DOI:10.20193/j.ices2097-4191.2025.0041

引用格式:杜禧瑞,尹国栋,陈一鸣,等. ROM-SRAM 混合存内计算架构综述[J]. 集成电路与嵌入式系统,2025,25(8):10-22. DU X R,YIN G D,CHEN Y M,et al. A review on ROM-SRAM hybrid compute-in-memory architecture[J]. Integrated Circuits and Embedded Systems,2025,25(8):10-22 (in Chinese).

ROM-SRAM 混合存内计算架构综述

杜禧瑞,尹国栋,陈一鸣,曾令安,于天熠,杨华中,李学清*

(清华大学 电子工程系 柔性电子技术实验室/北京信息科学与技术国家研究中心/天基网络与通信全国重点实验室,北京 100084)

摘要:神经网络是人工智能的代表性算法,然而其庞大的参数量对其在边缘端的硬件部署提出了新的挑战。在边缘端,一方面,为了应用的灵活性,要求计算硬件能够通过模型参数的微调来实现网络在任务间的迁移;另一方面,为了计算能效和性能,需要实现大容量的片上存储以减少片外访存开销。近期提出的 ROM-SRAM 混合存内计算架构是在成熟 CMOS 工艺下很有潜力的一种方案。得益于高密度 ROM 存内计算,神经网络的大部分权重可以部署在片内而不依赖片外访存;与此同时,SRAM 存内计算可以为基于高密度 ROM 的边缘端存内计算提供灵活性。为了扩展 ROM-SRAM 混合存内计算架构设计和应用的空间,需要进一步提高 ROM 存内计算的密度以支持更大的网络,并探索通过少量 SRAM 存内计算获得更大灵活性的方案。文中介绍了几种常见的提升 ROM 存内计算密度的方法,以及基于 ROM-SRAM 混合存内计算架构的神经网络微调以提升灵活性的方法,并讨论了超大规模神经网络的部署方案和长序列大语言模型中遇到的动态矩阵乘瓶颈的解决方案,展望了 ROM-SRAM 混合存内计算架构广阔的设计空间和应用前景。

关键词:人工智能;神经网络加速器;存内计算;只读存储器;集成电路

中图分类号:TN492 **文献标识码**:A **文章编号**:2097-4191(2025)08-0010-13

A review on ROM-SRAM hybrid compute-in-memory architecture

DU Xirui,YIN Guodong,CHEN Yiming,CHEONG Ling-An,YU Tianyi,YANG Huazhong,LI Xueqing*

(Department of Electronic Engineering,LFET/BNRist/SKLSNC,Tsinghua University,Beijing 100084,China)

Abstract: Neural networks are representative algorithms of artificial intelligence, but their huge number of parameters poses new challenges to their hardware deployment at the edge. On the one hand, for the flexibility of applications, computing hardware is required to be able to transfer the deployed model between tasks through parameter fine-tuning at the edge. On the other hand, in order to improve computing energy efficiency and performance, it is necessary to implement large-capacity on-chip storage to reduce off-chip memory access costs. The recently proposed ROM-SRAM hybrid compute-in-memory architecture is a promising solution under mature CMOS technology. Thanks to the high-density ROM-based compute-in-memory, most of the weights of the neural network can be stored on the chip, cutting the reliance on off-chip memory access. Meanwhile, SRAM-based compute-in-memory can provide flexibility for edge compute-in-memory based on high-density ROM. To expand the design and application space of ROM-SRAM hybrid compute-in-memory architecture, it is necessary to further improve the density of ROM-based compute-in-memory to support larger networks and explore

收稿日期:2025-06-03 修回日期:2025-07-02 录用日期:2025-07-04 网络出版日期:2025-07-17

基金项目:国家自然科学基金(#U21B2030,#U24B6015,#92264204)

杜禧瑞(博士研究生),主要研究方向为存内计算电路和架构设计;尹国栋(博士研究生),主要研究方向为存储器和存内计算电路设计;陈一鸣(博士研究生),主要研究方向为存内计算架构和人工智能协同优化;曾令安(硕士研究生),主要研究方向为存内计算电路设计和新型存储器;于天熠(硕士研究生),主要研究方向为数字存内计算架构和电路设计;杨华中(教授,博士生导师),主要研究方向为无线传感网络、数据转换器、能量收集电路、非易失处理器、类脑计算等;李学清(副教授,博士生导师),主要研究方向为混合信号电路设计、新兴存储器、面向存储的人工智能计算加速等

*通信作者. E-mail:xueqingli@tsinghua.edu.cn

solutions to obtain greater flexibility through a small amount of SRAM compute-in-memory. This paper introduces several common techniques to improve the memory density of ROM-based compute-in-memory, as well as the neural network fine-tuning methods based on the ROM-SRAM hybrid compute-in-memory architecture to improve flexibility. The solutions to the deployment of ultra-large-scale neural networks and the bottleneck of dynamic matrix multiplication in large language models with long sequences are discussed, and the outlook for the broad design space and application prospects of ROM-SRAM hybrid compute-in-memory architecture is provided.

Keywords: artificial intelligence; neural network accelerator; computing-in-memory; read-only memory; integrated circuit

1 应用需求分析

1.1 边缘端人工智能的应用需求

近年来，人工智能(Artificial Intelligence，AI)在各个领域被广泛应用。而作为人工智能代表性算法的神经网络，也在图像[1]、视频[2]、文本[3]等多个领域乃至跨领域的任务中表现出色。神经网络之所以能在多种多样的任务中展现出强大的表达能力，是因为其大量可学习参数能从大量样本中提取更加复杂的特征，从而实现高效表达。自从21世纪因大数据和计算硬件的发展而使得深度神经网络算法被重新关注以来，神经网络的可训练参数从最初的几百万量级[4]迅速增长到了如今大语言模型(Large Language Model，LLM)的百亿量级[5-6]。逐年呈指数增长的可训练参数充分保证了神经网络越来越强大的表达能力。

然而，随着模型规模的不断扩张，巨大的参数量导致的算力稀缺和存储容量不足，成为限制人工智能应用进一步普及的关键瓶颈，这一问题在边缘端尤其突出。边缘端是人们日常生活中最常见的能接触到的人工智能硬件载体。从手机、手表到家电、汽车，边缘端对人工智能的需求也在不断扩张。但是在边缘端，计算和存储资源高度受限，且往往会对功耗有一定限制[7]。当前，在边缘端的人工智能应用通常采用在嵌入式CPU上部署极小的低比特位宽量化模型[8]的方式，存在能力弱、功耗高的固有缺点。而更复杂的应用往往需要通过通信方式发送到云端进行处理。一方面，采用通信方式上传本地数据在云端进行推理的方式存在较大的通信延迟，同时对边缘端的电磁环境也有要求，在恶劣环境下无法使用；另一方面，云端推理的方式也存在隐私泄露等一系列问题[9]。因此，在边缘端进行低功耗的模型推理具有必要性和应用价值。

进一步地，边缘端人工智能应用相对稳定，尤其是在通信受限的情况下，模型参数的更新存在较大的物理限制。但是，神经网络的调整仍是适应边缘端具体任务和保护用户隐私的必要措施[10]。而由于边缘端的能耗和算力资源限制，无法在边缘端进行完整的模型训练，因此，对边缘端的模型进行局部调整就显得尤为必要。

1.2 边缘端人工智能加速硬件的存储容量需求

现代神经网络模型普遍具有大参数量的特性，由于当前片上嵌入式缓存容量的不足，权重矩阵和偏置向量构成的参数空间在推理过程中需要被反复遍历访问，这使得参数的片外访问成为影响计算效能的关键因素。如图1所示，经典深度模型的参数量随时间快速增长，其存储需求已远超主流边缘计算芯片的片上存储容量，两者之间极大的差距导致数据在总线上频繁搬移，进而造成了极大的计算功耗和延时开销，形成了“存储墙”问题[11]。虽然通过各种模型压缩技术优化的轻量化模型(如YOLOv8系列[12])显著降低了参数规模(YOLOv8n：3.2M，YOLOv8x：68.2M)，但参数存储优化仍是边缘部署的核心问题。

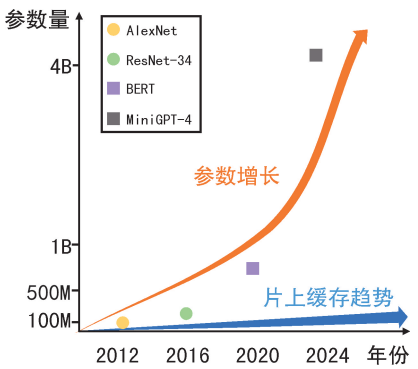


图1 模型参数量和片上缓存容量增长趋势对比

Fig. 1 Comparison between the growing trend of the model parameters and the cache capacity

在数据缓存层面，有些边缘设备需要实时处理高维传感器数据流的需求带来了双重存储压力。以典型视觉任务为例，单帧1 920×1 080 RGB图像(3×8位/像素)需要约6 MB存储空间，视频流处理场景下(30 f/s)，原始数据缓存需求将激增至180 MB/s。这与当前边缘处理器L2缓存容量形成显著差距，迫使系统频繁访问片外存储器，导致严重的性能瓶颈。

当前，业界通过高带宽内存(High Bandwidth Memory，HBM)技术[13-14]带来的超高访存带宽来缓解这一瓶颈，显著提升了性能。然而，由于高昂的成本和有限的可扩展性问题，这一方法难以大规模普及到边缘端加速。

针对这一问题，在近期的定制加速器芯片设计中积极探索了新兴的存内计算(Compute-in-Memory，CiM)架构[15-16]。存内计算架构将存储单元和计算单元融合，避免

了计算中产生的中间数据在总线上的搬移，从而缓解了“存储墙”问题。因此，存内计算架构能够极大地降低人工智能计算的功耗，保障算力，故其也能很好地应用于边缘端人工智能硬件加速。

存内计算得到了多种存储器件技术的支持，包括传统基于 CMOS 工艺的嵌入式存储器，如 SRAM[17-19] 和 eDRAM[20-21]，以及新型非易失存储器件[22-26] 等。相比其他存储器件，SRAM 凭借其低访问延迟、优异的可靠性和成熟的制造工艺[27-29]，在存内计算领域被广泛探索，显著改善了计算能量效率。然而，由于密度受限，在 SRAM 存内计算处理中、大型神经网络时，仍不可避免从片外反复搬移网络权重，从而使得片外权重的加载成为中、大型神经网络推理加速系统的能效瓶颈。

因此，为进一步提升存内计算系统能效，有以下两方面思路：一方面，可以通过增加片上存储密度尽可能将网络完全存储在片上，从而避免系统在使用网络不同部分进行推理时重新加载权重；另一方面，可以将权重非易失地存储在片上，从而进一步降低系统启动时权重加载的开销。

基于新型非易失器件的存内计算是对上述两种思路的一条探索途径。然而，新型非易失器件目前还存在器件偏差、可靠性等方面的问题[22, 30]。而在成熟的 CMOS 工艺下，只读存储器（Read - Only Memory，ROM）便成为另一条新的可行的探索路径。

然而，ROM 权重不可改的特性导致完全基于 ROM 的存内计算无法调整权重，故无法达到第 1.1 节中所述的对灵活性的需求。而完全基于 SRAM 的存内计算虽然达不到密度需求，也没有非易失性，但可以完全重新加载网络的权重，灵活性超出第 1.1 节中提出的需求。因此，将 ROM 存内计算和 SRAM 存内计算结合起来，牺牲一定灵活性换取更高的密度，从而实现高能效边缘端人工智能加速，成为一种新的思路。

ROM - SRAM 混合存内计算架构可以支持多种不同的应用。对于网络规模相对较小且对灵活性需求很低的简单应用，如边缘端的基于特征提取-比对的人脸识别等简单场景，可以将本就相对较少的权重放在 ROM 中，同时保留极少量的 SRAM 用于参数调整，ROM - SRAM 混合存内计算架构可以用很简单的实现方式满足应用需求。而对于一些更复杂的应用，ROM - SRAM 混合存内计算架构仍然具有应用潜力。对于一些网络规模较大但灵活性需求相对较低的应用，如手机等边缘端应用中的 LLM 部署，基于低秩参数调整的 LoRA 技术[31] 可以用极少量的可变 SRAM 权重保证灵活性，而相应的，ROM - SRAM 混合存内计算架构则具有通过提升片上 ROM 存储密度的方式将系统代价进一步降低的潜力；而对于需要更高通用性和灵活性的应用场景，如摄像头根据用户的配置将采集到的数据应用于不同类型的图像分类、目标检测等跨任务域的应用，其网络本身权重量相对较低，而 ROM - SRAM 混合存内计算架构则可以用潜在更少的 SRAM 权重量获得所需的灵活性，从而降低系统整体代价。为了将 ROM - SRAM 混合存内计算架构应用于更多应用，完整展现其潜力，则需要相应地拓展其设计空间。

图 2 展示了边缘端 AI 加速场景下 ROM - SRAM 混合存内计算架构的设计空间。为了同时实现更高的灵活性和片上容量，需要利用神经网络计算的特性进行优化：一方面，可以继续增大片上容量，在片上完整存储更大的网络；另一方面，可以通过少量的 SRAM 尽可能地提升系统的灵活性。在后文中，第 3 章和第 4 章分别就这两方面的优化进行讨论。通过这两方面的优化，并结合 ROM 和 SRAM 比例的调整，可以使得系统支持更大的模型，或者支持更灵活的调整。综上所述，ROM - SRAM 混合存内计算架构是一个具有前景的边缘端人工智能硬件加速的探索方向。

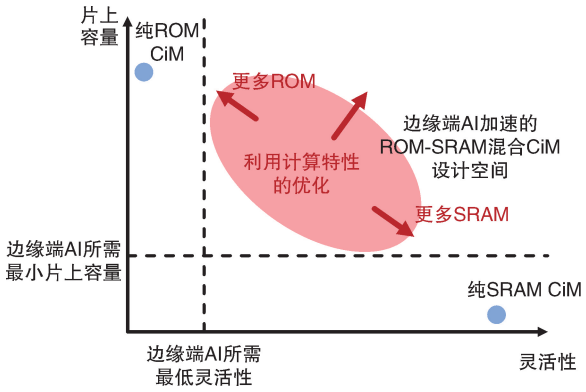


图 2　ROM - SRAM 混合存内计算架构设计空间

Fig. 2　Design space of ROM - SRAM hybrid CiM architecture

2　背　景

2.1　神经网络与参数迁移

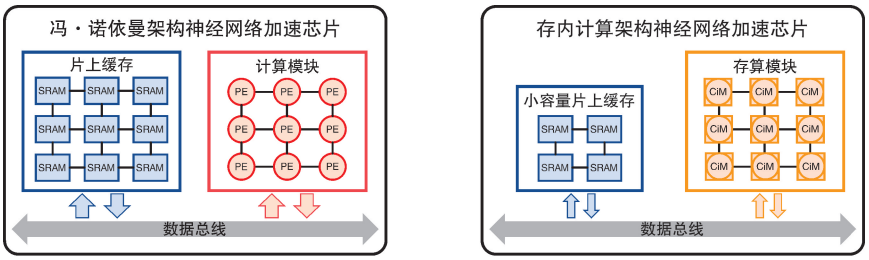
随着深度学习算法[32] 在 21 世纪被重新重视以来，卷积神经网络（Convolutional Neural Network，CNN）[4] 和基于注意力机制的 Transformer 大语言模型[5-6] 在各种应用中的表现均超越传统机器学习算法，甚至超越了人类。这类算法基于大数据，通过梯度下降方法对具有大量参数的原始模型进行训练。其中，卷积神经网络主要依靠其平移不变性和共享权重的特点，在图像识别和语音识别等领域取得了一定突破。而进一步地，针对在文本生成和语义理解等更困难的任务中，Transformer 首次提出了注意力机制[33]，解决了长距离信息提取问题，降低了在大规模数据训练中出现梯度消失和梯度爆炸的可能，为千亿甚至更大

规模的模型出现奠定了基础。

现在的神经网络规模不断增大，因此在边缘端应用中，完整更新整个神经网络通常缺乏足够的训练数据和网络带宽。针对这一问题，微调预训练模型是一个具有潜力的解决方案，而这一方案主要基于神经网络的迁移特性[34]。神经网络通过逐层方式进行特征提取，以卷积神经网络为例，浅层的卷积层提取边缘等浅层特征，而深层的卷积层则进一步提取纹理、形状和语义特征等，最后的全连接层则对种类进行细分。因此，基于这些共通的特征提取，卷积神经网络可以很好地迁移到相似的任务中。而基于 Transformer 的大模型更是基于提示词工程，获得了零样本学习的能力，从而具有很好的任务迁移特性。

2.2 存内计算

基于冯·诺依曼架构的存算分离特性导致的“存储墙”瓶颈[11]，存内计算技术通过融合存储与计算功能（如图 3 所示），为计算机体系结构创新提供了突破方向[27]。其核心思想在于利用存储介质的原位数据处理能力，消除冗余数据搬移带来的能耗损失，借助存储阵列的多行并发访问特性实现带宽跃升，通过二维计算单元的拓扑复用机制实现计算密度倍增。而现代神经网络算法中占主导地位的矩阵-向量运算范式，与存内计算阵列的二维网格结构存在天然的拓扑适配性。通过将权重矩阵静态映射至存算单元，配合输入向量的空间并行加载机制，可实现存储介质同时承载参数固化与乘累加运算的双重功能。这种架构的算法-硬件协同设计理念为计算范式创新开辟了全新维度。



(a) 冯·诺依曼架构　　(b) 存内计算架构

图 3　神经网络加速器芯片

Fig. 3　Neural network accelerators

在 CMOS 工艺下，由于 SRAM 工艺具有成熟性、良好的可微缩性以及高速读/写的特性，基于 SRAM 的存内计算得到了广泛关注。一种思路是通过模拟量进行并行计算。相对更早的 SRAM 存内计算设计[35-36]大多采用电流域的计算方式兼容 SRAM 单元通过位线放电读出的方式。但由于其受晶体管 PVT 变化和非线性影响较大，参考文献[37]和[38]提出了基于匹配良好的金属-氧化物-金属电容上的电荷重分配的电荷域计算方式。而为了降低模拟计算到数字输出之间的转换代价，参考文献[39]采用了时间域计算方式，利用延时进行计算并用时间-数字转换器读出。另一种思路则是采用面积、功耗等代价相对更大的全数字数据通路进行数字域计算[40]，在提升吞吐率的同时获取更优的精度。此外，近年来也有研究者尝试将模拟域计算和数字域计算相结合，综合各自优势得到混合域计算[41-42]。

近年来，除了计算方式的探索，为提升不同指标，SRAM 存内计算也探索了多种技术。参考文献[43]和[44]通过修改存储单元增加读/写端口获取更高的读/写稳定性与吞吐率。参考文献[45]提出了局部计算单元，用多个存储单元对应一个计算单元，以使用更紧凑的存储布局，减小面积代价。参考文献[46]和[47]通过电路级近似计算，借助神经网络的重训练恢复精度，同时降低了计算电路的代价。此外，除了定点格式，参考文献[48]～[50]还支持多种不同的浮点格式，以获得更优的片上精度。其中很多技术可以很容易地应用到基于其他存储介质的存内计算中，也为 ROM-SRAM 混合存内计算的电路和架构设计提供了参考。

现有存内计算芯片和理想状态的对比如图 4 所示，当前存内计算芯片仍面临片上存储容量受限的核心挑战。受限于芯片制造工艺的物理约束（尤其是良率控制要求），单芯片的集成密度存在理论上限，导致神经网络推理所需的完整权重矩阵无法完全在片上存储，仍需依赖片外 DRAM 进行权重动态加载。这种频繁的片内外数据交互不仅抵消了存算架构的能效优势，而且制约了系统整体性能的提升。因此，如何在既定工艺条件下优化存算阵列的集成密度进而构建高效的数据复用机制，成为突破现有技术瓶颈、释放存内计算理论潜能的关键研究方向。

3　基于 ROM 的大容量存内计算电路设计

3.1 电流域 ROM CiM

为了将更大量的权重部署在片上，同时避免启动系统时从片外加载权重，就需要探索更高密度、大容量的 ROM

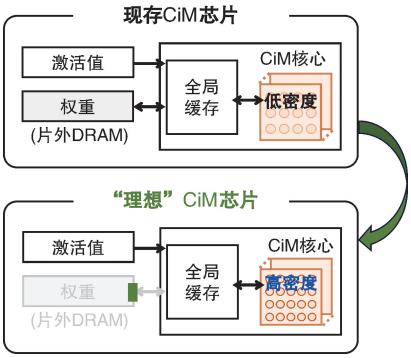


图 4　现有存内计算芯片和理想状态的对比

Fig. 4　Comparison between the existing CiM chips and the ideal CiM chips

存内计算电路。在成熟的 CMOS 工艺下，基于 ROM 的存内计算可以很好地满足上述需求。为了获得更高的存储密度，利用存内计算特点进行优化的新电路结构是值得探索的。

图 5 展示了一个采用电流域计算方式的 ROM 存内计算电路[51]。通过将单元内晶体管固定连接到多条位线之一，从而实现 ROM 的存储。对于一个 8 位权重网络，用 5 个 ROM 单元来存储：最高 2 位分别使用一个单比特 ROM 单元存储来保证计算精度，晶体管有两种不同的连接方式保证 1 位的存储；而后面 6 位两两分组后用 3 个双比特 ROM 单元存储，晶体管有 4 种不同的连接方式保证 2 位的存储。具体来说，对于单比特 ROM 单元，晶体管连接到 RBL 上表示 1，不连表示 0；对于双比特 ROM 单元，则是通过连接到 3 条不同的 RBL 上或者不连来表示相应的不同权重。在制造时，这几条 RBL 走在不同的金属层上，ROM 单元内的晶体管只需选择与哪层金属线相连即可，没有额外的走线代价。

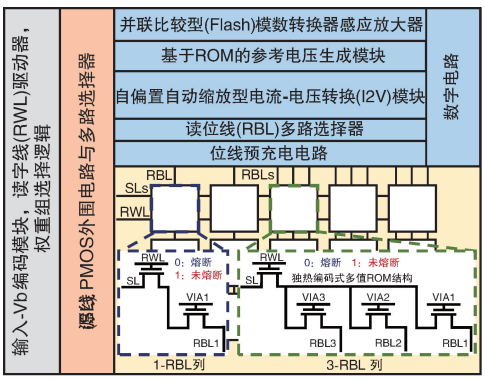


图 5　一种电流域 ROM 存内计算设计[51]

Fig. 5　A design of the current - domain ROM CiM[51]

计算时，首先选择参与计算的权重位，并在对应权重位上的几条位线上预充电。然后打开 RWL，通过改变源线 SL 驱动电路中源线 PMOS 的栅极电压一次性将 2 位输入送入电路，使得每条 SL 连接到 RBL 之一，从而根据每条 RBL 上连接的晶体管数量产生累加电流，改变 RBL 上的电压。最后通过 ADC 读出相应 RBL 上的电压即可完成乘累加运算。

3.2　电荷域 ROM CiM

图 6 展示了一个采用电荷域计算方式的 ROM 存内计算电路[52]。为了平衡 ROM 单元的小面积和周边电路相对较大的面积，阵列采用了局部计算单元思路：ROM 阵列由多个 ROM 块组成，每次每个 ROM 块中只有一个 ROM 单元被开启，而通过同时开启多个 ROM 块达到较高的并行度。

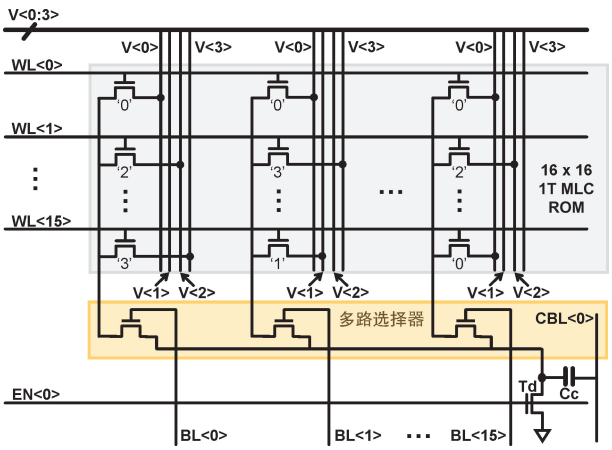


图 6　一种电荷域 ROM 存内计算设计[52]

Fig. 6　A design of the charge - domain ROM CiM[52]

在图 6 所示电路中，每个 ROM 块由 256 个 ROM 单元、16 个列选 MUX、1 个计算电容 Cc 和 1 个放电晶体管 Td 组成。在每个 ROM 单元内，晶体管通过选择连接到电源线 V<0:3>其中之一，实现 2 位存储功能。

在计算时，首先控制所有 CBL 和 EN 来清除所有计算电容 Cc 上的电荷。然后使 CBL 保持浮空状态，根据本次参与计算的行列打开对应的 BL，并根据输入 0 或 1 将相应电平送入到相应 WL 和 EN 上：如果输入为 0，那么将 WL 设置为低电平，EN 设置为高电平，此时计算电容 Cc 左极板电压变为 0；如果输入为 1，那么将 WL 设置为高电平，EN 设置为低电平，此时相应电源线的电压会通过单元晶体管和 MUX 晶体管送到计算电容 Cc 左极板。通过此操作，完成了 1 位输入与 2 位权重的“与”运算，即乘法运算。进一步，同一条 CBL 上所有计算电容左极板都变为特定电平，CBL 上会进行电荷重分配，其电压等于其上所有计算电容左极板电压的平均值，进而完成列上的求和。只需要后续通过 ADC 将 CBL 上的电压读出，即可完成一次乘累加运算。

在此基础上，参考文献[53]注意到，由于晶体管除了栅极需要控制是否开启以外，漏极和源极均可以存在不同

的连接方式，从而通过对单元的时分复用方式，进一步提高每个 ROM 单元内可以存储的位数。一种更高单元比特数的电荷域 ROM 存内计算设计如图 7 所示。以第 0 列 ROM 为例，每个 ROM 单元中的晶体管左端连接到 4 条位线 S<0，2，4，6>之一，表示高 2 位；晶体管右端连接到 4 条位线 S<1，3，5，7>之一，表示低 2 位。通过配置 EN_M 和 EN_L 两个信号，可以选择计算权重高 2 位的乘法或是权重低 2 位的乘法。以计算高 2 位为例（反之亦然）：此时 EN_M 为高、EN_L 为低，右侧 4 条位线 S<1，3，5，7>会直接与计算电容 Cc 的极板相连，而左侧 4 条位线 S<0，2，4，6>则会分别与 4 条电源线 V<0:3>相连，从而在同一列中所有计算电容上完成 1 位输入和高 2 位权重的乘累加运算。

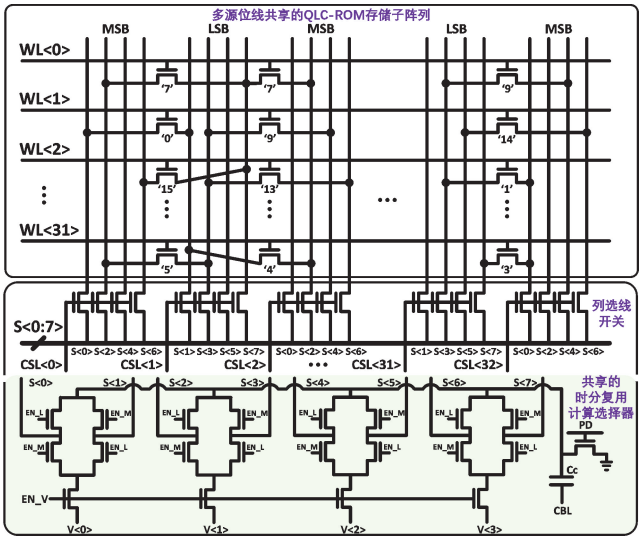


图 7　一种更高单元比特数的电荷域 ROM 存内计算设计[53]

Fig. 7　A design of the charge - domain ROM CiM with more bits per cell[53]

3.3　数字域 ROM CiM

传统的 ROM 电路结构可以很容易地应用到数字域存内计算之中。最朴素的想法是，首先从每个 ROM 块中读出 1 位权重，然后将其与输入进行“与”运算后直接将其送入加法树中，便可以得到乘累加结果。但是，由于对大容量的需求，仍然需要通过进一步的设计继续提高存储密度。

一种思路是将输入地址到读出权重的过程全部改由数字电路实现[54]，将 ROM 读出逻辑融合到数字电路中，从而从片上移除 ROM 阵列，只保留下数字电路，如图 8 所示。图中左侧展示的是等价的 ROM 结构，通过 2 位或多位地址选出 8 位（W0～W7）送到加法树中进行乘累加。在将 ROM 阵列转换成数字电路后，可以通过逻辑优化移除加法树中的一些冗余结构。在图 8 所示的例子中，对于各个地址而言，W7 全为 0，而 W6 全为 1，从而可以在加法树中优化掉一个加法器。对于其他位，根据其权重模式也可以有相应的化简方法。

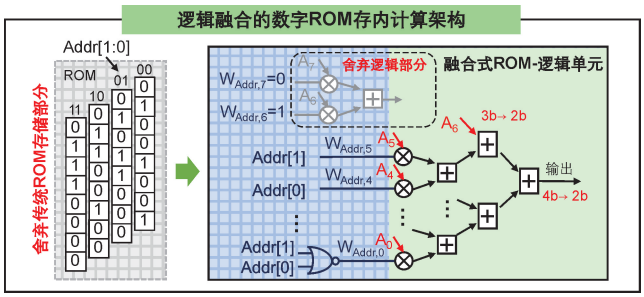


图 8　一种基于逻辑融合的数字域 ROM 存内计算设计[54]

Fig. 8　A design of the digital - domain ROM CiM with logic fusion[54]

另一种思路是继续提高 ROM 的密度。参考文献[55]注意到第 3.1 节和第 3.2 节中介绍的 ROM 电路中，单元内的晶体管本质上是一个开关，并且一个 ROM 块中的 ROM 单元数量较多，但由于受周边电路面积限制，每次一个 ROM 块中只能开启其中一个晶体管。因此这些开关在大部分时候是处于闲置状态的，进而可以通过将 ROM 单元内晶体管移除来实现全金属层的 ROM，从而能得到 3D - METRO。

一种基于更高密度 ROM 单元的数字域 ROM 存内计算设计如图 9 所示，3D - METRO 在单元内引入了一大一小两个寄生电容，通过大小寄生电容与 BL 和 BLB 的连接关系实现存储。在读取时，相应 ROM 单元对应的 BL 和 BLB 浮空，相应 WL 设置为高电平，其余 WL 保持低电平。由于单元内寄生电容大小的差异，BL 和 BLB 上会出现电压差，进而根据电压差得到单元内存储的数据。此外，由于 ROM 块可以完全在金属层上实现，因此可以将 ROM 块放在周边电路上方，且能在原生 CMOS 工艺下实现 ROM 块的 3D 堆叠，进一步提高存储密度。

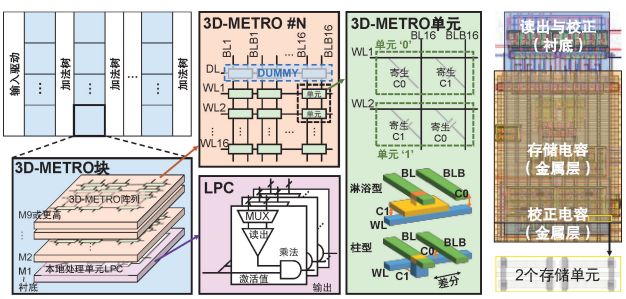


图 9　一种基于更高密度 ROM 单元的数字域 ROM 存内计算设计[55]

Fig. 9　A design of the digital - domain ROM CiM with higher ROM density[55]

3.4　比较与分析

就提高 ROM 存内计算密度的方式而言，上述 5 篇工

作[51-55]之间的关系如图 10 所示。存储密度由两方面决定：单元的密度以及在单元的面积中所存储的比特数。

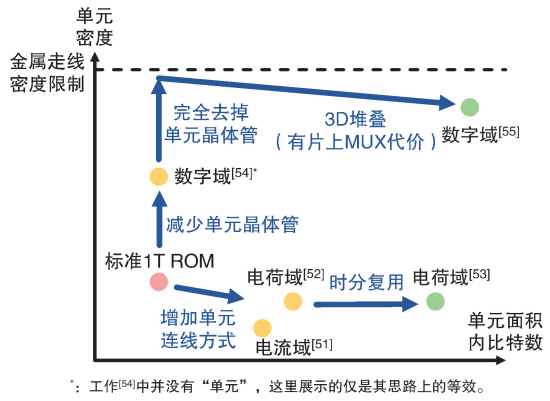


图 10　提高 ROM 存内计算存储密度的方式

Fig. 10　Approaches to higher memory density of ROM CiM

若要提升单元密度，则需要减少平均到每个单元中的晶体管数量，从而可以使用更紧密的布线方式，如数字域工作[54]中通过逻辑融合等效减少单元中晶体管数量，以及数字域工作[55]中完全去掉晶体管来逼近金属走线密度的上限。

若想提高在单元面积内存储的比特数，则有空间和时间两种方式：空间上，可以通过增加单元的连线方式[51-52]让单元中存储更多数据，或者在垂直于单元平面的方向上叠放多个单元[55]增加一个平面单元面积内放置的单元数量；而在时间上，可以通过时分复用的方式[53]重用单元，从而实现在一个单元中存储更多数据。

表 1 展示了上述几个工作的对比。从表中数据来看，通过在电路中采用 ROM 存内计算可以获得相当高的存储密度，同时存储密度的改善基本与以上分析的趋势相同。对于其中的部分细节问题分析如下：首先，从电荷域工作[52]到电荷域工作[53]，由于存在工艺演进的因素，存储密度的改善不够直观；而在参考文献[56]中，作者在 28 nm 工艺下采用了和参考文献[52]相同的 ROM 结构，其 ROM 存内计算部分的阵列级存储密度约为 15 Mb/mm^2，因此电荷域工作[53]对存储密度确实有显著的提升。从数字域工作[54]到数字域工作[55]，虽然后者为仿真数据，实测性能会有所下降，且可能因为流片工艺对金属层数的限制影响 ROM 堆叠层数，但考虑到其密度本身足够高，同时具有额外金属层带来的 3D 堆叠的可扩展性，其仍然具有很高的密度潜力。

表 1　现有 ROM 存内计算工作对比

Table 1　Comparison of existing ROM CiM works

参 数	电流域[51] ESSCIRC'23	电荷域[52] JSSC'24	电荷域[53] A-SSCC'24	数字域[54] ASP-DAC'25	数字域[55]仿真数据 ASP-DAC'25
CMOS 工艺节点/nm	65	65	28	65	28
输入位数	1～8	1～8	1～8	1～8	8
权重位数	1～8	2/4/6/8	2/4/6/8	4	8
ADC 位数	3	5	5	N/A	N/A
供电电压/V	1.1	0.7～1.2	0.6～1.1	0.6～1.2	0.9
工作频率/MHz	100	50～210	40～200	130～800	200
8b×8b 能量效率/(TOPS·W^{-1})	66.21	1.24～4.33	8.49	9.00～38.00	1.20
8b×8b 面积效率/(TOPS·mm^{-2})	0.230	0.021～0.087	0.060	0.550～2.060	1.280
存储密度/(Mb·mm^{-2})	0.059	3.890	19.660	0.476	165.600

对于第 3.1 节和第 3.2 节中介绍的模拟域存内计算，在低功耗和高并行度计算方面具有一定的优势。然而，由于模拟域计算精度容易受到工艺偏差和噪声的影响，需要一定的额外代价来降低这些因素的影响，而同时还需要平衡好 ROM 阵列和周边电路的比例关系，避免抵消 ROM 的高密度优势。如在电流域工作[51]中，利用 ROM 结构产生 ADC 的参考电压，从而减小 ADC 代价并提高对抗工艺偏差的能力。而在两篇电荷域工作[52-53]中，则采用了相对较大的电容和 ADC，与此相应的是，通过增加每个 ROM 块内的 ROM 单元数量保证面积的平衡。

对于第 3.3 节中介绍的数字域存内计算，可以保证较好的计算精度和可靠性。然而，大面积的加法树仍然可能成为制约 ROM 阵列密度的瓶颈。对于参考文献[54]而言，每个 ROM 块对应的地址位数是值得讨论的，每个 ROM 块对应的地址位数越多，则同一个加法器树可以被更多的等效权重复用，但相应的是更难以化简的逻辑，因此需要选择合适的地址位数。对于参考文献[55]而言，虽然 ROM 阵列可以堆叠在计算电路上方，然而更多的 ROM 层意味着更大的 MUX 开销，过多的层数反而会降低密度。此外，由于电路结构的特殊性，SRAM 单元很难直接放在 ROM 阵列中，因此只能采用后文第 4.1 节中讨论的对权重的调整方式实现高灵活性的 ROM-SRAM

混合架构。如何在保持数字域 ROM 存内计算的高密度的前提下采用后文第 4.2 节讨论的 ROM－SRAM 混合架构，是值得进一步研究的。

4　ROM－SRAM 混合的神经网络调整结构

4.1　SRAM 部分对权重的调整

鉴于边缘端人工智能的需求，由 ROM 存内计算和 SRAM 存内计算混合而成的计算架构可以更加适应对边缘端人工智能加速的需求。

参考文献[57]首次引入 ROM－SRAM 混合存内计算架构。由于 ROM 具有不可改的特性，若需要更新完全基于 ROM 的存内计算的权重，则需要重新开模和流片，成本较高。而在参考文献[57]中，提出了将大容量、固定权重的 ROM 存内计算作为神经网络的主干，作为预训练模型，存储网络中的绝大部分权重，同时保证较低的面积开销；小容量、可变权重的 SRAM 存内计算则用于对神经网络进行调整，将预训练模型迁移到其他多种特定任务之中，满足不同任务的需求。因此，参考文献[57]成功将 ROM 的高密度特性引入到了存内计算系统之中，并保留了系统的可调整。

理论上来说，SRAM 存内计算部分可以对完整神经网络的权重进行调整。然而，在实际部署中，由于 SRAM 存内计算的密度受限，在片上无法大量部署可变权重。因此，除了特定应用（如边缘端的人脸识别等简单任务），将神经网络的部分层作为可变权重或者直接用 SRAM 权重加上 ROM 权重来调整权重等方案，会丧失掉 ROM 存内计算的高密度优势。因此，想要尽可能灵活地调整神经网络，同时保证如何用尽量少的可变权重来满足所需要的灵活性，成为一个挑战。

YOLoC [57] 采用了残差分支（ReBranch）技术来解决此挑战。利用 SRAM 存内计算部分对 ROM 存内计算部分的权重进行低秩微调，从而可以把在一种数据集上预训练的权重（存放在 ROM 中）迁移到同领域的其他数据集上。以图像分类为例，如果将 CIFAR－100 数据集上预训练的权重存储在 ROM 存内计算部分中，那么利用 SRAM 存内计算部分的调整能力便可以将此模型迁移到 CIFAR－10、Fashion－MNIST、Caltech－101 等数据集上。

ReBranch 结构由两条并行的路径组成，如图 11 所示。左侧的主干路径上是由 ROM 存内计算构成的权重固定的一组深度卷积层，而右侧的分支路径则由部署在 ROM 上的残差（解）压缩层和一组部署在 SRAM 上的残差卷积层组成。在分支路径上，通过固定权重的残差压缩和解压缩层，利用逐点卷积[58]对特征图的通道数进行变换，从而达到压缩部署在 SRAM 上的残差卷积层的参数量的目的。而由于运算的线性性质，对主干路径和分支路径得到的输出激活值求和，等价于将两条路径等效的变换张量求和，从而达到分支路径对主干路径进行低秩调整的结果。

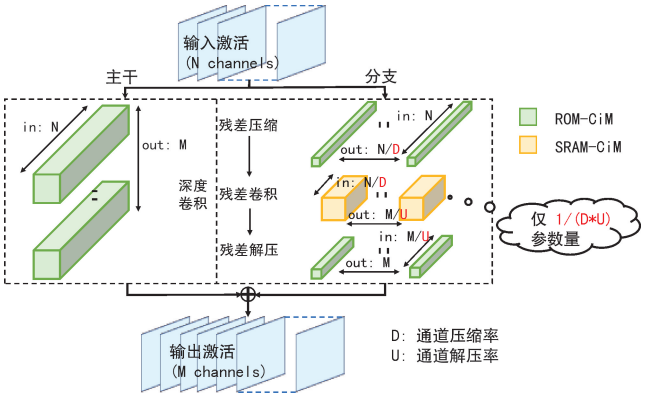


图 11　ReBranch 架构[57]

Fig. 11　Structure of ReBranch[57]

基于 ReBranch 技术可以搭建出由 ROM－SRAM 混合存内计算的 YOLoC 架构，如图 12 所示。YOLoC 架构由 ROM 存内计算、SRAM 存内计算及周边电路组成。ROM 存内计算部分负责主干路径的网络推理，可以将超过 90％的网络权重存储于片上；而 SRAM 部分参与分支路径的推理计算，对网络进行微调，且仅需要在系统启动时将少量权重从片外 DRAM 搬移到片上。

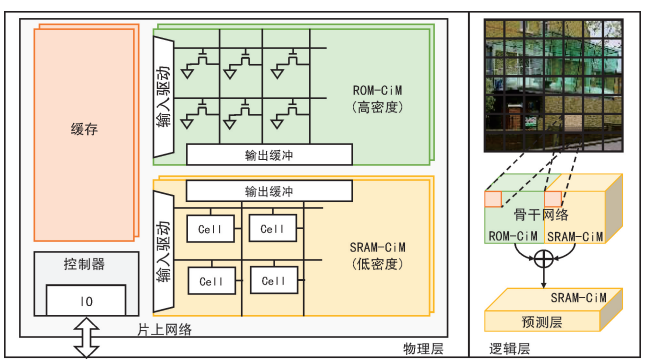


图 12　YOLoC 架构[57]

Fig. 12　Structure of YOLoC[57]

4.2　SRAM 部分对网络结构的调整

ReBranch 技术是对权重进行的调整，然而，这样的调整并无法对网络结构进行修改。一方面，仅修改权重无法将网络从一种任务迁移到不同域中的另一种任务上；另一方面，也无法将网络扩展得到更大的网络，如同为 ResNet[1]，就无法将预训练好的 ResNet－18 网络通过以上权重修改的方式变成 ResNet－101 等更大网络的一部分，以支持同种类但更复杂的任务。因此，还需要一种手段让 SRAM 中的可变权重直接修改网络的连接结构。考虑到此需求，参考文献[59]在 ROM－SRAM 混合存内计算系统重使用 HNN 技术[60]方案，在 ROM 中存储权重，而用 SRAM 中的可变权重来修改 ROM 权重之间的连接方式，

从而调整网络结构。

图 13 展示的 Hidden－ROM[59] 工作利用 HNN 技术实现了网络结构的可调。Hidden－ROM 的主体是一个 ROM－SRAM 混合阵列，包括多个 ROM－SRAM 混合单元。在每个混合单元中，存放了多个 ROM 单元用来存储一个或者多个权重，还有一个 SRAM 单元来控制对应的 ROM 单元权重是否有效。如图 13 展示的结构所示，若 SRAM 单元控制的开关闭合，则 ROM 的局部字线与输入相连，从而相应的 ROM 单元参与运算；若 SRAM 单元控制的开关断开，则相应的 ROM 单元无法被输入驱动，对应的神经网络参数被剪枝，不参与推理。

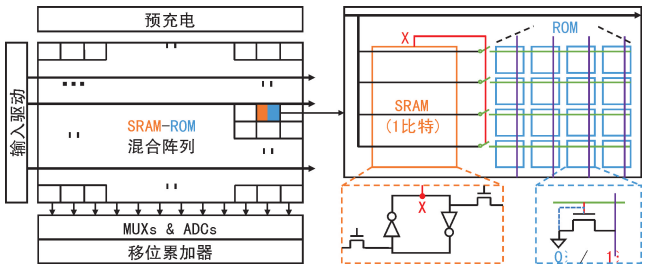


图 13　Hidden－ROM 结构[59]

Fig. 13　Structure of Hidden－ROM[59]

采用 Hidden－ROM 结构，除了可以修改网络权重以外，还可以通过时分复用方式对已有的模型结构进行扩展。例如在 ResNet[1] 中，相比于 ResNet－18，ResNet－101 等更大的模型可以通过增加相似结构的层来提升网络深度，从而在更复杂任务场景下获得更高的精度。而 HNN 通过 Kaiming 算法[61] 对预训练模型的权重进行初始化，在训练中通过权重掩码在初始权重中选择一个子网络。考虑到 Kaiming 初始化的参数只与卷积层输入维度相关，所以预训练出来的权重可以在相同大小的层之间复用。因此，通过加载不同的掩码配置方案可以将同一块固定权重配置成神经网络的不同层，进而实现网络结构的扩展。

4.3　比较与分析

表 2 展示了在 28 nm 工艺、8 位输入和 8 位权重设置下，基线 SRAM 存内计算[45] 和 YOLoC 及 Hidden－ROM 的阵列密度、系统能效和面积效率的对比。考虑到在计算过程中，SRAM 存内计算需要从片外加载数据，因此在系统级能效方面，ROM－SRAM 混合存内计算相比纯 SRAM 存内计算有显著的能效提升。

图 14 展示了在 CIFAR－100 数据集上预训练的 ResNet－18 模型迁移到其他数据集上时的精度表现。对于部署后迁移到同领域的简单任务，如 MNIST 数据集和 CIFAR－10 数据集，YOLoC 和 Hidden－ROM 都实现了接近软件的精度。而对于不同领域的简单任务，如人脸情感分类的 FER－2013 数据集，YOLoC 对权重的简单调整则无法满足迁移需求，而 Hidden－ROM 对网络结构的修改仍然能够维持较好的精度。而对于同领域的复杂任务（如 ImageNet 数据集），YOLoC 的迁移能力也无法胜任，而 Hidden－ROM 也有约 10％的精度损失。然而，由于 Hidden－ROM 的可扩展性，将其扩展为更大的模型 ResNet－101 后，由于模型本身表达能力增加，Hidden－ROM 的精度损失可以有明显的减小。

表 2　YOLoC 与 Hidden－ROM 对比[59]

Table 2　Comparison between YOLoC and Hidden－ROM[59]

参　数	SRAM	YOLoC	Hidden－ROM
外部存储容量/Mb	1.2	0	0
制造工艺	28 nm CMOS	—	—
预部署模型	ResNet－18	—	—
阵列面积/mm^2	0.32	0.48	0.44
阵列容量/Mb	0.06	1.26	N/A
阵列密度/(Mb·mm^{-2})	0.20	2.62	2.86
输入位数	8	—	—
权重位数	8	—	—
阵列级能效/(TOPS·W^{-1})	11.54	11.50	11.85
面积效率/(GOPS·mm^{-2})	94.25	119.40	140.30

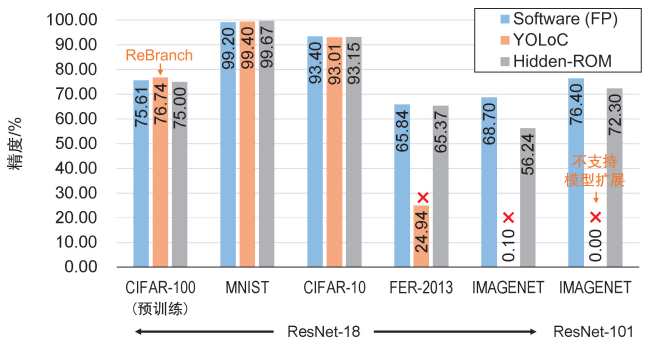


图 14　YOLoC 与 Hidden－ROM 执行不同任务的精度对比[59]

Fig. 14　Comparison between the accuracy of YOLoC and Hidden－ROM under different tasks[59]

总结来说，YOLoC 对权重的简单修改已经足以满足同领域内任务上的迁移，并且减少片外访存需求，极大地提高了硬件的能量效率。而对于更复杂的任务或者不同领域任务的迁移，Hidden－ROM 对网络结构的调整可以更好地适应，具有更强的灵活性和可扩展性。

5　ROM－SRAM 混合存内计算架构的挑战与展望

5.1　更大的片上容量

目前，基于 ROM 的存内计算芯片受制造工艺水平等因素的限制，权重容量仍难以达到 GB 量级。因此，目前

的 ROM－SRAM 混合存内计算架构中仍然只能部署相对较轻量的模型，而对于当前热门的 GB 量级乃至 TB 量级的大模型，基于 ROM－SRAM 混合架构的存内计算阵列仍然需要受到片外访存和通信的开销限制。

为了突破上述限制，进一步释放 ROM－SRAM 混合架构的潜力，先进封装技术是一种极有优势的解决方案。通过 3D 堆叠技术[62]可以将多个存储芯片在垂直方向堆叠，不同层的芯片采用硅通孔（Through Silicon Via，TSV）等技术进行连接，保证了数据在不同层之间的高效传输。此外，芯粒技术[63]可以将不同功能的芯粒利用封装集成在一起，可以极大提升芯片的集成度和灵活性，通过对存储芯粒和计算芯粒的合理组合，可以更好地提升存储容量，优化系统性能。

就 ROM－SRAM 混合存内计算本身来说，采用这两种技术并不会增加太多的成本：不同的管芯之间的差别仅在于 ROM 的权重，而修改 ROM 的权重只需要修改金属层的少量几张掩模版。与之相比，更值得讨论的问题在于，是否能通过某种方式对网络的划分—特别是对固定权重和可变权重的划分，以利用好先进封装技术带来的高并行的特点，降低用于通信的封装内网络的巨大开销[64]，从而能够达到突破掩模版尺寸极限的片上容量[65]，进一步扩大 ROM－SRAM 混合存内计算架构的设计空间，将其应用在 GB 量级乃至 TB 量级的大模型中，极大地扩展其应用范围。

5.2 LLM 背景下的灵活性提升

在第 4 章中，主要讨论对 CNN 的调整。而对于 LLM，也可以采用类似的方法进行模型调整。不同于 CNN，由于 LLM 预训练网络本身的表达能力充足，现有研究更集中于对其参数进行调整。因此，基于低秩参数调整的 LoRA 技术[31]更容易被采用到 ROM－SRAM 混合存内计算的架构中，如参考文献[56]中提出的基于 LoRA 的 ROM－SRAM 混合存内计算架构。对于令牌长度较短的场景，如边缘端的很多应用场景中只需要处理短序列，由于全连接层占主要的运算需求，因此将固定权重的 Q、K、V 生成以及后面的全连接层结构部署在 ROM 上，将动态矩阵乘法和对权重的低秩修改部署在 SRAM 中，可以极大降低对片外访存的需求。

而对于涉及长序列的应用，动态矩阵乘法可能会成为系统的瓶颈。在现有的 LLM 加速系统中，只使用少量 SRAM 存内计算电路中存储的权重，所带来的灵活性很难直接支持动态矩阵乘法加速的需求。因此，对于长序列大模型的支持目前仍然是 ROM－SRAM 混合存内计算架构面对的挑战。从软件角度而言，目前有 QuickLLaMA[66]和 ThinK[67]等技术可以对 K、V 缓存进行剪枝，还有 H2O[68]和 SnapKV[69]等技术来选择关键令牌保留，这些技术有利于减小动态矩阵乘法的规模。而从硬件的角度而言，如第 4.2 节中所介绍的 Hidden－ROM，进一步探索 SRAM 和 ROM 在阵列级的融合，从而让 SRAM 控制 ROM 的计算而非直接参与计算，以此方式来获得更强的灵活性，也有可能可以突破现有动态矩阵乘法所带来的瓶颈。通过减小 ROM－SRAM 混合存内计算架构中动态矩阵乘法的代价，可以为 ROM－SRAM 混合存内计算在更多长序列场景中的应用提供可能。

6 结 论

神经网络作为人工智能领域最具影响力的范式之一，深刻影响着计算机视觉、自然语言处理等多个领域，然而神经网络庞大的参数量则对其在边缘端的部署提出了新的挑战。在边缘端，一方面，为了应用的灵活性，要求计算电路能够通过少量可变存储来微调网络，实现网络在任务间的迁移；另一方面，为了降低能耗延迟等开销，需要实现大容量的片上存储来减小片外访存开销。在成熟的 CMOS 工艺下，ROM－SRAM 混合存内计算架构是一种很有潜力的方案。得益于 ROM 存内计算本身的高密度，以及相应的进一步提升密度的方法，可以使得神经网络的大部分权重存储在片内，从而减小片外访存开销；与此同时，针对 ROM 不可修改的问题，SRAM 可以通过重载神经网络权重乃至调整神经网络结构的方式为边缘端计算提供足够的灵活性。此外，对于现有 ROM－SRAM 混合存内计算架构难以处理的挑战，如 GB 量级乃至 TB 量级的超大规模模型的部署问题，以及 LLM 中长序列输入面临的大量动态矩阵乘等，也存在着可行的解决方案，如在硬件上采用先进封装、软件上考虑令牌剪枝等方案，从而有可能极大地扩展 ROM－SRAM 混合存内计算架构的设计空间和应用范围。

参考文献

[1] HE K，ZHANG X，REN S，et al. Deep Residual Learning for Image Recognition[C]//CVPR2016. Las Vegas，NV，USA：IEEE，2016：770－778.

[2] BLATTMANN A，DOCKHORN T，KULAL S，et al. Stable Video Diffusion：Scaling Latent Video Diffusion Models to Large Datasets：arXiv：2311.15127[M/OL]. arXiv，2023.

[3] TOUVRON H，LAVRIL T，IZACARD G，et al. LLaMA：Open and Efficient Foundation Language Models：arXiv：2302.13971[M]. arXiv，2023.

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

[5] YENDURI G，M R，G C S，et al. Generative Pre－trained

Transformer: A Comprehensive Review on Enabling Technologies, Potential Applications, Emerging Challenges, and Future Directions: arXiv: 2305.10435[M]. arXiv, 2023.

[6] ZHANG S, ROLLER S, GOYAL N, et al. OPT: Open Pre-trained Transformer Language Models: arXiv: 2205.01068 [M]. arXiv, 2022.

[7] YE L, WANG Z, LIU Y, et al. The Challenges and Emerging Technologies for Low-Power Artificial Intelligence IoT Systems[J]. IEEE Transactions on Circuits and Systems I: Regular Papers, 2021, 68(12): 4821-4834.

[8] ZHOU S, WU Y, NI Z, et al. DoReFa-Net: Training Low Bitwidth Convolutional Neural Networks with Low Bitwidth Gradients: arXiv: 1606.06160[M]. arXiv, 2018.

[9] CHEN T, BAO H, HUANG S, et al. THE-X: Privacy-Preserving Transformer Inference with Homomorphic Encryption: arXiv: 2206.00216[M]. arXiv, 2022.

[10] SONG J, WANG Y, TANG X, et al. A 16Kb Transpose 6T SRAM In-Memory-Computing Macro Based on Robust Charge-Domain Computing[C]//2021 IEEE Asian Solid-State Circuits Conference (A-SSCC). IEEE, 2021: 1-3.

[11] WULF W, MCKEE S A. Hitting the Memory Wall: Implications of the Obvious[R]. USA: University of Virginia, 1994.

[12] SOHAN M, SAI RAM T, RAMI REDDY C V. A review on yolov8 and its advancements[C]//International Conference on Data Intelligence and Cognitive Informatics. Springer, Singapore, 2024: 529-545.

[13] JOONYOUNG KIM, YOUNSU KIM. HBM: Memory solution for bandwidth-hungry processors[C]//2014 IEEE Hot Chips 26 Symposium (HCS). Cupertino, CA, USA: IEEE, 2014: 1-24.

[14] SMITH A, LOH G H, WUU J, et al. AMD Instinct™ MI300X Accelerator: Packaging and Architecture Co-Optimization[C]//2024 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits). Honolulu, HI, USA: IEEE, 2024: 1-2.

[15] PREZIOSO M, MERRIKH-BAYAT F, HOSKINS B D, et al. Training and Operation of an Integrated Neuromorphic Network Based on Metal-Oxide Memristors[J]. Nature, 2015, 521(7550): 61-64.

[16] ZHANG J, WANG Z, VERMA N. A Machine-Learning Classifier Implemented in a Standard 6T SRAM Array [C]//2016 IEEE Symposium on VLSI Circuits (VLSI-Circuits). 2016: 1-2.

[17] AGRAWAL A, JAISWAL A, LEE C, et al. X-SRAM: Enabling In-Memory Boolean Computations in CMOS Static Random Access Memories[J]. IEEE Transactions on Circuits and Systems I: Regular Papers, 2018, 65(12): 4219-4232.

[18] BISWAS A, CHANDRAKASAN A P. CONV-SRAM: An Energy-Efficient SRAM With In-Memory Dot-Product Computation for Low-Power Convolutional Neural Networks[J]. JSSC, 2019, 54(1): 217-230.

[19] SI X, CHEN J J, TU Y N, et al. A Twin-8T SRAM Computation-in-Memory Macro for Multiple-Bit CNN-Based Machine Learning[C]//2019 IEEE International Solid-State Circuits Conference-(ISSCC). IEEE, 2019: 396-398.

[20] XIE S, NI C, SAYAL A, et al. eDRAM-CIM: Compute-In-Memory Design with Reconfigurable Embedded-Dynamic-Memory Array Realizing Adaptive Data Converters and Charge-Domain Computing[C]//ISSCC'21. San Francisco, CA, USA: IEEE, 2021: 248-250.

[21] HA S, KIM S, HAN D, et al. A 36.2 dB High SNR and PVT/Leakage-Robust eDRAM Computing-In-Memory Macro With Segmented BL and Reference Cell Array[J]. IEEE Transactions on Circuits and Systems II: Express Briefs, 2022, 69(5): 2433-2437.

[22] XIA L, GU P, LI B, et al. Technological Exploration of RRAM Crossbar Array for Matrix-Vector Multiplication [J]. Journal of Computer Science and Technology, 2016, 31(1): 3-19.

[23] PAN Y, OUYANG P, ZHAO Y, et al. A Multilevel Cell STT-MRAM-Based Computing in-Memory Accelerator for Binary Convolutional Neural Network[J]. IEEE Transactions on Magnetics, 2018, 54(11): 1-5.

[24] CAI H, BIAN Z, HOU Y, et al. A 28nm 2Mb STT-MRAM Computing-in-Memory Macro with a Refined Bit-Cell and 22.4-41.5 TOPS/W for AI Inference[C]//2023 IEEE International Solid-State Circuits Conference (ISSCC). IEEE, 2023: 500-502.

[25] SOLIMAN T, MULLER F, KIRCHNER T, et al. Ultra-Low Power Flexible Precision FeFET Based Analog In-Memory Computing[C]//2020 IEEE International Electron Devices Meeting. San Francisco, CA, USA: IEEE, 2020: 29.2.1-29.2.4.

[26] WANG L, LI W, ZHOU Z, et al. A Flash-SRAM-ADC-Fused Plastic Computing-in-Memory Macro for Learning in Neural Networks in a Standard 14nm FinFET Process [C]//2024 IEEE International Solid-State Circuits Conference (ISSCC). IEEE, 2024, 67: 582-584.

[27] JHANG C J, XUE C X, HUNG J M, et al. Challenges and Trends of SRAM-Based Computing-In-Memory for AI Edge Devices[J]. IEEE Transactions on Circuits and Systems I: Regular Papers, 2021, 68(5): 1773-1786.

[28] HUNG J M, JHANG C J, WU P C, et al. Challenges and Trends of Nonvolatile In-Memory-Computation Circuits for AI Edge Devices[J]. IEEE Open Journal of the Solid-State Circuits Society, 2021: 1.

[29] YAN B, HSU J L, YU P C, et al. A 1.041 - Mb/mm² 27.38 - TOPS/W Signed - INT8 Dynamic - Logic - Based ADC - Less SRAM Compute - in - Memory Macro in 28nm with Reconfigurable Bitwise Operation for AI and Embedded Applications[C]//2022 IEEE International Solid - State Circuits Conference (ISSCC). IEEE, 2022, 65: 188 - 190.

[30] CHATTERJEE K, KIM S, KARBASIAN G, et al. Self - Aligned, Gate Last, FDSOI, Ferroelectric Gate Memory Device With 5.5 - nm $Hf_{0.8}Zr_{0.2}O_2$, High Endurance and Breakdown Recovery[J]. IEEE Electron Device Letters, 2017, 38(10): 1379 - 1382.

[31] HU E J, SHEN Y, WALLIS P, et al. LoRA: Low - Rank Adaptation of Large Language Models: arXiv: 2106.09685 [M]. arXiv, 2021.

[32] HINTON G E, SALAKHUTDINOV R R. Reducing the Dimensionality of Data with Neural Networks[J]. Science, 2006, 313(5786): 504 - 507.

[33] VASWANI A, SHAZEER N, PARMAR N, et al. Attention Is All You Need: arXiv: 1706.03762[M]. arXiv, 2023.

[34] YOSINSKI J, CLUNE J, BENGIO Y, et al. How Transferable are Features in Deep Neural Networks[J]. Advances in Neural Information Processing Systems, 2014, 27.

[35] KANG M, KEEL M S, SHANBHAG N R, et al. An Energy - Efficient VLSI Architecture for Pattern Recognition via Deep Embedding of Computation in SRAM[C]//2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2014: 8326 - 8330.

[36] SI X, CHEN J J, TU Y N, et al. A Twin - 8T SRAM Computation - in - Memory Unit - Macro for Multibit CNN - Based AI Edge Processors[J]. IEEE Journal of Solid - State Circuits, 2019, 55(1): 189 - 202.

[37] VALAVI H, RAMADGE P J, NESTLER E, et al. A 64 - Tile 2.4 - Mb In - Memory - Computing CNN Accelerator Employing Charge - Domain Compute[J]. IEEE Journal of Solid - State Circuits, 2019, 54(6): 1789 - 1799.

[38] JIANG Z, YIN S, SEO J S, et al. C3SRAM: An In - Memory - Computing SRAM Macro Based on Robust Capacitive Coupling Computing Mechanism[J]. IEEE Journal of Solid - State Circuits, 2020, 55(7): 1888 - 1897.

[39] YANG J, KONG Y, WANG Z, et al. 24.4 Sandwich - RAM: An Energy - Efficient In - Memory BWN Architecture with Pulse - Width Modulation[C]//2019 IEEE International Solid - State Circuits Conference -(ISSCC). IEEE, 2019: 394 - 396.

[40] CHIH Y D, LEE P H, FUJIWARA H, et al. An 89TOPS/W and 16.3 TOPS/mm² All - Digital SRAM - Based Full - Precision Compute - In Memory Macro in 22nm for Machine - Learning Edge Applications[C]//2021 IEEE International Solid - State Circuits Conference (ISSCC). IEEE, 2021, 64: 252 - 254.

[41] YUAN Y, YANG Y, WANG X, et al. A 28nm 72.12 TFLOPS/W Hybrid - Domain Outer - Product Based Floating - Point SRAM Computing - in - Memory Macro with Logarithm Bit - Width Residual ADC[C]//2024 IEEE International Solid - State Circuits Conference (ISSCC). IEEE, 2024, 67: 576 - 578.

[42] CHEN X, LI S, ZHANG Z, et al. A 28nm 64kb Bit - Rotated Hybrid - CIM Macro with an Embedded Sign - Bit - Processing Array and a Multi - Bit - Fusion Dual - Granularity Cooperative Quantizer[C]//2025 IEEE International Solid - State Circuits Conference (ISSCC). IEEE, 2025, 68: 260 - 262.

[43] ZHANG Y, XU L, DONG Q, et al. Recryptor: A Reconfigurable Cryptographic Cortex - M0 Processor with In - Memory and Near - Memory Computing for IoT Security[J]. IEEE Journal of Solid - State Circuits, 2018, 53(4): 995 - 1005.

[44] FUJIWARA H, MORI H, ZHAO W C, et al. A 5 - nm 254 - TOPS/W 221 - TOPS/mm² Fully - Digital Computing - in - Memory Macro Supporting Wide - Range Dynamic - Voltage - Frequency Scaling and Simultaneous MAC and Write Operations[C]//2022 IEEE International Solid - State Circuits Conference (ISSCC). IEEE, 2022, 65: 1 - 3.

[45] SI X, TU Y N, HUANG W H, et al. A Local Computing Cell and 6T SRAM - Based Computing - in - Memory Macro with 8 - b MAC Operation for Edge AI Chips[J]. IEEE Journal of Solid - State Circuits, 2021, 56(9): 2817 - 2831.

[46] DIAO H, HE Y, LI X, et al. A Multiply - Less Approximate SRAM Compute - in - Memory Macro for Neural - Network Inference[J]. IEEE Journal of Solid - State Circuits, 2024.

[47] YUAN Y, ZHANG B, YANG Y, et al. A 28nm 192.3 TFLOPS/W Accurate/Approximate Dual - Mode - Transpose Digital 6T - SRAM CIM Macro for Floating - Point Edge Training and Inference[C]//2025 IEEE International Solid - State Circuits Conference (ISSCC). IEEE, 2025, 68: 258 - 260.

[48] WANG Y, YANG X, QIN Y, et al. A 28nm 83.23 TFLOPS/W POSIT - Based Compute - in - Memory Macro for High - Accuracy AI Applications[C]//2024 IEEE International Solid - State Circuits Conference (ISSCC). IEEE, 2024, 67: 566 - 568.

[49] WANG X, JIAO T, YANG Y, et al. A 28nm 17.83 - to - 62.84 TFLOPS/W Broadcast - Alignment Floating - Point CIM Macro with Non - Two's - Complement MAC for CNNs and Transformers[C]//2025 IEEE International Solid - State Circuits Conference (ISSCC). IEEE, 2025, 68: 254 - 256.

[50] YUE Z, XIANG X, WANG Y, et al. A 51.6 TFLOPs/W Full -

Datapath CIM Macro Approaching Sparsity Bound and $<2^{-30}$ Loss for Compound AI[C]//2025 IEEE International Solid-State Circuits Conference (ISSCC). IEEE, 2025, 68: 1-3.

[51] SEHGAL R, MEHRA R, NI C, et al. Compute-MLROM: Compute-in-Multi Level Read Only Memory for Energy Efficient Edge AI Inference Engines[C]//ESSCIRC 2023 IEEE 49th European Solid State Circuits Conference (ESSCIRC). 2023: 37-40.

[52] YIN G, CHEN Y, ZHOU M, et al. Cramming More Weight Data onto Compute-in-Memory Macros for High Task-Level Energy Efficiency Using Custom ROM with 3984-Kb/mm^2 Density in 65-nm CMOS[J]. IEEE Journal of Solid-State Circuits, 2024, 59(6): 1912-1925.

[53] CHEONG L A, WANG C, ZHOU M, et al. A 28nm 166.9 TOPS/W x Mb/mm^2 DRAM-Free QLC Compute-in-ROM Macro Supporting High Task-Level Inference Energy Efficiency for Tiny AI Edge Devices[C]//2024 IEEE Asian Solid-State Circuits Conference (A-SSCC). IEEE, 2024: 1-3.

[54] YU T, LIAO T, ZHOU M, et al. DCiROM: A Fully Digital Compute-in-ROM Design Approach to High Energy Efficiency of DNN Inference at Task Level[C]//2025 30th Asia and South Pacific Design Automation Conference (ASP-DAC). IEEE, 2025: 100-105.

[55] CHEN Y, DU X, YIN G, et al. 3D-METRO: Deploy Large-Scale Transformer Model on a Chip Using Transistor-Less 3D-Metal-ROM-Based Compute-in-Memory Macro[C]//2025 30th Asia and South Pacific Design Automation Conference (ASP-DAC). IEEE, 2025: 642-647.

[56] YIN G, CHEN Y, LEE M, et al. A 28nm 8928Kb/mm^2-Weight-Density Hybrid SRAM/ROM Compute-in-Memory Architecture Reducing $>95\%$ Weight Loading from DRAM[C]//2024 IEEE Custom Integrated Circuits Conference (CICC), 2024: 1-2.

[57] CHEN Y, YIN G, TAN Z, et al. YOLoC: Deploy Large-Scale Neural Network by ROM-Based Computing-in-Memory Using Residual Branch on a Chip[C]//Proceedings of the 59th ACM/IEEE Design Automation Conference. San Francisco California: ACM, 2022: 1093-1098.

[58] SZEGEDY C, LIU W, JIA Y, et al. Going Deeper with Convolutions[M/OL]. 2014. arXiv: 1409.4842.

[59] CHEN Y, YIN G, LEE M, et al. Hidden-ROM: A Compute-in-ROM Architecture to Deploy Large-Scale Neural Networks on Chip with Flexible and Scalable Post-Fabrication Task Transfer Capability[C]//Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design. San Diego California: ACM, 2022: 1-9.

[60] RAMANUJAN V, WORTSMAN M, KEMBHAVI A, et al. What's Hidden in a Randomly Weighted Neural Network[C]//2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Seattle, WA, USA: IEEE, 2020: 11890-11899.

[61] HE K, ZHANG X, REN S, et al. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification[C]//2015 IEEE International Conference on Computer Vision (ICCV). Santiago, Chile: IEEE, 2015: 1026-1034.

[62] LOH G H. 3d-stacked memory architectures for multi-core processors[C]//ISCA'08: Proceedings of the 35th Annual International Symposium on Computer Architecture. USA: IEEE Computer Society, 2008: 453-464.

[63] KRISHNAN G, MANDAL S K, PANNALA M, et al. SIAM: Chiplet-Based Scalable In-Memory Acceleration with Mesh for Deep Neural Networks[J]. ACM Trans. Embed. Comput. Syst., 2021, 20(5s).

[64] SHAO Y S, CLEMONS J, VENKATESAN R, et al. Simba: Scaling deep-learning inference with multi-chip-module-based architecture[C]//Proceedings of the 52nd Annual IEEE/ACM International Symposium on Microarchitecture, 2019: 14-27.

[65] NAFFZIGER S, BECK N, BURD T, et al. Pioneering Chiplet Technology and Design for the AMD EPYC and Ryzen Processor Families: Industrial Product[C]//2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA). IEEE, 2021: 57-70.

[66] LI J, SHI H, JIANG X, et al. QuickLLaMA: Query-aware Inference Acceleration for Large Language Models: arXiv: 2406.07528[M]. arXiv, 2024.

[67] XU Y, JIE Z, DONG H, et al. ThinK: Thinner Key Cache by Query-Driven Pruning: arXiv: 2407.21018 [M]. arXiv, 2024.

[68] ZHANG Z, SHENG Y, ZHOU T, et al. H2O: Heavy-Hitter Oracle for Efficient Generative Inference of Large Language Models [C]//Advances in Neural Information Processing Systems, 2023: 34661-34710.

[69] LI Y, HUANG Y, YANG B, et al. SnapKV: LLM Knows What You are Looking for Before Generation[C]//Advances in Neural Information Processing Systems, 2024: 22947-22970.

(责任编辑:薛士然)