉阵列在不到 1 纳秒的时间内稳定并产生列和 [17]。第二级中，模数转换器在 100 ns 内转换采样 + 保持电路的结果 [54]。由于较慢的模数转换器处理阶段，交叉阵列周期的总时长为 100 ns。

RAELLA 利用输入位稀疏性来降低列和的数值及交叉开关能耗，并受益于无符号输入的高位稀疏性（图 8）。如果输入为有符号数，RAELLA 会分两个周期分别处理正输入和负输入，以产生稀疏性。

5.2 原位乘累加

四个交叉阵列与一个输入缓冲区 [54]组成原位乘累加（IMA）单元。输入网络将输入向量发送至交叉阵列；如果两个交叉阵列共享所有输入，则会对输入向量进行组播。为了利用时间输入复用 [47, 59],，输入缓冲区会在交叉阵列周期之间存储复用的输入。四个交叉阵列可在所有行上处理最多 $4 \times 512 = 2048$ 个输入，因此缓冲区大小为 2 kB。

为支持中心值偏移权重，每个 IMA 都包含一个权重中心缓冲器和数字加法电路，用于计算输入和。每个交叉阵列都维护一个累加和。为利用输入复用 [47],，我们在输入首次用于交叉阵列列时将其加到和中，并在最后一次使用时将其减去。如果不同的交叉阵列列使用不同的输入子集，RAELLA 会在处理各列时以流式方式加减输入。

5.3 瓦片

八个 IMA 组成一个瓦片。每个瓦片都包含一个 64kB 的嵌入式动态随机存取存储器缓冲区 [54]，用于存储 8 位输入/输出、数字最大池化单元和量化电路。RAELLA 以数字方式执行 8 位逐通道量化 [82],，为每个输出通道分配 32 位来存储一个半精度浮点数量化比例因子和偏置 [82],，即每个瓦片 32kB。

5.4 加速器与编程

与 ISAAC [54], 类似，每四个瓦片共用一个路由器，以实现片上通信。当一个瓦片完成一组输出后，会通过路由器将数据发送到下一个瓦片。如果一层的权重超出单个瓦片的存储容量，这些权重就会拆分到多个瓦片上。

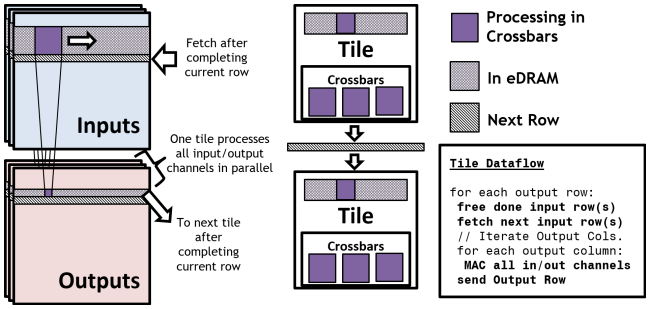

图11：数据流。每次计算某一层的一行输出。瓦片各接收/发送一次输入/输出。

与其他存内计算加速器 [24, 47, 54, 80], 一样，RAELLA 只需编程一次，即可执行多次推理，从而降低阻变存储器较高的写入能耗 [45]。编译 RAELLA 的配置时，我们会对中心值 + 偏移和自适应权重切片进行轻量级预处理，如第4.2.2节所述。

5.5 深度神经网络数据流

每个深度神经网络层若能容纳，就映射到一个交叉阵列上。否则，就会扩展到更多交叉阵列、IMA 和瓦片上。RAELLA 的层间数据流遵循 ISAAC 的 [54]，以尽量减小 eDRAM 占用和瓦片间通信需求。图11展示了 RAELLA 的数据流。深度神经网络层在并行瓦片上以流水线方式运行。瓦片每次生成一层输出张量的一行，复用先前使用过的输入行，并仅获取新的输入行。当瓦片从上到下生成输出张量的各行时，输入行也按照相同顺序从前一个瓦片中读取。通信和数据复用模式由模式生成器协调，并在程序运行时固定。在瓦片层级以下，使用 Timeloop [41] 来寻找最优的数据复用模式。

RAELLA 遵循先前工作 [24, 54, 56]，通过复制权重来提高吞吐量。如果空间充足，就会在交叉阵列内复制权重，以使用部分托普利兹展开计算多个卷积步骤 [11, 24]。权重还可以进一步跨交叉阵列、IMA 或瓦片复制。复制采用贪心方案：只要还有空余瓦片，就复制吞吐量最低的层。

5.6 RAELLA 降低模拟非理想效应

存内处理交叉阵列可能会受到红外压降和潜行电流等非理想效应的影响。与 ISAAC 相比，RAELLA 能够降低这些影响。

较大的电流流经较长的交叉阵列列时会产生红外压降，从而导致精度损失 [7, 75]。正/负 2T2R 器件会从相邻器件汲取电流，从而降低红外压降 [28, 64, 81]。此外，当处于最高电导状态的阻变存储器少于五个时，RAELLA 的模数转换器就会饱和。因此，与类似 ISAAC 的设计需要汇总 128 个阻变存储器的电流相比，RAELLA 的列只需承受来自五个阻变存储器的电流。

潜行电流，即流经关断状态阻变存储器的漏电流，可能导致精度损失 [7]。在 2T2R 交叉阵列中，潜行电流为零，因为正、负阻变存储器的漏电流会相互抵消 [81]。

6 评估

RAELLA 与加速器 ISAAC [54], FORMS-8 [80], 和 TIMELY [24]进行比较。ISAAC 无需 DNN重新训练。FORMS

是权重数量受限型，而 TIMELY 是求和保真度受限型，因此两者都需要重新训练来恢复深度神经网络准确率。

首先，我们通过比较 RAELLA 与 ISAAC 的能耗和吞吐量，展示 RAELLA 在无需重新训练的设置下带来的效率和吞吐量提升。我们表明，RAELLA 无需更改深度神经网络模型，即可实现高吞吐量和高效率。

接下来，我们通过将 RAELLA 与 FORMS 和 TIMELY 进行比较，展示它与需要重新训练深度神经网络的架构相比具有竞争力。我们表明，无需重新训练，RAELLA 也能达到这些架构的效率和性能。

然后，我们展示 RAELLA 的低精度损失，并将其与 FORMS 和 TIMELY 进行比较。我们还展示 RAELLA 的中心值+偏移编码在精度方面的优势。

6.1 方法

RAELLA、ISAAC 和 FORMS 的模型使用 Accel- ergy/Timeloop [41, 71, 72] 在 32 纳米工艺节点上构建。这些架构经过修改，以支持第 6.1.2 节所述的 8b 深度神经网络。在 $600mm^2$ 面积预算下，RAELLA 可容纳 743 个瓦片，而 ISAAC 和 FORMS 各可容纳 1024 个瓦片。TIMELY 的结果来自其原始论文 [24]。为与 TIMELY 进行比较，我们将 RAELLA 缩放至 TIMELY 的 65 纳米工艺节点，并在 RAELLA 中采用 TIMELY 的模拟组件（TDC、电流加法器、充电+比较器）和 ReRAM器件 [13] 。在所有测试中，RAELLA 的误差预算均设为 0.09。

6.1.1 组件模型。SRAM 使用 CACTI [18]进行建模。网络、路由器和嵌入式动态随机存取存储器缓冲区的模型来自 ISAAC [54]。由于瓦片消耗数据的速度快于刷新周期，因此嵌入式动态随机存取存储器刷新并不是问题 [63]。RAELLA 使用来自 ISAAC 的模数转换器 [23] ，并根据 [52] 将其缩放至 7b。数模转换器、输入驱动器以及交叉阵列的面积和能耗使用修改后的 NeuroSim [2, 44]生成。2T2R 的面积按两个电阻式随机存取存储器和两个最小尺寸晶体管的面积之和进行保守估算，忽略芯片层之间可能存在的堆叠 [27, 66]。数模转换器为每一行使用一个触发器和一个与门来生成脉冲序列 [32], ，其中每个脉冲为 1ns，每个 4b 输入切片最多可包含 15 个脉冲。ReRAM 参数取自 TIMELY [5, 24], ，采用 0.2V 的读取电压以及 1kΩ/20kΩ 的开关态电阻 [13, 17]。用于捕获模拟列和的电流缓冲器取自 TIMELY [24]。输出通过乘法/截断进行量化，激活函数则融合到量化过程中 [82]。最大池化单元和采样电容消耗的能耗和面积都小到可以忽略不计 [24, 54]。一个交叉阵列周期为 100ns，交叉阵列每 11 个周期生成一组部分和（包括三个推测输入切片 + 和八个恢复输入切片），除非受到层间数据流的瓶颈限制。有符号输入的延迟会加倍，因为正负输入在分离的周期中进行处理。在禁用推测的情况下，交叉阵列需要 8 个周期、即 800ns，才能生成一组部分和。

6.1.2 ISAAC 和 FORMS 的模型。我们根据论文中给出的结果验证了 ISAAC [54] 和 FORMS [80]模型，能耗和吞吐量误差为 $< 10\%$。验证后，为了进行公平的一对一比较，我们使用 RAELLA 中采用的相同组件对 ISAAC 和 FORMS 建模。具体而言，我们使用经过修改的 NeuroSim [2, 44] 对数模转换器/交叉阵列建模，以捕捉模拟组件的数据相关能耗。我们修改了这两种架构，并添加量化硬件以运行 8b 深度神经网络。扩展到 8b 后，我们的 ISAAC 基线具有 $\sim 4\times$ 比原始 ISAAC 更高的效率和吞吐量，而我们的 FORMS 基线则比原始 FORMS 高出 $\sim 2\times$ 。对于 FORMS，我们采用报告过的最高剪枝率。为进行公平比较，我们修改 ISAAC，使其支持 RAELLA 所支持的部分托普利兹映射 [11, 24] ，这使 ISAAC 的吞吐量额外提升了 $1 - 1.9\times$。这些映射对 FORMS 并无益处。

6.2 深度神经网络模型与测试集

我们在其中共有三类的深度神经网络上进行测试：其中一种是来自 PyTorch [42] Torchvision [31] 量化库的卷积神经网络：GoogLeNet [60],InceptionV3 [61], Resnet18 [16], ResNet50 [16], ShuffleNetV2 [30], 和 MobileNetV2 [53]。ShuffleNetV2 和 MobileNetV2 是采用小型滤波器的紧凑型模型，而其他模型则较大。我们报告了 ImageNet [10] 验证集上的精度。

此外，我们在斯坦福问答数据集 [49] 上测试 Transformer [64] BERT-Large [70]，以展示 RAELLA 在尖端 Transformer 上的有效性。对于 BERT-Large，我们加速其前馈层。其他研究则探索了加速 Transformer 注意力机制 [39, 58, 77]。BERT-Large 展示了 RAELLA 在采用非 ReLU 激活函数和有符号输入时的性能。

6.3 效率与吞吐量：无需重新训练

RAELLA 在运行所有深度神经网络的现成模型时进行了评估，并与 ISAAC 进行了比较。图12展示了效率和吞吐量结果。RAELLA 将能效提升了 2.9 至 $4.9\times$（几何平均值为 $3.9\times$）。能效提升主要来自模数转换器能耗的降低。RAELLA 使用 7b 模数转换器，而 ISAAC 使用 8b 模数转换器。此外，RAELLA 采用更大的交叉阵列、每个输入切片/权重切片包含更多位，并通过推测将模数转换器转换次数减少了 5 至 $15\times$。

RAELLA 的吞吐量优势来自更大的 512×512（ISAAC 为 128×128）以及更密集的每个权重切片 2-4 位（ISAAC 的每个权重切片为 2b）。更大的权重存储容量和更密集的权重存储与计算使 RAELLA 的吞吐量提升了 0.7 至 $3.3\times$（几何平均值为 $2.0\times$）。

在不使用推测的情况下，RAELLA 仅运行恢复切片；由于模数转换器能耗较高，相对能效优势降至 $2.8\times$ 几何平均值。相对吞吐量则增至 $2.7\times$ 几何平均值，因为交叉阵列无需运行三个推测切片，且部分和在八个交叉阵列周期内即可计算完成，而非十一个周期。

RAELLA 在使用无符号输入和较大深度神经网络时效果更好。有符号输入（例如 BERT 中的输入）需要分周期处理，因此吞吐量提升较小；而 ShuffleNet 和 MobileNet 中的小型滤波器无法充分利用 RAELLA 的大型交叉阵列。

6.4 与重新训练架构的比较

我们展示 RESNET18/ResNet50 结果的比较将 RAELLA 与 TIMELY 和 FORMS 进行比较。我们展示 RESNET18/ResNet50 结果的几何平均值，因为所有基线都提供了这些深度神经网络的数据。RAELLA 运行现成模型，而 FORMS [80] 运行剪枝后重新训练的版本，TIMELY [24] 运行重新量化后重新训练的版本。

图 13 比较了 RAELLA 与 FORMS 和 TIMELY 的效率/吞吐量。RAELLA 能够达到与 FORMS 相当的吞吐量，并且效率超过 FORMS 和 TIMELY。在与 TIMELY 的比较中，我们发现 65nm RAELLA 无需推测即可实现更高的效率。这是因为 65nm-RAELLA 使用了 TIMELY 的模拟组件，包括 TIMELY 高效的模数转换器。推测

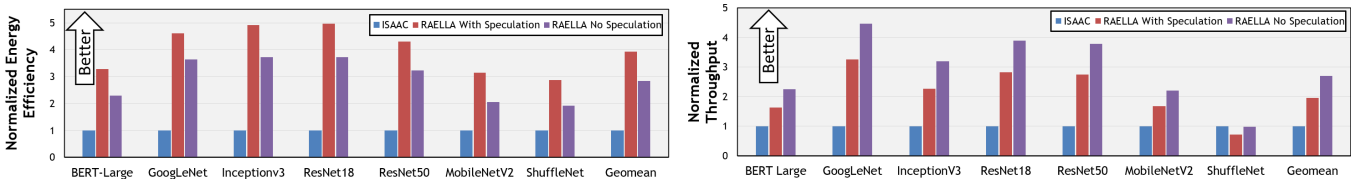

图12：相对于 ISAAC架构归一化的效率和吞吐量。两种架构均可在无需重新训练的情况下运行深度神经网络。采用/不采用推测的 RAELLA，其效率提升 3.9×/2.8×，吞吐量提升 2.0×/2.7× 几何平均值。

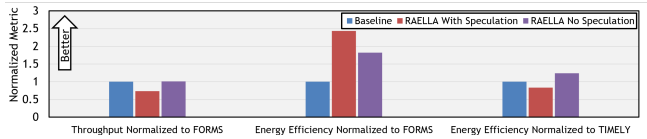

图13：与 FORMS 和 TIMELY 的比较。FORMS 和 TIMELY 运行经过重新训练的深度神经网络。RAELLA 无需重新训练即可实现有竞争力/更优的吞吐量/效率。

	RAELLA 中心值+偏移量	RAELLA 零+偏移	FORMS [80]	TIMELY [24]
重新训练	**No**	**No**	**Yes**	**Yes**
准确率下降 %。负值表示精度提升。				
ResNet18	0.06	0.16	0.62	≤ 0.1
ResNet50	-0.08	0.30	0.70	≤ 0.1
MobileNetV2	0.03	10.17	-	-
ShuffleNetV2	0.14	16.36	-	-
GoogLeNet	-0.02	1.53	-	-
InceptionV3	-0.03	3.72	-	-
BERT-Large	0.12	0.46	-	-

表 4：精度比较。BERT-Large 比较 F1损失，其他模型则比较 ImageNet Top-5损失。Zero+Offset 会导致较高的精度损失；Center+Offset 对于保持精度至关重要。FORMS 和 TIMELY 需要重新训练，而 RAELLA 无需重新训练即可将精度损失维持在较低水平。

当模数转换器成本占主导时，这种做法很有用；但如果模数转换器对总体能耗的贡献不大，这些权衡可能并不值得。

6.5 精度比较

我们将 RAELLA 与三个基线进行比较。RAELLA Center+Offset 是标准的 RAELLA，配置了 0.09 的误差预算。为了展示 Center+Offset 的优势，我们将其与 RAELLA Zero+Offset 进行比较；后者通过将中心值设为零，实现了一种常见的差分编码方式（见第4.1节）。我们对 RAELLA Center+Offset 和 RAELLA Zero+Offset 使用相同的切片，以匹配效率和吞吐量。此外，我们还展示了 FORMS 和 TIMELY 在重新训练后的报告精度。

表 4 展示了精度结果。采用 Center+Offest 编码的 RAELLA 几乎不会造成精度损失。正如第4.1节所述，Zero+Offset（差分编码）由于模数转换器饱和率较高，会导致显著的精度下降。由于滤波器权重分布各异，Zero+Offset 带来的精度下降在不同深度神经网络之间差异很大。TIMELY 和 FORMS 通过重新训练深度神经网络来弥补精度损失。

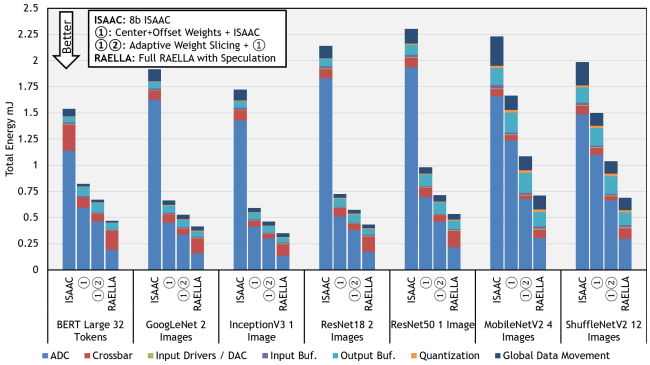

图14：能耗消融。RAELLA 的每种策略都能提高存内计算架构的效率。针对不同深度神经网络调整批量大小，使总体能耗保持在相近范围内。

7 消融研究

为单独考察 RAELLA 各项策略的影响，我们从 ISAAC 架构开始，依次应用这些策略。在能耗消融中，我们测试 RAELLA 各项策略带来的效率收益。在精度消融中，我们测试 RAELLA 的策略在模拟噪声逐渐增大时的表现。所有测试配置均保持高保真度。四种测试配置如下：

- ISAAC：8 位 ISAAC。 128×128 交叉阵列，无符号算术。4 个 2 位权重切片，8 个 1 位输入切片。8 位模数转换器。
- 中心值+偏移：采用前述设置，另将交叉阵列尺寸增大至 512×512 2 T2R，并使用中心值+偏移算术。模数转换器的分辨率降低至 7 位。

- 中心值+偏移、自适应权重切片：采用前述设置，另按照第4.2.2节为每层选择权重切片方式。大多数层采用 4 位、2 位、2 位的模式，使用三个权重切片。
- RAELLA：采用前述设置，另启用动态输入切片和推测。加入 RAELLA 的寄存器/网络。RAELLA 运行一个 2 至 4 位的推测输入切片，随后运行 2 至 4 个 1 位的恢复输入切片。在恢复周期中，对于推测成功的列，模数转换器不会进行转换。

7.1 能耗消融

Fig. 14 展示了以下结果：

- ISAAC：模数转换器占总体能耗的大部分。每次乘加运算的转换次数 为 0.25。各组件的能耗分解因深度神经网络的输入／权重值、交叉阵列利用率以及数字数据传输需求而异。

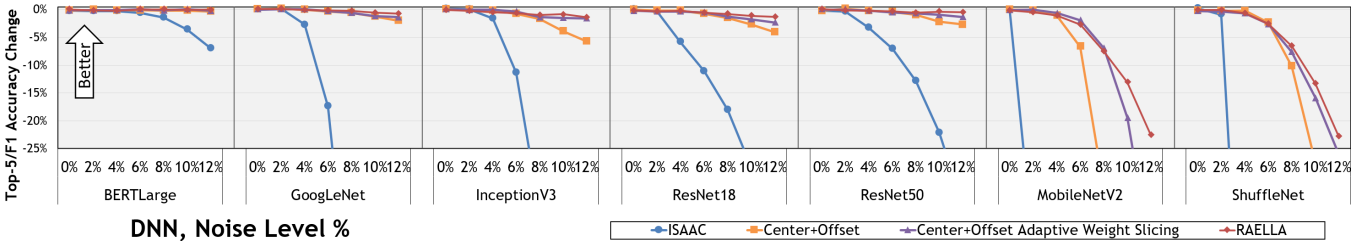

图 15：模拟噪声增加时的准确率下降。中心值+偏移和自适应权重切片提高了噪声容限。动态输入切片即使在推测失败时也能保持准确率；恢复机制可避免精度损失。

• 中心值+偏移：可实现交叉阵列行／列的 4× 扩展，并降低模数转换器分辨率。中心值+偏移的位稀疏性可降低交叉开关能耗。较大的交叉阵列可降低数据搬运能耗，并将每次乘加运算的转换次数 从 0.25 降至 0.063。数字中心值处理只需每几百次乘加运算进行一次输入加法和一次乘／减运算，因此能耗可以忽略不计。

• 自适应权重切片：由于大多数层使用三个而非四个权重切片，模数转换器能耗降低 ~ 25%。每次乘加运算的转换次数 降至 0.047。

• 推测：模数转换器能耗降低 60%。由于推测周期，交叉阵列/数模转换器能耗增加。由于 2× 次读取，输入缓冲区能耗增加。由于部分和回写次数减少，输出缓冲区能耗通常会降低。每次乘加运算的转换次数 为 0.018。

7.2 准确率与交叉阵列内噪声消融分析

使用相同的四种消融设置，我们评估了不同噪声水平下 RAELLA 上运行的深度神经网络准确率。所有存内处理架构都会受到模拟变化和噪声的影响，但 RAELLA 的噪声容限更高，精度损失更低。

我们将变化和噪声建模为高斯分布，并将其加到列和 [17]上。给定正向/负向切片乘积和 N_+ 和 N_-，我们将列和建模为 $N(\mu,\ \sigma^2)$。对于噪声水平 E，我们将均值 μ 设为理想列和（$N_+ - N_-$），标准差设为 $\sigma = E\sqrt{N_+ + N_-}$。计算出列和后，会将其发送至模数转换器；若超出范围，则会在 $[-64, 64)$处饱和。噪声会在正向/负向切片乘积之间累加。我们测试的误差最高为 12%，即 512 $2b \times 2b$ 次乘加运算下的 $\sigma \approx 4$。

为了进行公平比较并提升噪声容限，我们对 ISAAC 做了两处改动。ISAAC 的编码策略依赖一个模拟电路，用于对交叉阵列输入 [54]求和。研究表明，该组件在噪声下会导致精度下降 [73],，因此我们用数字等效电路替换了它。为了评估 BERT 的精度，我们让 ISAAC 用两个周期处理正向/负向输入，与 RAELLA 保持一致。这能进一步增强抗噪能力。图 15展示了以下内容：

• ISAAC：当噪声达到 $> 4\%$ 时，所有深度神经网络都会遭受严重的精度损失。ISAAC 使用无符号权重，其高位密集（图 8）。密集位会产生更大且噪声更高的数值，而高位切片中的噪声会导致结果出现较大误差。• 中心值+偏移：这至关重要。偏移编码可提供抗噪能力 [73],，而中心值+偏移可提高位稀疏性并降低噪声。直观而言，数字中心值处理将大量计算移出噪声较大的模拟域。

•自适应权重切片：精度进一步提升。RAELLA 的经验切片策略能够感知噪声，使 RAELLA 能够根据噪声水平的变化调整切片方式。随着噪声增大，自适应权重切片会减少每个权重切片的位数以降低误差；在测试的最高噪声水平下，大多数层使用五个权重切片。

• RAELLA：通过推测，RAELLA 达到的精度与不采用推测的方法相近。恢复步骤可防止推测失败导致精度下降。

我们发现，RAELLA 能在更高的噪声水平下维持深度神经网络准确率，而在 ISAAC 上，所有深度神经网络都会在较低的噪声水平下遭受严重的精度损失。与较大的深度神经网络相比，紧凑型深度神经网络因错误而遭受的精度下降更严重 [76]。BERT 则能从双周期正/负输入产生的稀疏性中独特受益。再加上 BERT 的庞大规模，使其能够在高噪声水平下维持更高的精度。

RAELLA 能适应不同的噪声水平；自适应权重切片会在高噪声场景下自动减少每个切片的位数，从而在存储密度、效率与正确性之间进行权衡。这使 RAELLA 无需重新训练即可维持精度，同时在噪声约束下尽可能提高效率。

8 相关工作

Xiao[等人]深入且富有洞见地探讨了深度神经网络准确率与保真度之间的关系、差分编码以及存内计算设计空间的决策。我们建议读者阅读这项工作，以深入了解 RAELLA 所探讨的权衡。

多项研究致力于突破低模数转换器分辨率的限制。TinyADC [79] 在剪枝深度神经网络权重位的同时进行重新训练，大幅降低了列和分辨率。BRAHMS [57]为每一层量身定制模数转换器量化步长，以在保真度损失的约束下最大限度地提高深度神经网络准确率。Guo 等人 [15] 利用列和天然较低这一特点来降低模数转换器分辨率，并根据列和预测调整所用交叉阵列行的数量。McDanel 等人 [33] 研究了低分辨率模数转换器量化和深度神经网络误差容限。与这些研究相比，RAELLA 大幅降低了模数转换器分辨率（2b–3b，而非 10b）。

Newton [36]通过调整模数转换器分辨率、采用异构瓦片以及使用可减少计算量的变换，改进了 ISAAC。这些方法与 RAELLA 相互正交；值得探讨加速器如何结合两者。

9 结论

RAELLA 表明，存内计算加速器无需重新训练或修改深度神经网络，即可降低高昂的模数转换器成本。通过针对低分辨率模拟输出进行编码并改变切片模式，RAELLA

可以重塑计算所得模拟值的分布。RAELLA 利用这一能力，使计算所得的模拟值保持低分辨率和高保真度，同时尽可能提升每个深度神经网络层的效率和吞吐量。我们希望，通过扩展存内处理设计者可用的免重新训练策略，RAELLA 能够启发未来的硬件策略，创造全新的协同设计机会，并拓展存内处理的应用范围。

10 致谢

本研究部分经费由爱立信、MIT人工智能硬件计划和 MIT Quest 提供。

参考文献

[1] Fabien Alibart、Ligang Gao、Brian D Hoskins 和 Dmitri B Strukov。2012。通过可适应的变异容忍算法对忆阻器件的状态进行高精度调节。《纳米技术》23, 7（2012年1月），075201。https://doi.org/10.1088/0957- 4484/23/7/075201[2] Pai-Yu Chen、Xiaochen Peng 和 Shimeng Yu。2017。NeuroSim+：用于突触器件和阵列架构基准测试的集成式器件到算法框架。载于*2017 IEEE*国际电子器件会议（*IEDM*）。6.1.1–6 1 4 10 1109 2017 8268337 。。https://doi.org/ . /IEDM. . Yuzong [3] Chen、Lu Lu、Bongjin Kim 和 Tony Tae-Hyoung Kim。2020。适用于多用途数据存储和存内计算的可重构2T2R电阻式随机存取存储器架构。*IEEE*超大规模集成（*VLSI*）系统汇刊28, 12 2020 2636 2649 10 1109 2020 3028848 ( ), –。https://doi.org/ . /TVLSI. .[4] Yunji Chen、Tao Luo、Shaoli Liu、Shijin Zhang、Liqiang He、Jia Wang、Ling Li、Tianshi Chen、Zhiwei Xu、Ninghui Sun 和 Olivier Temam。2014。DaDianNao：机器学习超级计算机。载于*2014*年第*47*届 *IEEE/ACM* 微体系结构国际研讨会。609–622。https://doi.org/10.1109/MICRO.2014. 58[5] Ping Chi、Shuangchen Li、Cong Xu、Tao Zhang、Jishen Zhao、Yongpan Liu、Yu Wang 和 Yuan Xie。2016。PRIME：一种用于基于电阻式随机存取存储器的主存中神经网络计算的新型存内处理架构。载于*2016*年*ACM/IEEE* 第*43*届计算机体系结构年度国际研讨会（*ISCA*）。27–39。https://doi.org/10.1109/ISCA.2016.13[6] Jungwook Choi、Swagath Venkataramani、Vijayalakshmi (Viji) Srinivasan、Kailash Gopalakrishnan、Zhuo Wang 和 Pierce Chuang。2019。精确且高效的2位量化神经网络。载于《机器学习与系统会议论文集》，A. Talwalkar、V. Smith 和 M. Zaharia（编），第1卷。348–359。https://proceedings.mlsys.org/paper/2019/file/006f52e9102a8d3be2fe5614f42ba989-Paper.pdf [7] Teyuh Chou、Wei Tang、Jacob Botimer 和 Zhengya Zhang。2019。CASCADE：在端到端存内处理范式中连接电阻式随机存取存储器，以扩展模拟数据流。载于*IEEE/ACM*第*52*届微体系结构国际研讨会论文集（美国俄亥俄州哥伦布）（*MICRO '52*）。计算机协会，纽约，美国，114–125。https://doi.org/10. 1145/3352460.3358328[8] Chaoqun Chu、Yanzhi Wang、Yilong Zhao、Xiaolong Ma、Shaokai Ye、Yunyan Hong、Xiaoyao Liang、Yinhe Han 和 Li Jiang。2020。PIM-Prune：面向基于交叉阵列的存内处理架构的细粒度深度卷积神经网络剪枝。载于*2020*年第*57*届1 6 10届 *ACM/IEEE* 设计自动化会议（*DAC*）。– 。https://doi.org/ . 1109/DAC18072.2020.9218523[9] Philip Colangelo、Nasibeh Nasiri、Eriko Nurvitadhi、Asit Mishra、Martin Margala 和 Kevin Nealis。2018。使用 Intel ® FPGA 探索低数值精度深度学习推理。载于*2018*年 *IEEE* 第*26*届现场可编程定制计算机年度国际研讨会（*FCCM*）。73–80。https: //doi.org/ 10 1109 2018 00020 . /FCCM. . Jia Deng、Wei Dong、Richard Socher、Li-[10]Jia Li、Kai Li 和 Li Fei-Fei。2009。ImageNet：大型分层图像数据库。载于*2009*年 *IEEE* 计算机视觉与模式识别会议。248–255。https://doi.org/10.1109/CVPR. 2009.5206848[11] Lei Deng、Ling Liang、Guanrui Wang、Liang Chang、Xing Hu、Xin Ma、Liu Liu、Jing Pei、Guoqi Li 和 Yuan Xie。2020。SemiMap：一种在交叉阵列上实现速度与开销平衡的半折叠卷积映射。*IEEE*计算机辅助集成电路与系统设计汇刊39, 1（2020），117–130。https: //doi.org/ 10 1109 2018 2883959 . /TCAD. . Andrea Fasoli、Chia-Yu Chen、[12]Mauricio Serrano、Xiao Sun、Naigang Wang、Swagath Venkataramani、George Saon、Xiaodong Cui、Brian Kingsbury、Wei Zhang、Zoltán Tüske 和 Kailash Gopalakrishnan。2021。基于长短期记忆网络的语音识别模型的4位量化。载于*Interspeech 2021*论文集。2586–2590。https://doi.org/ 10 21437 2021 1962 . /Interspeech. - Ligang Gao、Fabien [13]Alibart 和 Dmitri B. Strukov。2013。一种高分辨率非易失性模拟存储器离子器件。[14] Sujan K. Gonugondla、Charbel Sakr、Hassan Dbouk 和 Naresh R. Shanbhag。2020。存内架构精度的基本极限。载于第*39*届国际计算机辅助设计会议论文集（线上活动，美国）（*ICCAD '20*）。计算机协会，纽约，纽约州，美国，文章 128, 9 10 1145 3400302 3416344 页。https://doi.org/ 。/。[15] Mengyu Guo、Zihan Zhang、Jianfei Jiang、Qin Wang 和 Naifeng Jing。2022。通过行激活超额订阅提升基于ReRAM的DNN。载于*2022*年第*27*届亚洲及南太平洋设计自动化会议（*ASP-DAC*）。604–609。https: //doi.org/ 10 1109 52403 2022 9712520 。/ASP-DAC。。何恺明、X. Zhang、任少卿和孙剑。2016。用于图像识别的深度残差学习。*2016 IEEE*计算机视觉与模式识别会议（*CVPR*）（2016），770–778。[17] Miao Hu、John Paul Strachan、Zhiyong Li、Emmanuelle M. Grafals、Noraica Davila、Catherine Graves、Sity Lam、Ning Ge、Jianhua Joshua Yang 和 R. Stanley Williams。2016。用于神经形态计算的点积引擎：编程1T1M交叉阵列以加速矩阵-向量乘法。载于*2016*年第*53*届*ACM/EDAC/IEEE*设计自动化会议（*DAC*）。1–6。https://doi.org/10.1145/2897937.2898010[18] Norman P. Jouppi、Andrew B. Kahng、Naveen Muralimanohar 和 Vaishnav Srinivas。2015。CACTI-IO：结合片外功耗-面积-时序模型的CACTI。*IEEE*超大规模集成电路系统汇刊23, 7（2015），1254 1267 10 1109 2014 2334635 –。https://doi.org/。/TVLSI。。[19] Tinu Theckel Joy、Santu Rana、Sunil Gupta 和 Svetha Venkatesh。2016。使用贝叶斯优化为大数据调优超参数。载于*2016*年第*23*届国际模式识别会议（*ICPR*）。2574–2579。https: //doi.org/ 10 1109 2016 7900023 。/ICPR。。Sangyeob Kim、Sangjin Kim、Soyeon Um、Soyeon Kim、Kwantae Kim 和 Hoi-Jun Yoo。2022。[20]Neuro-CIM：一种310.4 TOPS/W神经形态存内计算处理器，具有低字线/位线活动和数模混合模式神经元发放。载于*2022 IEEE*超大规模集成电路技术与电路研讨会（*VLSI Technology and Circuits*）。38–39。https://doi.org/10.1109/VLSITechnologyandCir46769.2022.9830276[21] Michael Klachko、Mohammad Reza Mahmoodi 和 Dmitri Strukov。2019。提升混合信号神经网络的噪声容限。载于*2019*年国际神经网络联合会议（*IJCNN*）。1–8。https://doi.org/10.1109/IJCNN.2019.8851966[22] Raghuraman Krishnamoorthi。2018。面向高效推理的深度卷积网络量化：白皮书。*CoRR* abs/1806.08342（2018）。arXiv:1806.08342 http://arxiv.org/abs/1806.08342[23] Lukas Kull、Thomas Toifl、Martin Schmatz、Pier Andrea Francese、Christian Menolfi、Matthias Braendli、Marcel Kossel、Thomas Morf、Toke Meyer Andersen 和 Yusuf Leblebici。2013。一种采用交替比较器、用于提升速度的3.1mW、8b、1.2GS/s单通道异步逐次逼近型模数转换器，采用32纳米数字SOI CMOS工艺。载于*2013 IEEE*国际固态电路会议技术论文集。468 469 10 1109 2013 6487818 –。https://doi.org/。/ISSCC。。[24] Weitao Li、Pengfei Xu、Yang Zhao、Haitong Li、Yuan Xie 和 Yingyan Lin。2020。TIMELY：推动存内计算加速器中的数据移动与接口迈向局部及时域。载于*ACM/IEEE*第*47*届年度计算机体系结构国际研讨会论文集（线上活动）（*ISCA '20*）。IEEE出版社，832 845 10 1109 45697 2020 00073 –。https://doi.org/。/ISCA。。[25] Tailin Liang、John Glossner、Lei Wang、Shaobo Shi 和 Xiaotong Zhang。2021。深度神经网络加速的剪枝与量化：综述。神经计算 461 2021 370 403 10 1016 2021 07 045 （），–。[26] 2019 https://doi.org/。/j.neucom.。。Jilan Lin、Zhenhua Zhu、Yu Wang 和 Yuan Xie。。学习ReRAM稀疏性：面向ReRAM加速器的稀疏神经网络映射与剪枝。载于第*24*届亚洲及南太平洋设计自动化会议论文集（日本东京）（*ASPDAC '19*）。计算机协会，纽约，纽约州，美国，639 644 10 1145 3287624 3287715 –。[27] 2010 https://doi.org/。/。Eike Linn、Roland Rosezin、Carsten Kügeler 和 Rainer Waser。。用于无源纳米交叉阵列存储器的互补电阻开关。《自然·材料》9, 5（2010年5月1日），403–406。https://doi.org/10.1038/nmat2748[28] Qi Liu、Bin Gao、Peng Yao、Dong Wu、Junren Chen、Yachuan Pang、Wenqiang Zhang、Yan Liao、Cheng-Xin Xue、Wei-Hao Chen、Jianshi Tang、Yu Wang、Meng-Fan Chang、He Qian 和 Huaqiang Wu。2020。33.2 一款全集成模拟ReRAM存内计算芯片，实现78.4TOPS/W和全并行乘加计算。载于*2020 IEEE*国际固态电路会议（*ISSCC*）。500 502 10 1109 19947 2020 9062953 –。https://doi.org/。/ISSCC。。[29] Anni Lu、Xiaochen Peng、Wantong Li、Hongwu Jiang 和 Shimeng Yu。2021。使用40nm RRAM存内计算宏单元验证NeuroSim。载于*2021 IEEE*第三届人工智能电路与系统国际会议（*AICAS*）。1 4 10 1109 51828 2021 9458501 –。[30] https://doi.org/。/AICAS。。Ningning Ma、Xiangyu Zhang、Hai-Tao Zheng 和 Jian Sun。2018。ShuffleNet V2：高效卷积神经网络架构设计的实用指南。载于欧洲计算机视觉会议论文集（*ECCV*）。[31] Sébastien Marcel 和 Yann Rodriguez。2010。Torchvision：Torch的机器视觉软件包。载于第*18*届*ACM*国际多媒体会议论文集（意大利佛罗伦萨）（*MM '10*）。计算机协会，纽约，纽约州，美国，1485–1488。https://doi.org/10.1145/1873951.1874254

[32] Matthew J. Marinella、Sapan Agarwal、Alexander Hsia、Isaac Richter、Robin Jacobs-Gedrim、John Niroula、Steven J. Plimpton、Engin Ipek 和 Conrad D. James。2018。对基于ReRAM的模拟神经网络训练加速器的能耗、延迟、面积和精度进行多尺度协同设计分析。*IEEE*新兴与精选电路和系统主题期刊8, 1（2018），86–101。https://doi.org/10. 1109 2018 2796379 / JETCAS.。。Bradley McDanel、Sai Qian Zhang 和 H. T. Kung。2021。利用项量化产生的位级稀疏性实现饱和型阻变式随机存取存储器。载于*2021 IEEE*国际电路与系统研讨会（*ISCAS*）。1–5。https://doi.org/ 10.1109/ISCAS51556.2021.9401293[34] Sparsh Mittal。2019。面向存内处理和神经网络的基于ReRAM架构综述。机器学习与知识提取1, 1（ 2019 75 114 10 3390 1010005 ），–。https://doi.org/。/make Boris Murmann。。模数转换器的能耗极限。载于*2013 IEEE*低电压低功耗会议。1–4。https://doi.org/10.1109/FTFC.2013.6577781[36] Anirban Nag、Rajeev Balasubramonian、Vivek Srikumar、Ross Walker、Ali Shafiee、John Paul Strachan 和 Naveen Muralimanohar。2018。Newton：迈向交叉阵列加速的物理极限。*IEEE Micro* 38, 5（2018）， 41 49 10 1109 2018 053631140 –。https://doi.org/。/MM。。Markus Nagel、Mart van Baalen、Tijmen Blankevoort 和 Max Welling。2019。通过权重均衡和偏置校正实现无数据量化。（2019）。https://doi.org/ 10 48550 1906 04721 。/ARXIV。。M. O' Halloran 和 R. Sarpeshkar。。一种10-nW、12位精度、漏电流为10-aA的模拟存储单元。*IEEE* 固态电路期刊39, 11（2004），1985–1996。https://doi.org/10.1109/JSSC.2004.835817[39] Atsuya Okazaki、Pritish Narayanan、Stefano Ambrogio、Kohji Hosokawa、Hsinyu Tsai、Akiyo Nomura、Takeo Yasuda、Charles Mackin、Alexander Friz、Masatoshi Ishii、Yasuteru Kohda、Katie Spoon、An Chen、Andrea Fasoli、Malte J. Rasch 和 Geoffrey W. Burr。2022。一款基于模拟存储器的14nm硬件加速器，用于稠密深度神经网络（包括Transformer）。载于*2022 IEEE*国际电路与系统研讨会（*ISCAS*）。3319–3323。https://doi.org/10.1109/ISCAS48785.2022.9937292[40] Shunsuke Okumura、Makoto Yabuuchi、Kenichiro Hijioka 和 Koichi Nose。2019。一款基于三值、位宽可扩展的8.80 TOPS/W卷积神经网络加速器，采用众核存内处理架构，具有896K突触/mm2。载于*2019*年超大规模集成电路技术研讨会。C248–C249。https://doi.org/10.23919/VLSIT.2019.8776544[41] Angshuman Parashar、Priyanka Raina、Yakun Sophia Shao、Yu-Hsin Chen、Victor A. Ying、Anurag Mukkara、Rangharajan Venkatesan、Brucek Khailany、Stephen W. Keckler 和 Joel Emer。2019。Timeloop：一种系统化的DNN加速器评估方法。载于*2019 IEEE*系统与软件性能分析国际研讨会（*ISPASS*）。304–315。https://doi.org/10.1109/ISPASS.2019.00042[42] Adam Paszke、Sam Gross、Francisco Massa、Adam Lerer、James Bradbury、Gregory Chanan、Trevor Killeen、Zeming Lin、Natalia Gimelshein、Luca Antiga、Alban Desmaison、Andreas Kopf、Edward Yang、Zachary DeVito、Martin Raison、Alykhan Tejani、Sasank Chilamkurthy、Benoit Steiner、Lu Fang、Junjie Bai 和 Soumith Chintala。2019。PyTorch：一种命令式风格、高性能深度学习库。载于神经信息处理系统进展第*32*卷。Curran Associates, Inc.，8024–8035。
h
t
t
p://
p
a
p
e
r
s.
neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf[43] David Patterson、Joseph Gonzalez、Quoc Le、Chen Liang、Lluis-Miquel Munguia、Daniel Rothchild、David So、Maud Texier 和 Jeff Dean。2021。碳排放与大型神经网络训练。https://doi.org/ 10 48550 2104 10350 。/ARXIV。。Xiaochen Peng、Shanshi Huang、Hongwu Jiang、Anni Lu 和 Shimeng Yu。。DNN+NeuroSim V2.0：用于片上训练存内计算加速器的端到端基准测试框架。*IEEE*集成电路与系统计算机辅助设计汇刊40, 11（2021），2306–2319。https: //doi.org/ 10 1109 2020 3043731 。/TCAD。。Lillian Pentecost、Alexander Hankin、Marco Donato、Mark Hempstead、Gu-Yeon Wei 和 David Brooks。2022。NVMExplorer：用于嵌入式非易失性存储器跨层级比较的框架。载于*2022 IEEE*高性能计算机体系结构国际研讨会（*HPCA*）。938–956。https: //doi.org/ 10 1109 53966 2022 00073 。/HPCA。。Antonio Polino、Razvan Pascanu 和 Dan Alistarh。2018。通过蒸馏与量化实现模型压缩。https://doi.org/ 10 48550 1802 05668 。/ARXIV。。Ximing Qiao、Xiong Cao、Huanrui Yang、Linghao Song 和 Hai Li。。Atom-Layer：一种采用原子层计算的通用型基于ReRAM的CNN加速器。载于*2018*年第*55*届*ACM/ESDA/IEEE*设计自动化会议（*DAC*）。1–6。https://doi.org/ 10 1109 2018 8465832 。/DAC。。Songyun Qu、Bing Li、Ying Wang 和 Lei Zhang。2021。ASBP：面向基于RRAM的神经网络加速器的自动结构化位剪枝。载于*2021*年第*58*届*ACM/IEEE*设计自动化会议（*DAC*）。745–750。https://doi.org/10.1109/DAC18074.2021.9586105[49] Pranav Rajpurkar、Jian Zhang、Konstantin Lopyrev 和 Percy Liang。2016。SQuAD：用于文本机器阅读理解的100,000+ 个问题。载于*2016*年自然语言处理经验方法会议论文集。计算语言学协会，美国得克萨斯州奥斯汀，2383–2392。 https://doi.org/ 10 18653 1 16 1264 . /v /D - Aditya Ramesh、Prafulla Dhariwal、Alex Nichol、Casey Chu 和 Mark Chen。2022。利用 CLIP 潜变量进行分层文本条件图像生成。https: //doi.org/ 10 48550 2204 06125 . / ARXIV. . Babak Rokh、Ali Azarpeyvand 和 Ali Reza Khanteymoori。2022。深度神经网络模型量化的全面综述。*arXiv* abs/2205.07877 (2022)。[52] Mehdi Saberi、Reza Lotfi、Khalil Mafinezhad 和 Wouter A. Serdijn。2011。用于逐次逼近型模数转换器的电容式数模转换器的功耗与线性度分析。*IEEE* 电路与系统汇刊 *I*：常规论文 58, 8 (2011), 1736–1748。https://doi.org/10.1109/ TCSI.2011.2107214[53] Mark Sandler、Andrew Howard、Menglong Zhu、Andrey Zhmoginov 和 Liang-Chieh Chen。2018。MobileNetV2：倒置残差与线性瓶颈。4510–4520。https://doi.org/10.1109/CVPR.2018.00474[54] Ali Shafiee、Anirban Nag、Naveen Muralimanohar、Rajeev Balasubramonian、John Paul Strachan、Miao Hu、R. Stanley Williams 和 Vivek Srikumar。2016。ISAAC：一种利用交叉阵列原位模拟运算的卷积神经网络加速器。载于*2016 ACM/IEEE* 第*43*届计算机体系结构年度国际研讨会（*ISCA*）。 14 26 10 1109 2016 12 – 。https://doi.org/ 。/ISCA. . Mahmut E. Sinangil、Burak Erbagci、Rawan Naous、Kerem Akarvardar、Dar Sun、Win-San Khwa、Hung-Jen Liao、Yih Wang 和 Jonathan Chang。2021。一种支持多位输入、权重和输出、实现 351 TOPS/W 和 372.4 GOPS 的 7 nm 存算一体 SRAM 宏。*IEEE* 固态电路期刊 56, 1 2021 188 198 10 1109 2020 3031290 ( ), – 。https://doi.org/ 。/JSSC. . Linghao Song、Xuehai Qian、Hai Li 和 Yiran Chen。。PipeLayer：一种用于深度学习的基于 ReRAM 的流水线加速器。载于*2017 IEEE* 国际高性能计算机体系结构研讨会（*HPCA*）。541–552。https://doi.org/ 10 1109 2017 55 . /HPCA. . Tao Song、Xiaoming Chen、Xiaoyu Zhang 和 Yinhe Han。2021。BRAHMS：利用混合模拟存储系统，超越传统的基于RRAM 的神经网络加速器。载于*2021*年第*58*届 *ACM/IEEE* 设计自动化会议（*DAC*）。 1033 1038 10 1109 18074 2021 9586247 – 。https://doi.org/ 。/DAC . . Katie Spoon、Hsinyu Tsai、An Chen、Malte J. Rasch、Stefano Ambrogio、Charles Mackin、Andrea Fasoli、Alexander M. Friz、Pritish Narayanan、Milos Stanisavl-jevic 和 Geoffrey W. Burr。2021。借助模拟存储器件实现 Transformer 深度神经网络的软件等效精度。《计算神经科学前沿》15 (2021)。https://doi.org/10.3389/fncom. 2021.675741[59] Vivienne Sze、Yu-Hsin Chen、Tien-Ju Yang 和 Joel S. Emer。2020。深度神经网络的高效处理。Springer International Publishing。https: //doi.org/10.1007/978-3-031-01766-7[60] Christian Szegedy、Wei Liu、Yangqing Jia、Pierre Sermanet、Scott Reed、Dragomir Anguelov、Dumitru Erhan、Vincent Vanhoucke 和 Andrew Rabinovich。2015。通过卷积实现更深层次的网络。载于*2015 IEEE* 计算机视觉与模式识别会议（*CVPR*）。1–9。https://doi.org/10.1109/CVPR.2015.7298594[61] Christian Szegedy、Vincent Vanhoucke、Sergey Ioffe、Jon Shlens 和 Zbigniew Wojna。2016。重新思考用于计算机视觉的 Inception 架构。载于*2016 IEEE* 计算机视觉与模式识别会议（*CVPR*）。2818–2826。https://doi.org/ 10 1109 2016 308 . /CVPR. . Yaniv Taigman、Ming Yang、Marc'Aurelio Ranzato 和 Lior Wolf。2014。DeepFace：缩小人脸验证与人类水平性能之间的差距。载于*2014 IEEE* 计算机视觉与模式识别会议。1701–1708。https://doi. org/ 10 1109 2014 220 . /CVPR. . Fengbin Tu、Weiwei Wu、Shouyi Yin、Leibo Liu 和 Shaojun Wei。2018。RANA：采用刷新优化型嵌入式 DRAM 实现高效神经网络加速。*2018 ACM/IEEE* 第*45*届计算机体系结构年度国际研讨会（*ISCA*）(2018), 340–352。[64] Ashish Vaswani、Noam Shazeer、Niki Parmar、Jakob Uszkoreit、Llion Jones、Aidan N Gomez、Ł ukasz Kaiser 和 Illia Polosukhin。2017。Attention is All you Need。载于神经信息处理系统进展，I. Guyon、U. Von Luxburg、S. Bengio、H. Wallach、R. Fergus、S. Vishwanathan 和 R. Garnett（编），第30卷。Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf[65] Marian Verhelst 和 Boris Murmann。2012。CMOS 模数转换器的面积缩放分析。《电子快报》48 (2012), 314–315。[66] Ching-Hua Wang、Yi-Hung Tsai、Kai-Chun Lin、Meng-Fan Chang、Ya-Chin King、Chrong-Jung Lin、Shyh-Shyuan Sheu、Yu-Sheng Chen、Heng-Yuan Lee、Freder-ick T. Chen 和 Ming-Jinn Tsai。2010。采用兼容 CMOS 逻辑的工艺制造的三维 4F² ReRAM 单元。载于*2010* 国际电子器件会议。29.6.1–29.6.4。https://doi.org/10.1109/IEDM.2010.5703446[67] Linfang Wang、Wang Ye、Chun meng Dou、Xin Si、Xiaoxin Xu、Jing Liu、Dashan Shang、Jianfeng Gao、Feng Zhang、Yongpan Liu、Meng-Fan Chang 和 Qi Liu。2021。基于 2T2R RRAM 电压分压、采用输入相关感测控制的高效稳健非易失性存算一体。*IEEE* 电路与系统汇刊 *II*：快报 68, 5 (2021), 1640–1644。https://doi.org/10.1109/TCSII.2021.3067385

[68] 维基百科。2022。处理器性能铁律 — 维基百科，自由的百科全书。http://en.wikipedia.org/w/index.php?title=Iron%20law% 20of%20processor%20performance&oldid=1112639388。 [在线；访问日期：22-November-2022]。[69] Hao Wu、Patrick Judd、Xiaojie Zhang、Mikhail Isaev 和 Paulius Micikevicius。2020。深度学习推理的整数数量化：原理与实证评估。*arXiv* abs/2004.09602 (2020)。[70] Hao Wu、Patrick Judd、Xiaojie Zhang、Mikhail Isaev 和 Paulius Micikevicius。2020。深度学习推理的整数数量化：原理与实证评估。[71] Yannan Nellie Wu、Joel S. Emer 和 Vivienne Sze。2019。Accelergy：面向加速器设计的架构级能耗估算方法。载于*2019 IEEE/ACM* 国际计算机辅助设计会议（*ICCAD*）。 1 8 10 1109 45719 2019 8942149 – 。https://doi.org/ 。/ICCAD . . [72] Yannan Nellie Wu、Vivienne Sze 和 Joel S. Emer。 2020 . 面向存内处理加速器设计的架构级能耗与面积估算器。载于*2020 IEEE* 国际系统与软件性能分析研讨会（*ISPASS*）。 116 118 10 1109 48437 2020 00024 – 。 [73] https://doi.org/ 。/ISPASS . . T. Patrick Xiao、Ben Feinberg、Christopher H. Bennett、Venkatraman Prabhakar、Prashant Saxena、Vineet Agrawal、Sapan Agarwal 和 Matthew J. Marinella。2021。关于模拟神经网络推理加速器的精度 [特刊]。*IEEE* 电路与系统杂志 22 (2021), 26–48。[74] J. Joshua Yang、Dmitri B. Strukov 和 Duncan R. Stewart。2013。用于计算的忆阻器件。《自然·纳米技术》8, 1 (01 Jan 2013), 13–24。https: // doi.org/ 10 1038 2012 240 . /nnano. . Tzu-Hsien Yang、Hsiang-Yun [75] Cheng、Chia-Lin Yang、I-Ching Tseng、Han-Wen Hu、Hung-Sheng Chang 和 Hsiang-Pang Li。2019。稀疏ReRAM引擎：联合探索压缩神经网络中的激活稀疏性和权重稀疏性。载于*2019 ACM/IEEE* 第*46*届计算机体系结构年度国际研讨会（*ISCA*）。236–249。[76] Tien-Ju Yang 和 Vivienne Sze。2019。面向存内处理加速器上高效深度神经网络的设计考量。22.1.1–22.1.4。https: //doi.org/ 10 1109 19573 2019 8993662 。/IEDM。。Amir [77] Yazdanbakhsh、Ashkan Moradifirouzabadi、Zheng Li 和 Mingu Kang。2022。通过协同式存内剪枝和片上重计算加速稀疏注意力。https://doi.org/ 10 48550 2209 00606 。/ARXIV。。Shihui [78] Yin、Zhewei Jiang、Jae-Sun Seo 和 Mingoo Seok。。用于二值/三值深度神经网络的 XNOR-SRAM：存内计算 SRAM 宏。*IEEE* 固态电路期刊55, 6 (2020), 1733–1743。https://doi.org/10. 1109/JSSC.2019.2963616[79] Geng Yuan、Payman Behnam、Yuxuan Cai、Ali Shafiee、Jingyan Fu、Zhiheng Liao、Zhengang Li、Xiaolong Ma、Jieren Deng、Jinhui Wang、Mahdi Bojnordi、Yanzhi Wang 和 Caiwen Ding。2021。TinyADC：面向混合信号深度神经网络加速器的外设电路感知权重剪枝框架。载于*2021*年欧洲设计、自动化与测试会议暨展览会（*DATE*）。926–931。https: // doi.org/10.23919/DATE51398.2021.9474235[80] Geng Yuan、Payman Behnam、Zhengang Li、Ali Shafiee、Sheng Lin、Xiaolong Ma、Hang Liu、Xuehai Qian、Mahdi Nazm Bojnordi、Yanzhi Wang 和 Caiwen Ding。2021。FORMS：面向混合信号深度神经网络加速器的基于ReRAM的细粒度极化原位计算。载于*2021*年第*48*届*ACM/IEEE*计算机体系结构国际年会（*ISCA*）。265–278。https://doi.org/10.1109/ ISCA52012.2021.00029[81] Jinshan Yue、Yongpan Liu、Fang Su、Shuangchen Li、Zhe Yuan、Zhibo Wang、Wenyu Sun、Xueqing Li 和 Huazhong Yang。2019。AERIS：面积和能效兼优的1T2R ReRAM存内处理神经网络片上系统。载于第*24*届亚洲及南太平洋设计自动化会议（日本东京）（*ASPDAC '19*）。计算机协会，纽约，美国， 146 151 10 1145 3287624 3287635 –。https://doi.org/。/。 [82] Xiandong Zhao、Ying Wang、Xuyi Cai、Chuanming Liu 和 Lei Zhang。2020。面向低精度整数硬件的神经网络线性对称量化。载于*ICLR*。

收稿日期：2022年11月22日；修订日期：2023年2月21日；录用日期：2023年3月9日