DS-CIM：面向边缘人工智能模型的数字随机存算一体：通过样本区域重映射实现精确的或运算累加

Kunming Shao[1],*,†, Liang Zhao[2],3,*, Jiangnan Yu[1],*, Zhipeng Liao[4],Xiaomeng Wang[1], Yi Zou[2,3], Tim Kwang-Ting Cheng[1],Chi-Ying Tsui[1]1香港科技大学， [2]华南理工大学，[3]教育部集成电路设计与技术协同优化工程研究中心， [4]西湖大学 *作者贡献相同。†邮箱：kshaoaa@connect.ust.hk

arXiv:2601.06724v1 [cs.AR] 10 Jan 2026

摘要—随机计算（SC）硬件简单，但吞吐量较低；而高吞吐量的数字存内计算（DCIM）在矩阵向量乘法（MVM）中受制于高昂的加法器逻辑。为解决这一权衡，本文提出一种数字随机存内计算（DS-CIM）架构，兼具高精度与高效率。我们通过修改数据表示，在紧凑的无符号基于或门的电路中实现有符号乘累加（MAC）。仅增加 1× 面积，即可将这一低成本电路复制64次，从而提升吞吐量。我们的核心策略是采用共享伪随机数生成器（PRNG）并进行二维分区，实现单周期互斥激活，从而消除或门冲突。我们还通过随机过程分析和数据重映射解决了1饱和问题，显著提升精度并增强对输入稀疏度的适应能力。高精度版本DS-CIM1在CIFAR-10上运行INT8 ResNet18时，精度达到94.45%，均方根误差（RMSE）仅为0.74%。与此同时，高能效版本DS-CIM2的能效达到3566.1 TOPS/W，面积效率达到363.7TOPS/mm^2,，同时将RMSE维持在较低的3.81%。此外，通过在ImageNet上运行INT8 ResNet50以及测试FP8 LLaMA-7B模型的实验，进一步展示了DS-CIM处理更大模型的能力。

I. 引言

深度神经网络（DNN）和 Transformer 模型是现代 AI 的基础，但其计算需求给受能耗和面积限制的边缘设备 [1]–[7]带来了巨大挑战。高效的矩阵向量乘法（MVM）对其在边缘设备上的部署至关重要。人们已探索通过近似计算提升效率，但 Transformer 对精度的高要求限制了许多方法的适用性。

如图 1 所示，基于 OR 门的随机计算（SC）是这类近似方法之一，可实现低成本、容错计算。然而，由于比特流延迟较长且在 [8] 低稀疏度累加过程中存在显著的 1s 饱和误差，其吞吐量较低，限制了其在现代 AI 应用 [9]–[14]中的使用。数字存算一体（DCIM）是另一种提升边缘端 MVM 性能的方法，它通过直接在 SRAM 阵列中执行并行计算并复用权重来实现这一目标，但其

本研究获得 ACCESS—人工智能芯片与新兴智能系统中心的支持，该中心由 InnoHK 资助，位于中国香港特别行政区。

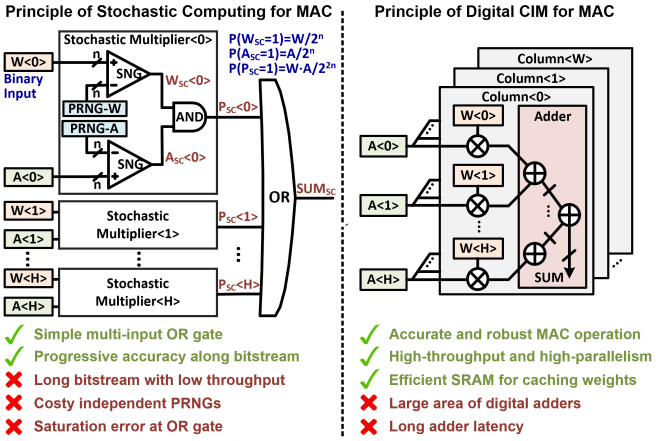

图 1：随机计算与数字存内计算原理。

效率往往受限于用于精确累加的庞大且高功耗的数字加法器树 [15]–[17]。

模拟存内计算（ACIM） [18], [19]和近似数字存内计算 [20]–[22] 通过采用低精度模数转换器或近似加法器来牺牲精度，以提升效率。模拟存内计算采用高效但对工艺、电压和温度敏感的模数转换器进行累加，而近似数字存内计算则采用简化加法器 [21]，由此引入逻辑错误。然而，它们的精度无法满足 Transformer 模型的要求。

通过增加位流长度，SC 可以显著降低误差。将 SC 与存内计算相结合是一种很有前景的替代方案，但现有方法在权重位宽较大时效率低下，并且并行度有限，饱和误差持续存在，尤其是在有符号运算中 [23]–[27]。

为应对这些挑战，我们提出了 DS-CIM 架构。**首先**，我们提出一种新颖的方法，将有符号数转换为无符号表示，从而实现高效的单极性随机计算，并支持有符号激活值和权重。**其次**，我们复制简单的单极性 OR-乘累加电路，将并行度提升 64×，以仅增加 1×面积为代价，弥补长比特流导致的吞吐量下降。**第三**，我们采用共享伪随机数生成器和数据重映射技术，确保每个随机采样点最多激活一行，从而彻底消除 1 饱和误差。**最后**，我们优化伪随机数生成器配置，以

最小化整体均方根误差。

在 40纳米工艺下进行的布局后仿真验证了我们的方法。preciseDS-CIM1 实现了 669.7 TOPS/W 和 117.1 TOPS/mm^2，均方根误差仅为 0.74%。高效的 DS-CIM2 达到 3566.1 TOPS/W 和 363.7 TOPS/mm^2，均方根误差为 3.81%。在所有乘积稀疏度级别（0–100%）下，DS-CIM 均表现出高效率、低误差和稳健的精度。

II. 预备知识

A. 随机计算

随机计算通过将二进制数与随机值进行比较来生成一元比特流，确保每个位置为“1”的概率与二进制值相匹配。这种转换简化了高位宽二进制值的表示，降低了逻辑计算成本并提升了硬件效率[9], [11]–[14]。

经典的随机计算应用（如乘法和缩放）利用一元比特流的概率特性，实现更简单且可扩展的硬件方案。然而，生成较长的一元比特流会增加延迟，因此需要根据应用需求，谨慎权衡效率与计算时间。此外，将随机计算应用于乘累加运算时也面临挑战：累加会导致失真误差 [8], [10]。

B. 数字存内计算

人工智能加速器面临存储墙问题，即存储器访问速度限制了整体性能。数字存内计算（CIM）通过将处理集成到存储单元中，减少数据传输需求并提高效率 [15]–[17]。然而，仍存在计算密度和吞吐量较低、加法器树开销显著等挑战。克服这些问题对于充分发挥数字存内计算在人工智能加速器中的潜力至关重要。

C. 近似存内计算

一些研究将近似计算技术应用于存内计算，例如使用模拟信号进行累加[18]，、设计近似加法器 [21]，以及引入随机计算 [27]。然而，这些方法都面临精度问题。尽管较高的均方根误差在深度神经网络或事件相机中也可能取得良好结果，但随着 Transformer 的发展，以往的近似计算方法已不再适用。

III. DS-CIM 硬件架构

A. 总体架构

如图 2(a) 所示，DS-CIM 将随机计算的伪随机数生成器和随机数生成器（SNG）集成到数字存内计算框架中，以高效的单比特 OR 运算乘加（OR-MAC）电路取代成本高昂的多位加法器。该宏采用 128×32 结构，每列包含 128 个 8 位 SRAM、128 个随机数生成器和 64 个 OR-MAC。输入 SNG 阵列将二进制激活值转换为随机比特流。为支持数据重映射（第 IV 节），该宏仅使用两个相互独立的伪随机数生成器：PRNGA 用于激活值比特流，PRNGW 用于权重比特流。这些伪随机数生成器为随机数生成器提供随机数，以将二进制数据转换为随机比特流 A_{SC} 和 W_{SC}。A_{SC} 在各列间广播，而 W_{SC}由同一列中的 64 个 OR-MAC 共享。

乘法后，乘积比特流（P_{SC}）通过或门进行累加，并在累加器中逐周期累加部分和。为兼顾精度与效率，我们设计了两个版本：精确 DS-CIM1 在每列使用八个基于 OR16 的乘加器（OR-MAC16），而高效的 DS-CIM2 使用两个基于 OR64 的乘加器（OR-MAC64）。两者均执行精确的乘累加运算，但累加粒度和数据重映射粒度不同。

B. 用于有符号乘加的单极性或门乘加器

随机计算采用概率过程执行无符号运算。如图 3(a) 所示，为在随机计算存内计算（S-CIM）中执行有符号乘加， [27] 提出了一种带符号感知的双极性 MAC 电路，适用于有符号权重和无符号激活值。该差分电路将正权重比特流 W_{SC+} 与负权重比特流W_{SC-} 分离，并以两种方式执行乘加运算，最终结果为两种运算结果之差。然而，这种双极性乘加方案显著增加了开销，并限制了并行度和效率。

相比之下，DS-CIM 采用一种适用于有符号激活值和权重的单极性乘加方案：先将有符号数转换为无符号数，再在无符号乘加后将输出转换为有符号数。如公式 1 所示，部分和可由两列以二进制补码表示的 8 位有符号激活值 $x[i]$和权重 $w[i]$ 的乘加运算逻辑得出。要将有符号数转换为无符号数，只需反转其符号位，这等效于加上 128。例如，将 1000 0011（二进制补码，-125）的符号位反转后，可得到 0000 0011（无符号数，3）；将 0111 1111（二进制补码，127）$^-$的符号位反转后，可得到 1111 1111（无符号数，255）。公式 2 和 3 展示了我们将有符号乘法分解为三项的方法，而公式 4 则说明了如何将高开销的有符号乘加拆分为高效的无符号乘加、加法和减法。

$$psum = \sum_{i=0}^{H-1} x[i]w[i] \tag{1}$$

$$x'[i] = x[i] + 2^7 \quad and \quad w'[i] = w[i] + 2^7 \tag{2}$$

$$x[i]w[i] = x'[i]w'[i] - 2^7x[i] - 2^7w'[i] \tag{3}$$

$$psum = \sum_{i=0}^{H-1} x'[i]w'[i] - 2^7 \sum_{i=0}^{H-1} x[i] - 2^7 \sum_{i=0}^{H-1} w'[i] \tag{4}$$

在我们的 DS-CIM 硬件中，项 b 可通过我们的无符号单极性或门乘加器高效计算，项 c在运行时通过单指令多数据流处理器求和，项 d 则离线计算并存储为查找表。项 c 的开销可在多个权重列之间进行空间摊销。与

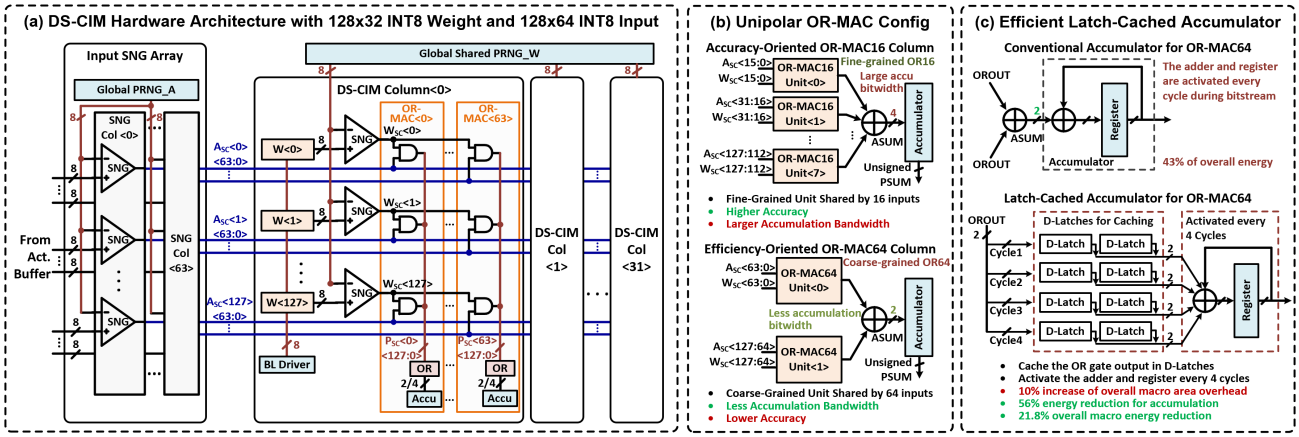

图 2：(a) 所提出的 DS-CIM 总体架构。(b) 所提出的单极性或门乘加器配置示意图：DS-CIM1 采用注重精度的 OR-MAC16 配置，DS-CIM2 采用注重效率的 OR-MAC64 配置。(c) 高效的锁存器缓存累加器，可降低传统累加器的高能耗。

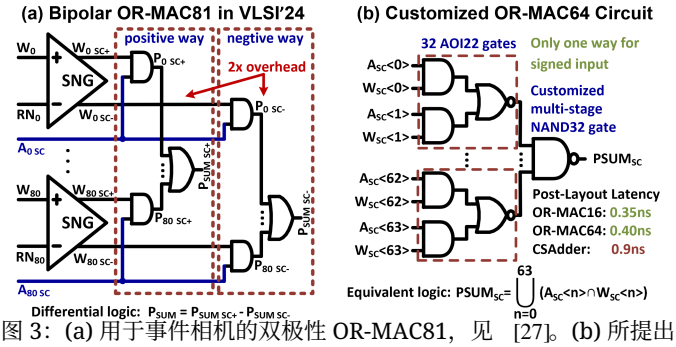

图 3：(a) 用于事件相机的双极性 OR-MAC81，见 [27]。(b) 所提出的单极性 OR-MAC64 电路，在 40纳米工艺下延迟仅为 0.4ns。

大量的乘累加开销相比，减法和摊销加法的开销可以忽略不计。

图 2(b) 展示了两种单极性或运算乘加电路配置。OR-MAC16 注重精度，具有更细的或运算粒度和更大的加法位宽；OR-MAC64 注重效率，其或运算粒度较粗，加法位宽较小。图 3(b) 展示了我们定制的 OR-MAC64 电路原理图，其尺寸更小、速度更快，优于进位保存加法器 [28]。

C. 计算/存储比与面积效率

为实现n位精度，随机计算需要长度为 2^n的比特流。然而，对于8位乘累加运算，256点随机计算的延迟是8位串行数字存内计算的32倍，而后者仅需8个周期。此外，在仅包含一组存储器和一组乘累加电路的S-CIM中，或运算乘加电路的开销显著低于SRAM和随机数生成器的开销，从而导致计算效率与面积效率不均衡。

为解决这一问题，我们在同一 DS-CIM 列中复制 OR-MAC 电路 [27]。通过在一列中集成 64 个超高效 OR-MAC 电路（记为计算/存储比（CMR） = 64），DS-CIM 的并行度提升了 64×，从而提高了吞吐量和面积效率。版图后数据显示，采用

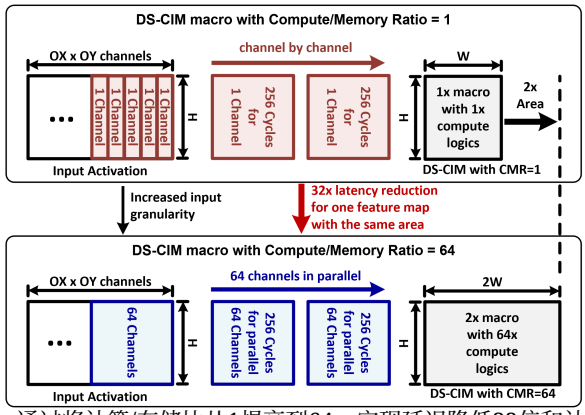

图4：通过将计算/存储比从1提高到64，实现延迟降低32倍和计算密度提升32倍的示意图。

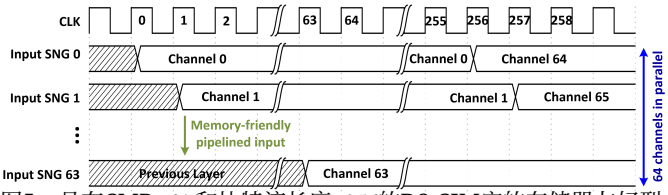

图5：具有CMR=64 和比特流长度=256的DS-CIM宏的存储器友好型流水线层通道输入时序图。

CMR=64 的 DS-CIM 相较于采用 CMR=1的设计，仅增加了 1× 的面积。如图 4 所示，DS-CIM 仅增加 2× 的面积，吞吐量便提升 64× ，使特定卷积层的延迟降低了 32×。此外，DS-CIM 具备渐进精度特性，在比特流长度为=64时，可额外提升 4× 。通过复制低成本的单极性 OR-MAC 电路，我们缓解了随机计算固有的吞吐量限制，从而实现更高的吞吐量。

如图5所示，由于激活值和权重在随机计算过程中保持时间稳定，因此可以高效利用SRAM带宽处理不同的输入通道。

D. 锁存器缓存比特流累加器

如图2(c)所示，DS-CIM 需要逐周期累加一段较长的比特流。在 DS-CIM 中，由于

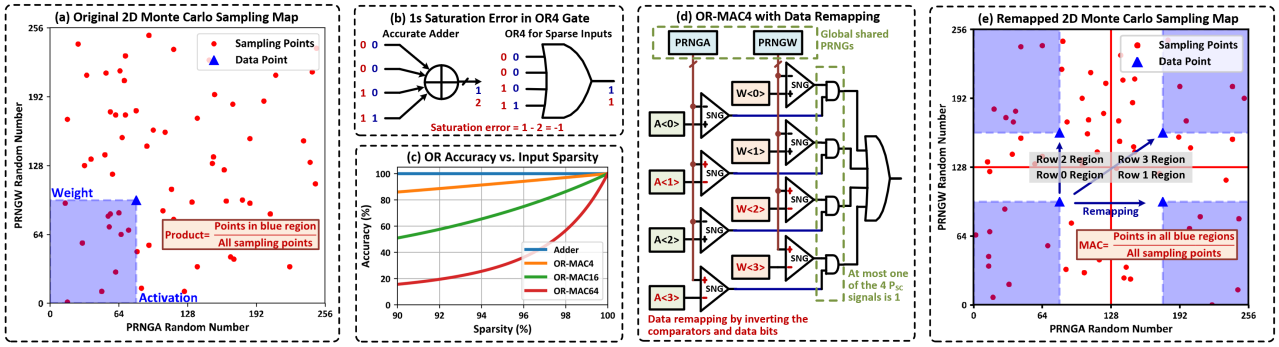

图 6：(a) 将随机乘法表示为二维蒙特卡洛过程。(b) OR4 门中的 1s 饱和误差。(c) 传统 OR-MAC 精度与输入乘积稀疏度之间的关系。(d) OR-MAC4 支持使用反相数据和比较器进行数据重映射。(e) 二维蒙特卡洛采样图中的数据重映射，其中一个采样点最多只会出现在四个蓝色区域之一。

低成本 OR-MAC 单元的大量复制，累加器的能耗占整个宏的 43%。对于累加位宽更小的 DS-CIM2，我们提出了一种锁存器缓存累加器。在 OR-MAC 的 2 位输出端设置八个 D锁存器。输出依次存入锁存器，每四个周期填满一次。到第四个周期时，D锁存器已满，此时加法器将缓存的四个 2 位数相加，并将结果与寄存器中先前存储的值累加。该方法每四个周期才启用一次累加器，使累加能耗降低 56%，宏的整体功耗降低 21.8%，而面积仅增加 10%。

IV. 误差分析与优化

在随机计算存内计算（S-CIM）架构中，由于随机采样和累加的特性，每一行中的较大误差可以相互抵消，从而使乘累加误差相对较小。然而，1s 饱和误差和采样点偏差等固有误差会引入系统性误差，影响 S-CIM。本节将分析 DS-CIM 中的误差来源，并提出相应的解决方案。

A. 随机采样与 *1s* 饱和误差

激活值与权重的乘法可以视为一个二维蒙特卡洛采样过程，通过计算由这两个值形成的面积来估计乘积。如图 6(a) 所示，一个红色采样点（$PRNGA$，$PRNGW$）会在每个周期中随机出现在二维采样图中。当采样点落入由权重和激活值定义的蓝色矩形区域内时，激活值和权重的随机数生成器比较器条件均得到满足，使得 A_{SC}和 W_{SC} 均为 1，从而得到值为 1 的输出乘积。因此，乘积比特流 P_{SC}中 1 的数量反映了激活值与权重乘积的大小，并取决于采样过程中落入矩形区域内的随机点数量。由此，随机乘法过程实际上是一个二维蒙特卡洛过程 [13]。

接下来，我们考虑 S-CIM，其中多个独立的蒙特卡洛采样过程在不同的行中进行。以往的工作 [27] 依赖每行一个独立伪随机数生成器来生成独立的 P_{SC}，然后将它们通过或门累加在一起。然而，或门对 P_{SC}的稀疏度尤其敏感。如图 6(b) 所示，以 OR4 门为例，如果输入稀疏度较高，大多数周期的输入全为 0，或只有一个 1。偶尔也可能有多个输入为 1。然而，或门只能输出一个 1，从而导致**1s 饱和误差**。图 6(c) 中的详细概率分析表明，或累加对乘积稀疏度高度敏感，且粒度更粗的或门对稀疏度变化更加敏感，这限制了基于或门的 S-CIM 的适用性。

B. 针对 *1s* 饱和误差的数据重映射

为解决或累加中的 1s 饱和误差，我们重新考察行之间的随机独立性，并发现，在各行之间共享伪随机数生成器，可以将所有数据点映射到同一个二维采样图中。通过平移和对称，我们确保每个采样点最多只能激活一行并输出 1。

具体而言，如图 6(a) 所示，我们将 [27] 中每行部署独立伪随机数生成器的策略，替换为所有权重共享一个 PRNGW、所有激活值共享一个 PRNGA，从而使该列中的每行都能使用相同的采样点。我们以 OR-MAC4 为例进行说明。在图 6(e) 中，我们将激活值和权重均右移 1 位，使红线围成的数据分布区域仅占据二维采样图的左下四分之一 。

接着，我们将第 1 行的激活值 $A[1]$ 的所有位取反，使其在空间上等效于关于 127.5 对称，例如从 0000 0010 变为 1111 1101。同时，我们改变第二行激活值的随机数序列生成器的判定方向。这会将第二行的红色数据分布区域和蓝色蒙特卡洛有效区域从左下角重新映射到右下角。类似地，通过将 $W[2]$ 及其随机数序列生成器取反，我们可以将第 2 行置于左上角；通过将 $A[3]$、$W[3]$ 及对应的两个随机数序列生成器取反，则可将第 3 行置于右上角 。对于一个共享采样点，**它最多只会落在四个蓝色矩形中的一个**，从而确保 OR4 门的四个输入中最多只有一个为 1。通过右移牺牲数据位宽，我们可以消除 1s 饱和误差。

表I：DS-CIM 在卷积神经网络模型上的性能

	DS-CIM1			DS-CIM2		
位流	64	128	256	64	128	256
RMSE	3.57%	2.03%	0.74%	3.81%	2.63%	0.84%
ResNet18@ CIFAR-10（vs 94.54%）	90%	93.08%	94.45%	89.46%	92.46%	94.31%
ResNet50@ ImageNet（对比 80.82%）	79.45%	80.11%	80.67%	79.34%	79.9%	80.65%

类似地，通过右移 2 位并将相应的数据位取反，我们可以将二维采样图划分为 4×4 个部分，每 16 行共用一个 OR16 门，从而构成 OR-MAC16 设计，称为精确的 **DS-CIM1**。此外，通过右移 3 位并重新映射数据，我们可以将二维采样图划分为 8×8 个部分，每 64 行共用一个 OR64 门，从而构成 OR-MAC64 设计，称为高效的 **DS-CIM2**。

C. 优化后均方根误差的测量

采样点的分布均匀性受伪随机数生成器类型和初始种子的影响，是随机计算精度的关键因素 [13]。为了尽量减小采样点的分布差异，我们收集了主流的 8 位伪随机数生成器，并为 PRNGA 和 PRNGW 的两组随机数序列搜索最优初始值。针对不同的数据分布，我们确定了适用于 64、128 和 256 个采样点的最优伪随机数生成器及初始值配置，以使 OR-MAC16 和 OR-MAC64 的总体均方根误差最小。这些配置可确保各应用在运行时达到最优均方根误差。

表I展示了不同 OR-MAC 电路及比特流长度下的均方根误差表现。比特流长度为=256的高精度 DS-CIM1实现了最低的均方根误差 0.74%，而比特流长度为=64的高效 DS-CIM2实现了可接受的均方根误差 3.81%，优于文献 [21] 报告的 4.03% 和 6.76%。由于右移造成的精度损失更大，OR-MAC64 的均方根误差略高于 OR-MAC16。此外，序列较短时，由于差异增大且采样点不足，均方根误差会显著上升。

测试结果表明，我们的单极性或门乘加电路能够抵御稀疏度变化，对于部分和的任意幅值分布均表现出一致的误差，因此，我们的 DS-CIM 能够适用于具有灵活数据分布的矩阵-向量乘法应用。

V. 实验与验证

为了展示 DS-CIM 在硬件效率、灵活性和精度方面的优势，我们采用 40nm CMOS 工艺实现了 DS-CIM1 和 DS-CIM2 的版图设计。定制的随机数生成器和 OR-MAC单元在 Cadence Virtuoso 中设计，并按照 [28] 中介绍的方法，在 Cadence Liberate 和 Cadence Abstract 中表征为标准单元，以用于综合。获得所有必需的数字标准单元后，使用 Synopsys Design Compiler 对数字门级网表进行综合。随后在 Cadence Innovus 中完成布局布线，并使用 Cadence Virtuoso 进行布局后仿真。

在软件评估方面，我们针对量化深度神经网络和 Transformer 模型开展了全面实验，以评估随机误差对模型精度的影响。我们将 DS-CIM 的误差模式添加到模型的 MVM 结果中。我们分别在 INT8 量化条件下，使用 CIFAR-10 上的 ResNet18 模型和 ImageNet 上规模更大的 ResNet50 模型评估精度。

我们还采用开源的 LLM-FP4 量化框架 [29] 对 LLaMA-7B 模型进行 FP8量化，以评估 DS-CIM 在主流数据集上的效果。此外，根据 [30], 中概述的方法，我们将 FP8激活值和权重以 128 为粒度对齐至 INT8，作为 DS-CIM 的输入。

A. 硬件评估

我们首先评估两种设计的功耗表现及其对有符号运算的敏感性。如图7所示，数据重映射确保每个或门的输入中最多只有一个 1，从而实现超低功耗。与 DS-CIM1 相比，DS-CIM2 的 OR-MAC粒度更粗，加法更简单，累加带宽更低，因此能够很好地兼容锁存器缓存累加器，并且其 MAC电路的能耗低于 DS-CIM1。在有符号运算中，我们将有符号数据转换为无符号数据 [0, 255], ，这会降低随机比特流的稀疏度，并因需要累加更多的 1 而增加功耗。

如图7所示，我们的评估表明，DS-CIM1 对有符号运算较为敏感，而采用 OR-MAC64 和简单加法逻辑的 DS-CIM2 则能够抵御有符号运算的影响。此外，我们通过在所有权重和激活值之间复用伪随机数生成器，降低了其功耗。然而，随机数生成器和累加器仍然是动态功耗的重要来源。在面积方面，我们观察到 DS-CIM1 中加法器占比较高；这也证实，增大 OR-MAC 的粒度能够降低加法器和累加器逻辑的开销，从而提高效率。此外，通过将 OR-MAC 列复制 64 次，我们分摊了权重随机数生成器的开销，使其面积和功耗占比保持在合理范围内。

B. 精度评估

表I显示，均方根误差受OR-MAC粒度和比特流长度的影响，揭示了精度与效率之间的权衡。结论与我们的初步直觉一致。**首先**，增加比特流长度可以提高精度；但由于需要额外的计算周期，这种提升会以效率下降为代价。**其次**，DS-CIM1在精度方面具有优势，而DS-CIM2则展现出更高的效率。**值得注意的是**，在比特流长度为=256时，DS-CIM1的精度达到94.45%，相比精确模型仅下降0.09%；而 DS-CIM2的精度为94.31%，下降了0.23%。我们还展示了 DS-CIM在更大型的ResNet50_Weights.IMAGENET1K_V2模型上的能力，该模型应用于ImageNet,

表 II：DS-CIM 在 FP8量化 LLaMA-7B 模型上的性能

LLaMA-7B 模型	BoolQ	PIQA	HellaSwag	WinoGrande	ARC-e	ARC-c	Avg.	准确率下降
全精度	75.1%	78.7%	56.9%	69.9%	75.3%	41.9%	66.3%	0
FP8量化	75.6%	78.2%	56.6%	70.2%	74.6%	40.7%	66.0%	与全精度相比相差 0.3%
DS-CIM1	71.7%	74.7%	52.1%	67.4%	70.3%	39.4%	62.6%	比 FP8 低 3.4%
DS-CIM2	67.1%	70.7%	47.5%	65.4%	66.3%	35.6%	58.8%	比 FP8 高 7.2%

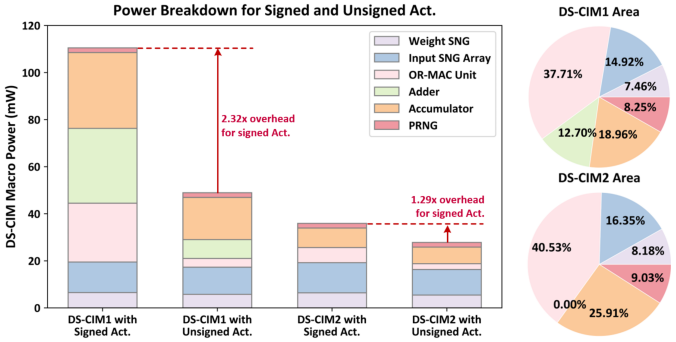

图7：DS-CIM1 和 DS-CIM2 在有符号和无符号激活下的功耗与面积分布，其中 DS-CIM2 配备了锁存器缓存累加器。

表III：与最先进水平近似计算工作的比较

	ISSCC'23 [22]	DIMCA1 [21]	ISSCC'24 [18]	VLSI'24 [27]	DS-CIM1	DS-CIM2
技术	28nm	28nm	65nm	12nm	40nm	
宏类型	数字	数字	模拟	数字随机	数字随机	
精度	A：2~8 位 W：2~8 位	A：1~4 位 W：1 位	A：4~8 位 W：4~8 位	A：2/6b W：6b	A：8b W：8b	
阵列大小	16Kb	16Kb	80Kb	15Kb	32Kb	
面积（mm2）	0.028	0.049	0.48	0.5	0.78	0.72
电压 (V)	0.54~0.9	0.45~1.1	0.6~1.1	0.64	0.7~1.2	
输入	18% 切换	25% 切换	N/A	N/A	87.5% 稀疏度	
TOPS/mm$^{2(1)}$	131.7	192.2	33.0	22.0	117.1$^{(2)}$ ~**468.4**$^{(3)}$	90.9$^{(2)}$ ~**363.7**$^{(3)}$
TOPS/W$^{(1)}$	4569.6	694.4	6652.8	928.8	669.7$^{(2)}$ ~**2677.2**$^{(3)}$	891.5$^{(2)}$ ~**3566.1**$^{(3)}$
重新训练	N/A	Yes	N/A	N/A	不需要	
ResNet18/20 @CIFAR-10$^{(4)}$	91.55%	90.41%	91.70%	N/A	90%$^{(3)}$ ~**94.45**%$^{(2)}$	89.46%$^{(3)}$ ~**94.31**%$^{(2)}$
ResNet50 @ImageNet$^{(4)}$	74.5%	N/A	N/A	N/A	79.45%$^{(3)}$ ~**80.67**%$^{(2)}$	79.34%$^{(3)}$ ~ **80.65**%$^{(2)}$
Transformer 模型	N/A	N/A	ViT-Small @CIFAR-10: 95.8%	N/A	LLaMA-7B @WG: 67.4%$^{(2)}$	LLaMA-7B @WG: 65.4%$^{(2)}$

（1）缩放至 40nm，1 位输入和 1 位权重。（2）在比特流长度为=256 时可获得最佳精度。（3）在比特流长度为=64 时可获得最佳效率。（4）基线精度不完整，模型版本也可能不同。

该模型通过采用新的训练方案改进了ResNet50，精度和鲁棒性均高于此前版本。特别是，在比特流长度为=64时，DS-CIM1的精度仅下降1.37%，而DS-CIM2仅下降1.48%。这不仅表明DS-CIM在大型深度神经网络上表现出色，也凸显了不断演进的算法正日益有利于近似计算。

表 II 展示了 FP8 LLaMA-7B 模型在不同数据集上的精度，以及使用 DS-CIM 进行推理时的精度。我们将 FP8 对齐到 INT8 [30]，，并在 LLaMA-7B 的前向传播中，将 DS-CIM 的误差模式应用于乘加输出，比特流长度为=256 。因此，此处 DS-CIM 的误差来源包括两个方面：一是将 FP8 对齐到 INT8 的过程，二是 DS-CIM 的随机误差。表 II 显示，DS-CIM 在复杂的 Transformer 模型中仍能保持良好的精度，其中 DS-CIM1 的平均精度仅下降 3.4%。这表明 DS-CIM 的应用场景得到了进一步拓展。

分析表明，消除或累加中的 1s 饱和误差，并结合不同行之间的误差相互抵消，可使 DS-CIM 的误差分布均匀且稳定。这使其能够实现媲美数字加法器的高精度，从而适用于各种应用场景。

C. 与最先进工作的比较

表III比较了所提出的DS-CIM1和DS-CIM2与最先进的近似存内计算设计。DS-CIM1在比特流长度为256时，峰值精度达到94.45%，能效为669.7TOPS/W，具有较强的竞争力。与此同时，DS-CIM2展现出更高的能效：在比特流长度为64时，其能效为3566.1 TOPS/W，面积效率为363.7 TOPS/mm^2 ，同时精度保持在89.46%。DS-CIM还支持更大型的模型，例如在ImageNet上运行ResNet50，以及Transformer模型LLaMA-7B。灵活的比特流调度使DS-CIM能够针对不同应用调整精度和效率。

VI. 结论

总之，DS-CIM 架构有效地将随机数字存内计算中的高精度与高效率结合起来，克服了比特流长度相关的挑战以及传统数字存内计算的局限。通过在单极性或乘加列上执行有符号乘加运算，并复制低成本的或乘加电路，我们以仅增加 1× 面积开销为代价，实现了计算密度提升 32× 。我们的随机过程分析和数据重映射技术有效解决了或累加中的1饱和问题，提高了精度，并增强了对稀疏度变化的鲁棒性。DS-CIM1在CIFAR-10数据集上的ResNet18推理精度高达94.45%，均方根误差仅为0.74%；DS-CIM2则实现了卓越的3566.1 TOPS/W能效，同时保持了3.81%的竞争力均方根误差。使用ImageNet上的INT8 ResNet50和FP8 LLaMA-7B开展的实验也全面展示了DS-CIM处理大型模型的能力。本研究展示了DS-CIM在推动高能效、高性能且精度可调的近似计算系统方面的潜力。

参考文献

[1] Kaiming He、Xiangyu Zhang、Shaoqing Ren和Jian Sun。用于图像识别的深度残差学习。载于IEEE计算机视觉与模式识别会议论文集，第770–778页，2016年。[2] Ross Girshick、Jeff Donahue、Trevor Darrell和Jitendra Malik。用于准确目标检测和语义分割的丰富特征层级。载于IEEE计算机视觉与模式识别会议论文集，第580–587页，2014年。[3] Alex Krizhevsky、Ilya Sutskever和Geoffrey E Hinton。使用深度卷积神经网络进行ImageNet分类。神经信息处理系统进展，第25卷，2012年。[4] Shaoqing Ren、Kaiming He、Ross Girshick和Jian Sun。Faster R-CNN：借助区域提议网络实现实时目标检测。IEEE模式分析与机器智能汇刊，39(6)：1137–1149，2016年。[5] Ashish Vaswani、Noam Shazeer、Niki Parmar、Jakob Uszkoreit、Llion Jones、Aidan N Gomez、Łukasz Kaiser和Illia Polosukhin。Attention Is All You Need。神经信息处理系统进展，第30卷，2017年。[6] Hugo Touvron、Thibaut Lavril、Gautier Izacard、Xavier Martinet、Marie-Anne Lachaux、Timoth´ee Lacroix、Baptiste Rozi`ere、Naman Goyal、Eric Hambro、Faisal Azhar等。Llama：开放且高效的基础语言模型。arXiv预印本，arXiv:2302.13971，2023年。[7] Daya Guo、Dejian Yang、Haowei Zhang、Junxiao Song、Ruoyu Zhang、Runxin Xu、Qihao Zhu、Shirong Ma、Peiyi Wang、Xiao Bi等。DeepSeek-R1：通过强化学习激发大语言模型的推理能力。arXiv预印本，arXiv:2501.12948，2025年。[8] Peter Schober、M Hassan Najafi和Nima TaheriNejad。用于一元随机计算的高精度乘累加（MAC）技术。IEEE计算机汇刊，71(6)：1425–1439，2021年。[9] Ethan G Rogers、Sohan Salahuddin Mugdho、Kshemal Kshemendra Gupte和Cheng Wang。Stox-Net：通过部分和的随机处理实现高效存内计算深度神经网络加速器。arXiv预印本，arXiv:2407.12378，2024年。[10] Aokun Hu、Wenjie Li、Dongxu Lyu和Guanghui He。使用伪Sobol比特流的高效并行随机计算乘累加（MAC）技术。IEEE纳米技术汇刊，2024年。[11] Christiam F Frasser、Pablo Linares-Serrano、Iv´an D´ıez de los R´ıos、Alejandro Mor´an、Erik S Skibinsky-Gitlin、Joan Font-Rossell´o、Vincent Canals、Miquel Roca、Teresa Serrano-Gotarredona和Josep L Rossell´o。面向边缘计算应用的卷积神经网络全并行随机计算硬件实现。IEEE神经网络与学习系统汇刊，34(12)：10408–10418，2022年。[12] Amogh Agrawal、Indranil Chakraborty、Deboleena Roy、Utkarsh Saxena、Saima Sharmin、Minsuk Koo、Yong Shim、Gopalakrishnan Srinivasan、Chamika Liyanagedera、Abhronil Sengupta等。纳米级非易失性技术时代下重新审视随机计算。IEEE超大规模集成电路系统汇刊，28(12)：2481–2494，2020年。[13] Armin Alaghi和John P Hayes。使用随机电路进行快速而准确的计算。载于2014年欧洲设计、自动化与测试会议暨展览会（DATE），第1–4页。IEEE，2014年。[14] Wenlun Zhang、Shimpei Ando、Yung-Chin Chen、Satomi Miyagi、Shinya Takamaeda-Yamazaki和Kentaro Yoshioka。PaCIM：一种通过概率近似实现的以稀疏性为中心的混合存内计算架构。arXiv预印本，arXiv:2408.16246，2024年。[15] Yu-Der Chih、Po-Hao Lee、Hidehiro Fujiwara、Yi-Chun Shih、Chia-Fu Lee、Rawan Naous、Yu-Lin Chen、Chieh-Pu Lo、Cheng-Han Lu、Haruki Mori等。16.4：面向机器学习边缘应用的22纳米全数字SRAM全精度存内计算宏，能效为89 TOPS/W，计算密度为16.3 TOPS/mm²。载于2021年IEEE国际固态电路会议（ISSCC），第64卷，第252–254页。IEEE，2021年。[16] Hidehiro Fujiwara、Haruki Mori、Wei-Chang Zhao、Mei-Chen Chuang、Rawan Naous、Chao-Kai Chuang、Takeshi Hashizume、Dar Sun、Chia-Fu Lee、Kerem Akarvardar等。支持宽范围动态电压频率缩放以及乘累加和写操作同时进行的5纳米全数字存内计算宏，能效为254 TOPS/W，计算密度为221 TOPS/mm²。载于2022年IEEE国际固态电路会议（ISSCC），第65卷，第1–3页。IEEE，2022年。[17] Hidehiro Fujiwara、Haruki Mori、Wei-Chang Zhao、Kinshuk Khare、Cheng-En Lee、Xiaochen Peng、Vineet Joshi、Chao-Kai Chuang、Shu-Huan Hsu, Takeshi Hashizume等。34.4 一款3纳米全数字存内计算宏，支持int12× int12，采用并行乘加架构和晶圆厂6T SRAM位单元，能效为32.5 TOPS/W，算力密度为55.0 TOPS/mm 2，存储密度为3.78 MB/mm 2。载于*2024*年 *IEEE*国际固态电路会议（*ISSCC*），第67卷，第572–574页。IEEE，2024年。[18] Kentaro Yoshioka。34.5 一款电容重构存内计算宏，能效为818–4094 TOPS/W，可实现卷积神经网络和Transformer的统一加速。载于*2024*年 *IEEE*国际固态电路会议（*ISSCC*），第67卷，第574–576页。IEEE，2024年。[19] Kodai Ueyoshi、Ioannis A Papistas、Pouya Houshmand、Giuseppe M Sarda、Vikram Jain、Man Shi、Qilin Zheng、Sebastian Giraldo、Peter Vrancx、Jonas Doevenspeck等。Diana：一种端到端高能效数字与模拟混合神经网络片上系统。载于*2022*年 *IEEE*国际固态电路会议（*ISSCC*），第65卷，第1–3页。IEEE，2022年。[20] Mengjie Li、Hongyi Zhang、Siqi He、Haozhe Zhu、Hao Zhang、Jinglei Liu、Jiayuan Chen、Zhenping Hu、Xiaoyang Zeng和Chixiao Chen。一种19.7 TFLOPS/W无乘法对数浮点存内计算架构，采用误差降低补偿近似加法器。载于*2024*年 *IEEE*国际电路与系统研讨会（*ISCAS*），第1–5页。IEEE，2024年。[21] Chuan-Tung Lin、Dewei Wang、Bo Zhang、Gregory K Chen、Phil C Knag、Ram Kumar Krishnamurthy和Mingoo Seok。DimCA：一种28纳米面积高效数字存内计算宏，采用近似算术硬件。*IEEE*固态电路期刊，2023年。[22] Yifan He、Haikang Diao、Chen Tang、Wenbin Jia、Xiyuan Tang、Yuan Wang、Jinshan Yue、Xueqing Li、Huazhong Yang、Hongyang Jia等。7.3 一款28纳米存内计算宏，能效为38–102 TOPS/W，采用8位无乘法近似数字SRAM存内计算宏，用于神经网络推理。载于*2023*年 *IEEE*国际固态电路会议（*ISSCC*），第130–132页。IEEE，2023年。[23] Mohsen Riahi Alam、M Hassan Najafi、Nima TaheriNejad、Mohsen Imani和Lu Peng。面向可靠忆阻器存内计算的随机计算。载于*2023*年大湖区超大规模集成电路研讨会论文集，第397–401页，2023年。[24] Mohsen Riahi Alam、M Hassan Najafi、Nima Taherinejad、Mohsen Imani和Raju Gottumukkala。后冯·诺依曼时代的随机计算：在忆阻器存储器中处理位流。*IEEE*电路与系统汇刊*II*：简报，69(5):2423–2427，2022年。[25] Shady Agwa和Themis Prodromakis。用于向量矩阵乘法的数字存内随机计算架构。纳米技术前沿，5:1147396，2023年。[26] Jiyue Yang、Tianmu Li、Wojciech Romaszkan、Puneet Gupta和Sudhakar Pamarti。一款65纳米8位全数字随机存内计算深度学习处理器。载于*2022*年 *IEEE*亚洲固态电路会议（*A-SSCC*），第10–11页。IEEE，2022年。[27] Jiyue Yang、Alexander Graening、Wojciech Romaszkan、Vinod K Jacob、Puneet Gupta和Sudhakar Pamarti。一款用于事件相机的无模数转换器随机存内计算卷积加速器，处理速度为278–514 M事件/秒。载于*2024*年 *IEEE*超大规模集成电路技术与电路研讨会（超大规模集成电路技术与电路），第1–2页。IEEE，2024年。[28] Kunming Shao、Fengshi Tian、Xiaomeng Wang、Jiakun Zheng、Jia Chen、Jingyu He、Hui Wu、Jinbo Chen、Xihao Guan、Yi Deng等。Synd-CIM：一种性能感知数字存内计算编译器，采用面向多规范的子电路综合。载于*2025*年欧洲设计、自动化与测试会议（*DATE*），第1–7页。IEEE，2025年。[29] Shih-yang Liu、Zechun Liu、Xijie Huang、Pingcheng Dong和Kwang-Ting Cheng。LLM-FP4：4位浮点量化Transformer。*arXiv*预印本 *arXiv:2310.16836*，2023年。[30] Fengbin Tu、Yiqi Wang、Zihan Wu、Ling Liang、Yufei Ding、Bongjin Kim、Leibo Liu、Shaojun Wei、Yuan Xie和Shouyi Yin。RedCIM：一种具有统一浮点/整数流水线、面向云端人工智能加速的可重构数字存内计算处理器。*IEEE*固态电路期刊，58(1):243–255，2022年。