一种使用忆阻器的 MobileNetV3 新型计算范式

Jiale Li*†, Zhihang Liu†, Sean Longyu Ma†, Chiu-Wing Sham†,Chong Fu‡
† School of Computer Science, The University of Auckland, Auckland, New Zealand
‡ School of Computer Science and Engineering, Northeastern University, Shenyang, China
{jli990, zliu604}@aucklanduni.ac.nz, {sean.ma, b.sham}@auckland.ac.nz, fuchong@mail.neu.edu.cn

摘要—机器学习领域的进步与其在专用硬件加速器（如 GPU 和 TPU）方面的同步进展密不可分。然而，更大模型和更多数据所带来的迅速增长的计算需求已经成为进一步推进机器学习的主要瓶颈，尤其是在移动设备和边缘设备中。目前，基于忆阻器的神经形态计算范式提供了一个有前景的解决方案。在本研究中，我们引入了一种基于忆阻器的 MobileNetV3 神经网络计算范式，并提供了完整的验证框架。结果表明，这种计算范式在 CIFAR-10 数据集上实现了超过 90%的准确率，同时节省了推理时间并降低了能耗。随着 MobileNetV3 的成功开发和验证，使用此计算范式和开源框架实现更多基于忆阻器的神经网络的潜力显著增加。这一进展为未来部署计划奠定了开创性的路径。

***Index Terms*—忆阻器神经网络，MobileNetV3，神经形态计算。**

I. 介绍

机器学习的进步与其计算资源的可用性密不可分。随着机器学习，特别是深度学习模型努力提高其任务性能，对浮点运算日益增长的需求给计算资源带来了重大负担 [1] 。幸运的是，超大规模集成（VLSI）技术已经取得了显著的进步 [2], [3] 。具体来说，图形处理单元（GPU）的应用使得在大规模数据集上训练深度神经网络（DNNs）成为可能，使研究人员能够探索更复杂的算法来增强其模型 [4] 。然而，目前深度学习模型正在以前所未有的速度向计算能力的极限扩展 [5] ，特别是在边缘设备中，对能耗和推理延迟有严格要求 [6]–[9] 。最近，一些轻量级网络如 MobileNet [10], ShuffleNet [11] 和 SqueezeNet [12] 被提出用于在保持高精度的同时压缩模型大小。然而，即使是这些轻量级网络，参数数量通常也达到数百万级别。由于成本过高和能耗较大的特性，在边缘推理时利用高性能 GPU 是不切实际的 [13]。此外，基于冯 · 诺依曼架构的其他传统硬件加速器也无法从根本上解决由内存与计算单元分离导致的数据移动广泛所引起的高能耗和延迟问题 [14]–[17]，特别是在 DNN 推理过程中 [18]。作为一种新兴解决方案，基于忆阻器交叉开关的神经形态计算能够显著提升速度并降低功耗，为机器学习提供了一种新的计算范式 [19]。

MobileNetV3 是一种为在边缘设备上部署而优化的高效深度神经网络，旨在减少计算负载而不牺牲识别精度。探索如何利用神经形态计算来实现软硬件协同工作，从而最大限度地发挥 MobileNetV3 低功耗和快速推理速度的优势，是一个引人注目的研究课题。由于 MobileNetV3 的复杂架构和激活函数，这一领域的研究论文很少。在本研究中，我们介绍了一种基于忆阻器的 MobileNetV3 新计算范式和硬件设计。MobileNetV3 广泛使用向量矩阵乘法（VMM），这些可以在忆阻器交叉开关阵列中的模拟域有效执行。在这个计算范式中，乘法操作通过欧姆定律来实现，求和则是通过基尔霍夫电流定律 [20] 实现的。传感器输出的电压直接应用于忆阻器交叉开关阵列的行。经过训练的权重作为忆阻器的电导值存储起来。数组中每列产生的电流构成输出向量，然后经由跨阻放大器（TIAs）转换为电压信号以供下一层输入使用。非线性激活函数通过互补金属氧化物半导体（CMOS）电路实现。基于忆阻器的 MobileNetV3 网络包含几个关键神经模块：基于忆阻器的卷积模块、基于忆阻器的批归一化模块、激活函数模块、基于忆阻器的全局平均池化模块以及基于忆阻器的全连接模块。它

*Corresponding Author

还包括用于残差连接 [21] 的加法模块和注意力模块中的乘法模块 [10]。这些组件协同工作，使用模拟信号处理 MobileNetV3 的所有计算。在这种方法中，无需在计算单元和内存之间移动数据，从而减少能耗。此外，基于忆阻器的 VMM 由于其固有的并行性 [22]，作为深度神经网络推理的加速器发挥作用。

为了验证这种计算范式和 MobileNetV3 的硬件实现，我们还提供了一个高层次综合框架，该框架将用高级编程语言编写的程序转换为硬件描述语言 [23]，快速构建基于 SPICE 的网表文件。研究人员只需要提供输入数据和权重文件，该框架就能根据网络结构自动生成可靠的忆阻器电路实现。我们使用这个框架在 CIFAR-10 数据集上进行了图像分类测试。结果显示，我们的方法在准确性方面超过了其他最先进的忆阻器实现，并且在时间和能耗方面也优于传统的中央处理单元（CPU）和 GPU 方法。总之，本文做出了以下主要贡献：

- 基于忆阻器，首次提出了一种新的计算范式和 MobileNetV3 的硬件设计。实验表明，所提出的基于忆阻器的计算范式在 CIFAR-10 数据集上实现了 $> 90\%$ 的准确率，并且在计算资源、计算时间和功耗方面具有优势。值得注意的是，该解决方案实现了显著的加速，与传统的 GPU 实现相比速度提高了 138 倍，相对于常规 CPU 解决方案则提高了 2827 倍。在能效方面，它比 GPU 节省了 4.5 倍的能量，比 CPU 节省了 61.7 倍的能量。
- 四个节能模块电路，具有层归一化、硬 S 形、硬 Swish 和卷积功能，首先被设计为各种 DNNs 的重要组件。我们的设计理念优先考虑简洁性，在卷积层和全连接层中，我们与传统实现方法相比减少了 50%的运算放大器（op-amps）的数量 [24], [25]。这一策略旨在最大限度地发挥我们提出的计算范式的内在效率和节能优势。
- 此外，我们开发了一个开源的自动映射框架，以促进基于忆阻器的神经网络的快速构建和模拟。代码可在以下位置获取：https://github.com/JialeLiLab/MMobileNetV3。该框架使用户能够在几分钟内生成可靠的网表文件，而这一过程传统上需要几天的手动操作 [26]。该框架结合了灵活性与效率，为研究人员提供了一个强大的工具，以探索机器学习中的新忆阻器计算范式。这一开发不仅强调了我们计算范式的实用性，还为该领域的研究开辟了新的途径。

II. 背景及相关工作

随着机器学习中大规模深度学习模型的发展带来的计算需求激增，基于忆阻器的新计算范式正在引起越来越多的研究兴趣 [27]。忆阻器因其能够动态调整电阻和记忆电压或电流序列而著称 [28]。它在突触模拟、有效权重调整、低能耗和平行处理方面的能力使其成为新计算范式中的有前景的解决方案，特别是在神经网络和脑启发计算中 [29]。

在 1971 年，莱昂 · 丘教授根据电子理论的原则，预测了忆阻器的存在 [30] 。在 2008 年，惠普实验室通过基于二氧化钛创建忆阻器设备，首次实验验证了丘的假设 [31] 。最近，基于忆阻器的神经网络模块已经被开发出来 [32]–[34] ，展示了使用忆阻器的机器学习计算范式的可行性。然而，由于电路复杂性和电路级模拟耗时性的问题 [35] ，目前关于基于忆阻器的经典 DNN 设计研究较为稀少，并且主要集中在小数据集上的测试。具体来说，文等人设计了一个基于忆阻器的稀疏紧凑卷积神经网络，在 CIFAR-10 数据集上获得了 67.21%的准确率 [36] 。冉等人提出了一种基于忆阻器的 ShuffleNetV2，并在 FER2013 数据集上达到了 55.59%的准确率 [37] 。杨等人介绍了一种基于忆阻器的 Transformer 网络，并在 3 字符图像识别和 8

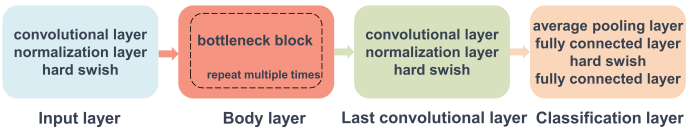

图 1. 基于 MobileNetV3 的网络架构流程图。

$$V_j = -\sum_{i=0}^{2N+1} \frac{V_i}{R_{i,j}} * R_f, i \in \mathbb{N}, j \in \mathbb{N} \tag{1}$$

MNIST 手写数字数据集上进行了测试 [38]。相比之下，我们的计算范式基于广泛使用的 DNN MobileNetV3，并在 CIFAR-10 数据集上实现了超过 90%的准确率。

III. 方法论

A. 网络架构

MobileNetV3 网络架构可以分为四个主要的忆阻器基础神经网络单元，其组成和连接关系如图 1 所示。

计算范式具有与 [39] 相同的神经网络层。输入层是对输入图像进行预处理的单元，包括卷积子层、归一化子层和硬 swish 激活子层。主体层包含几个瓶颈块，其中包括基于挤压和激励的轻量级注意力模块。最后一个卷积层负责从输入图像中提取高级特征并执行维度转换。它包括一个卷积子层、一个归一化子层和一个硬 swish 激活子层。最后一层是分类层，具有全局平均池化子层、两个全连接子层和一个硬 swish 激活子层。全连接层对于实现输入图像的分类至关重要。

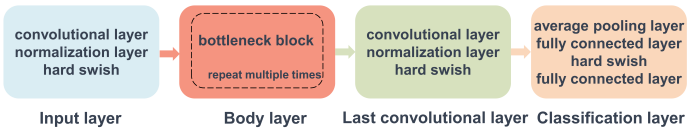

图 2. 基于 MobileNetV3 的网络架构流程图。

B. 忆阻器基卷积模块

在此计算范式中，存在三种卷积操作：常规卷积、深度可分离卷积和逐点卷积。这些卷积操作之间的差异如下：在常规卷积中，忆阻器被放置在交叉开关阵列的特定位置以执行滑动窗口操作，并且它们的输出电流相互连接以实现求和功能。相比之下，深度可分离卷积与常规卷积相比缺少这种求和操作 [40]。逐点卷积类似于单通道常规卷积 [41]。基于上述分析，我们将专注于常规卷积作为案例研究来展示忆阻器交叉开关阵列的实现细节。

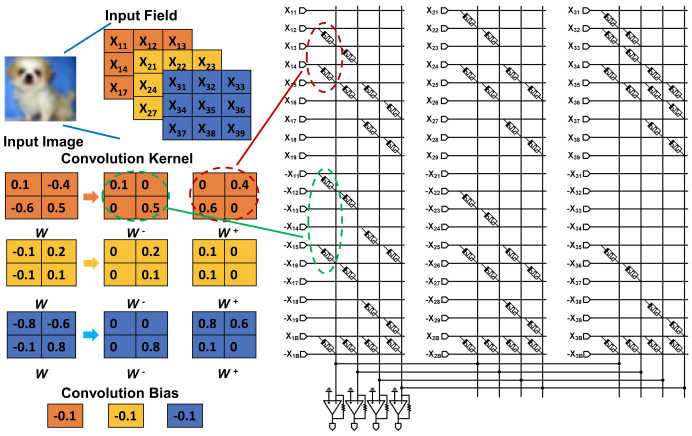

图 3. 用于卷积操作的忆阻器交叉棒电路图。

为了在基于忆阻器的卷积核和输入矩阵之间执行卷积操作，首先需要将原始权重矩阵分割为正权重矩阵和负权重矩阵，因为单个忆阻器的电阻是正值。如图 2 所示，与大多数研究论文中的传统方法 [32], [33], [42] 不同，我们将具有正权值的矩阵指定为负权重矩阵，并将其映射到包含反相输入部分的忆阻器交叉开关阵列中，而具有负权重的矩阵被标记为正权重矩阵，对应于忆阻器交叉开关阵列的原始输入。输出电流与实际结果极性相反。经过 TIA 后，由于 TIA 的反相性质，它被转换成与实际结果相同极性的电压。相较于上述方案，此方法最显著的优点是减少了每个输出端口的操作放大器数量。鉴于操作放大器的功耗在毫瓦级别而忆阻器的功耗则处于 μW 级别 [36]，这种方法确实具有意义。关于功耗的更详细分析将在实验部分讨论。

其次，输出矩阵的行数（O_r）和列数（O_c）由以下公式决定：

$$O_{r|c} = (\frac{W_{r|c} - F_{r|c} + 2P}{S} + 1) \tag{2}$$

其中，$O_{r|c}$、$W_{r|c}$ 和 $F_{r|c}$ 分别代表输出矩阵、输入矩阵和卷积核矩阵的行数或列数。P 和 S 分别代表填充和步幅。

第三，填充后的输入矩阵被用作新的输入矩阵，并按行展开作为正输入。随后，该新输入矩阵的否定形式被用作负输入。正输入区域和负输入区域内每个输出忆阻器的起始位置由以下方程确定：

$$P_{Pi} = (\lfloor \frac{i}{O_c} \rfloor * W_c + i \bmod O_c) * S, i \in \mathbb{N} \tag{3}$$

$$P_{Ni} = (\lfloor \frac{i}{O_c} \rfloor * W_c + i \bmod O_c) * S + W_r * W_c, i \in \mathbb{N} \tag{4}$$

其中 i 是输出索引，P_{Pi} 和 P_{Ni} 分别表示 O'_i 在正输入区域和负输入区域的起始位置。换句话说，这个公式表示（O'_i，P_{Pi}）是每列中放置第一个忆阻器的坐标。类似地，（O'_i，P_{Ni}）是每列负输入区域中放置第一个忆阻器的坐标。

第四，卷积核被展开为单行矩阵。从（O'_i,P_{Pi}）开始，依次将带有权重的 F_c 个忆阻器分配到忆阻器交叉阵列中，并以（W_c-F_c+2P）的间隔重复该过程。值得注意的是，具有零权重的忆阻器不会出现在忆阻器交叉阵列中，以此减少忆阻器的数量，因为它们对输出的贡献为零。然后，从（O'_i,P_{Ni}）开始，根据上述规则分配负输入区域的忆阻器。这两个偏置电压作为最后的输入，与忆阻器结合，形成卷积操作的偏置。

最后，忆阻器交叉开关中的卷积运算公式由以下方程表示：

$$V_j = -\sum_{i=0}^{2N+1} \frac{V_i}{R_{i,j}} * R_f, i \in \mathbb{N}, j \in \mathbb{N} \tag{5}$$

其中 V_i 是忆阻器交叉开关中第 $(i)^{th}$ 输入电压。$R_{i,j}$ 表示位于 (i, j) 位置的忆阻器的电阻，如果存在的话。R_f 表示 TIA 中的反馈电阻。公式中的负号由 TIA 的特性决定。V_j 是 $(j)^{th}$ 输出电压，对应于通过展平卷积结果矩阵获得的一维矩阵中的 $(j)^{th}$ 元素。

图 2 说明了一个常规卷积的示例，其中输入图像被分为三个通道，每个通道对应一个忆阻器交叉阵列。对于常规卷积，来自相同输出端口的电流被聚合，然后通过 TIA。相比之下，在深度卷积和点卷积中，每个输出端口独立连接到 TIA。以第一个通道对应的忆阻器阵列为例来说明忆阻器的布局。根据公式 1，忆阻器交叉阵列由 20 个输入（一个 3×3 正矩阵、一个 3×3 负矩阵和 2 个偏置）和 4 个输出（一个 2×2 矩阵）组成。此外，由于步长设置为一且填充设置为零，在此示例中，O_c、W_c 和 F_c 分别为二、三和二。忆阻器对于每个卷积输出的初始位置可以根据公式 2 确定（O'_0 对应 1，O'_1 对应 2，O'_2 对应 4，O'_3 对应 5）。然后，在忆阻器交叉条中按顺序将卷积核第一行元素 (0, 0.4) 的权重对应的忆阻器放置在起始位置。留出一个空位（根据 W_c-F_c+2P 的规则），继续放置具有第二行元素 (0.6, 0) 权重的两个忆阻器。由于有两个权重为零的忆阻器，因此在正输入区域只放置了两个分别带有 0.4 和 0.6 权重的忆阻器。膜忆阻器在每个卷积输出的负输入区域的初始位置可以根据公式 3 确定（对于 O'_0 为 9，对于 O'_1 为 10，对于 O'_2 为 12，对于 O'_3 为 13）。然后，在忆阻器交叉开关的起始位置依次放置卷积核第一行元素（0.1, 0）权重的忆阻器。接着，根据 W_c-F_c+2P 留出一个空位后，继续放置两个第二行元素（0,0.5）权重的忆阻器。与之前的分析一致，在忆阻器交叉开关的负输入区域仅放置了两个忆阻器。由于在这个例子中卷积偏置为负值，因此忆阻器将与正偏置电压结合形成偏置。在基于忆阻器的常规卷积层设计中，所需的忆阻器和运算放大器的数量分别由以下公式给出：

$$N_{cm} = O_c * O_r * (F_r * F_c * C_i + 1) * C_o \tag{6}$$

$$N_{co} = O_c * O_r * C_o \tag{7}$$

在这两个方程中，C_i 和 C_o 分别代表输入通道数和输出通道数，而 N_{cm} 和 N_{co} 表示卷积层中的忆阻器数量和运算放大器数量。

C. 基于忆阻器的批量归一化模块

批量归一化模块在 MobileNetV3 网络架构中被广泛使用，以实现卓越的准确性和效率。计算方法如下：

$$y = \frac{x - E[x]}{\sqrt{Var[x] + \epsilon}} * \gamma + \beta \tag{8}$$

其中 x 代表原始特征值，$E[x]$ 是均值，$Var[x]$ 是方差，epsilon 是一个小常量，用于防止除零错误（参见 [43]）。γ 和 β 分别是可学习的缩放和平移参数。y 是批归一化模块的输出。由于忆阻器交叉开关阵列适用于乘法和加法计算，批归一化的公式被分割成三个部分：减法、乘法和加法操作。转换后的公式如下：

$$y = (x - E[x]) * \left| \frac{\gamma}{\sqrt{Var[x] + \epsilon}} \right| + \beta, \gamma \geq 0 \tag{9}$$

$$y = (E[x] - x) * \left| \frac{\gamma}{\sqrt{Var[x] + \epsilon}} \right| + \beta, \gamma < 0 \tag{10}$$

在此电路设计中，使用了忆阻器和跨阻放大器来执行批归一化操作。具体来说，减法部分包括四个输入端口和两个忆阻器，输出通过跨阻放大器转换为电压以供后续的乘法操作使用。加法则类似于卷积操作中的偏置实现。为了最小化在设计批归一化模块时对运算放大器的使用，遵循了一组特定规则。V_b 被设置为一个常数值 1。如图 3(a) 所示，当输入值 γ 为非负时，第一组忆阻器的电阻值依次设置为 (1, 0, 0, 1)，便于执行减法操作 $(x - E[x])$。第二组忆阻器的电阻值取决于 β 的符号。对于正的 β，值被设置为 $(\left|\frac{\gamma}{\sqrt{Var[x]+\epsilon}}\right|, 0, \frac{1}{\beta})$；对于负的 β，它们被设置为 $(\left|\frac{\gamma}{\sqrt{Var[x]+\epsilon}}\right|, \frac{1}{\beta}, 0)$。如图 3(b) 所示，在 γ 为负的情况下，第一组忆阻器被设置为 (0, 1, 1, 0) 以实现减法操作 $(E[x] - x)$，而第二组则遵循与非负情况相同的电阻值分配。这种方法有效地减少了运算放大器的需求，并提升了批量归一化模块的整体效率。在基于忆阻器的批量归一化模块设计中，所需的忆阻器和运算放大器的数量分别由以下公式给出：

$$N_{bm} = 4 * C \tag{11}$$

$$N_{bo} = 2 * C \tag{12}$$

其中，C 表示通道数，而 N_{bm} 和 N_{bo} 则代表批量归一化模块中的忆阻器和运算放大器的数量。

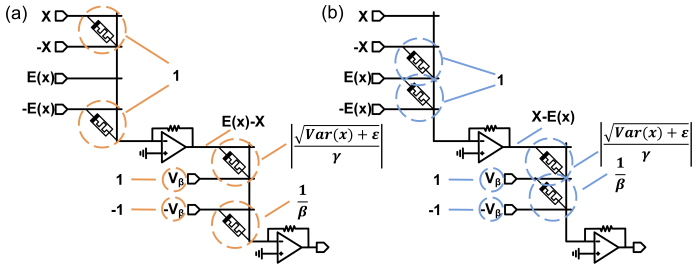

图 4. 忆阻器交叉开关电路用于批量归一化操作的原理图。(a)$\gamma \geq 0, \beta > 0$(b)$\gamma < 0, \beta < 0$。

D. 激活函数模块

在 MobileNetV3 中，使用了三种激活函数：ReLU、硬 Sigmoid 和硬 Swish。本文中的 ReLU 实现基于之前发表的论文 [44]。此外，我们首次设计了硬 Sigmoid 和硬 Swish 激活函数电路，如图 4(a) 和 4(b) 所示。在电路实现中，运算放大器被用来执行加法和除法操作。限幅器由一个二极管和一个电源构成，在执行最大化功能方面起着关键作用。与硬 Sigmoid 激活函数模块相比，硬 Swish 模块有一个额外的乘法操作，包括一个乘法器。根据图 4(c) 和 4(d) 所示的仿真结果，可以看出该模块实现了与软件设计一致的功能目标。

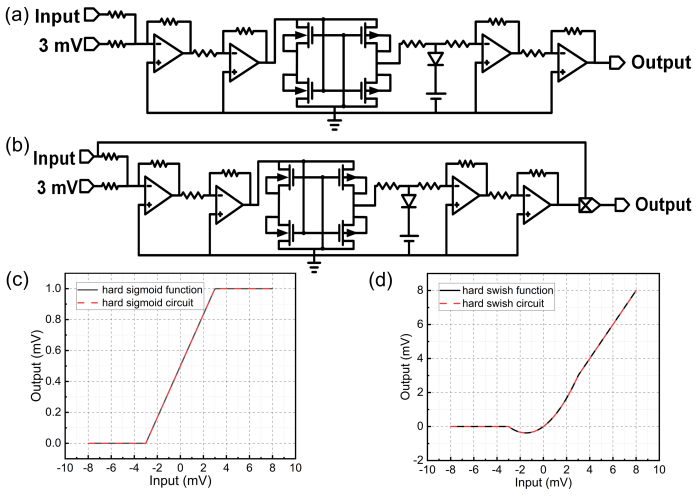

图 5. 激活函数操作的电路图。(a) 硬 Sigmoid，(b) 硬 Swish，以及仿真结果 (c) 硬 Sigmoid，(d) 硬 Swish。

E. 基于忆阻器的全局平均池化模块

在 MobileNetV3 中，全局平均池化位于网络末端，将深层卷积特征图转换为一维特征向量。这个生成的一维特征向量随后被用于分类任务。图 5(a) 展示了在忆阻器交叉开关阵列上执行全局平均池化计算的过程。输入矩阵的逆展平成一个一维向量，并作为电压施加到忆阻器交叉开关阵列的输入端。权重值设置等于输入矩阵的数量。根据欧姆定律和基尔霍夫电路定律，每个输入除以总输入数量后相加，导致负全局平均电流。这个负全局平均电流通过 TIAs 转换为正全局平均电压，作为输出结果。在基于忆阻器的全局平均池化模块设计中，所需的忆阻器和运算放大器的数量分别由以下公式给出：

$$N_{gm} = W_c * W_r * C \tag{13}$$

$$N_{go} = C \tag{14}$$

其中 N_{gm} 和 N_{go} 分别表示全局平均池化模块中的忆阻器和运算放大器数量。

F. 基于忆阻器的全连接模块

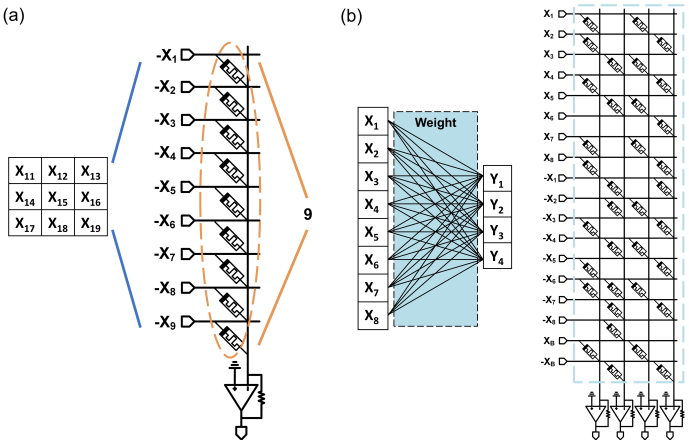

图 6. 电路示意图展示了用于（a）全局平均池化操作和（b）全连接操作的忆阻器交叉开关。

基于忆阻器的全连接模块整合并归一化了多次卷积后高度抽象化的特征，并为识别网络中的每个类别输出一个概率。基于忆阻器的全连接模块电路与基于忆阻器的卷积模块电路类似，但通常尺寸更大且包含更多的忆阻器。与卷积模块相比，全连接模块中忆阻器的排列规则相对简单。只需要将原始权重矩阵转换为正负权重矩阵，然后按垂直顺序排列即可。在基于忆阻器的全连接模块设计中，所需忆阻器和运算放大器的数量分别由以下公式给出：

$$N_{fm} = (W + 1) * O \tag{15}$$

$$N_{fo} = O \tag{16}$$

其中 W 是输入数量。O 表示输出数量。N_{fm} 和 N_{fo} 分别表示全连接模块中的忆阻器和运算放大器的数量。

G. 自动化框架

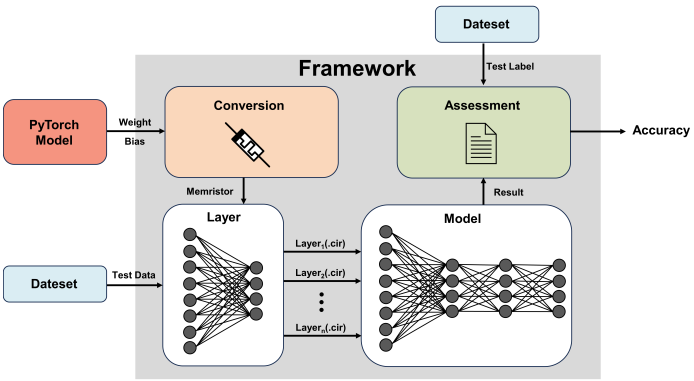

图 7. 自动化框架的框图。

尽管基于忆阻器的计算范式在执行神经拟态计算时表现出高能效，为解决机器学习中计算可行性的限制提供了有前景的解决方案，但随着神经网络层数的增加，电路结构的复杂性和所需的广泛仿真分析仍然阻碍了这一领域的进展 [45]。为了应对上述问题并验证我们提出的计算范式，本文提出了一种用于快速构建基于忆阻器的 MobileNetV3 神经网络的自动化映射框架。图 6 展示了该自动框架的模块图，该框架根据网络拓扑和特定超参数生成忆阻器电路的 SPICE 网表。该框架由四个主要模块组成：转换模块、层模块、模型模块和评估模块。转换模块负责将用 PyTorch 训练得到的权重和偏置转化为忆阻器所需电阻格式。在本文中，使用了 HP 模型来构建 MobileNetV3 神经网络，并遵循以下公式 [42]：

$$R_M = R_{on}w + R_{off}(1-w) \tag{17}$$

其中 R_M、R_{on} 和 R_{off} 分别表示忆阻器、掺杂层和未掺杂层的电阻。w 是掺杂层的归一化宽度。模型中的权重被用作 $\frac{1}{R_M}$ 来使用公式 16 计算忆阻器所需的参数 w。层模块根据上一节提到的规则为不同层次生成基于忆阻器的电路的 SPICE 电路网表文件。在生成基于忆阻器的神经网络后，该框架通过运行 SPICE 模拟自动对用户提供的数据集和权重文件进行图像分类任务。该框架促进了具有二级延迟的网表文件的创建，从而为研究人员将 DNN 映射到忆阻器交叉阵列开辟了途径。能够在一个非常短的时间内生成大规模忆阻器交叉阵列的网表文件是由于将忆阻器的布局设计规则抽象成算法的结果。首先，对于基于忆阻器的卷积模块，我们使用多个循环来实现布局方法，这允许高效地组织交叉阵列中的忆阻器。接下来，在基于忆阻器的批量归一化模块中，根据 γ 和 β 的符号条件语句确定忆阻器的布局。然后，基于忆阻器的全局平均池化模块使用输入的数量作为决定忆阻器电阻值的关键因素。最后，对于基于忆阻器的全连接模块，我们在忆阻器交叉阵列中按垂直顺序排列正负权重矩阵。

表 I
与其他基于新计算范式的神经网络的性能比较

Publication/Year	Device	Signal	Accuracy
DATE'18 [46]	RRAM	Digital	86.08%
TNSE'19 [36]	memristor	Analog	67.21%
TNNLS'20 [37]	memristor	Analog	84.38%
ISSCC' 21 [47]	eDRAM	Analog	80.1%
TCASII' 23 [48]	RRAM	Digital	86.2%
TCASII' 23 [49]	memristor	Analog	87.5%
This work	memristor	Analog	**90.36%**

IV. 实验与结果

A. 准确性

在本节中，我们使用开发的框架实现了一个缩小版的 MobileNetV3 神经网络，用于 CIFAR-10 数据集的分类应用，该数据集包含 32×32 彩色图像。经过神经网络计算后，CIFAR-10 数据集被归类为十个不同的标签。由忆阻器基础神经网络决定的分类结果被识别为电流最高的输出。网络权重来自一个在 CIFAR-10 数据集上训练过的离线服务器。表 1 比较了我们的研究与其他使用新型计算范式的神经网络的性能。我们基于忆阻器的 MobileNetV3 神经网络在 CIFAR-10 数据集上实现了 90.36%的高分类准确率，与使用 PyTorch 框架的传统实现相当，并且在此领域之前的成果中表现出显著提升。

B. 延迟

我们参考了论文中先前报告的方法来分析基于忆阻器神经网络的延迟和能耗 [19], [36]–[38]，以确定我们提出的计算范式的延迟。基于忆阻器的神经电路中，响应时间可以快达 100 ps [37]。然而，基于忆阻器的电路

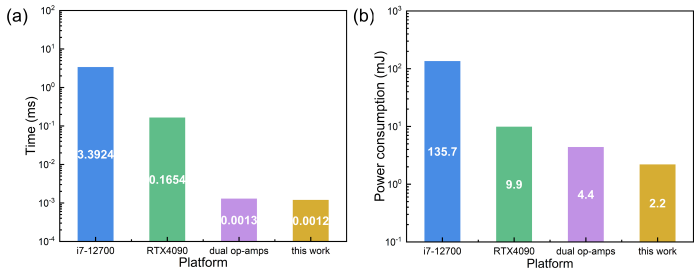

图 8. (a) 存储器电阻器（memristor）基础的 MobileNetV3 神经网络的延迟和 (b) 功耗。

的整体速度受到用于将电流转换为电压的运算放大器的压摆率的限制。因此，该网络的延迟遵循以下公式：

$$T_i = (T_m + T_o) * N_m + T_r \tag{18}$$

其中 T_m 表示忆阻器交叉开关阵列的响应时间，T_o 是运算放大器的转换时间，N_m 表示基于忆阻器的层数，T_r 表示其他层的延迟，T_i 表示推理过程的延迟。低功耗运算放大器的典型转换速率约为 10 V/μs 水平 [50]。T_r 包括激活函数层、加法器和乘法器等模块相关的延迟。我们使用现有设备的信息分析这些层的延时。因此，该电路的延时可低至 1.24 μs，相比之下，传统双运算放大器解决方案的延时为 1.30 μs。这比传统的 GPU 方法快得多，后者在配备 RTX 4090 服务器上的延时为 0.1654 ms。如图 7(a) 所示，我们在 CPU（i7-12700）上对单图像处理进行了推理时间测试，结果延时为 3.3924 ms。总之，这一计算范式实现了卓越的加速，其速度比传统的 GPU 实现快超过 138 倍，并且相对于传统基于 CPU 的解决方案来说，则超过了 2827 倍。

C. 能耗

图 8 展示了不同层中忆阻器权重的分布情况。如图 8 所示，忆阻器交叉条的权重主要分布在-0.2 和 0.2 之间。基于忆阻器的神经网络的功耗源自忆阻器的权重。因为我们已将输入数据映射到 ±2.5 mV，所以可以通过假设所有输入数据为 2.5 mV 且所有权重均为 0.2 来估计单个忆阻器的最大预测功耗。因此，忆阻器的最大功耗可以估计为 1.1μ 瓦。因此，我们使用以下公式来估算能耗：

$$W_i = \sum U_{max}^2 G_{max} * T_m + P_o * T_o + Pr * T_r \tag{19}$$

其中 U_{max} 表示单个忆阻器上的最大电压，G_{max} 指忆阻器的等效电导率，P_o 代表运算放大器的功耗，P_r 表示其他层使用的功率，而 W_i 则表示总功耗。根据设备的标准功耗，我们估计神经网络单次完整的前向推理消耗 2.2 毫焦耳。使用 CPU、GPU 和双运算放大器的实现比较结果如图 7(b) 所示。如图所示，与 GPU 实现相比节能 4.5 倍，与 CPU 实现相比节能 61.7 倍。

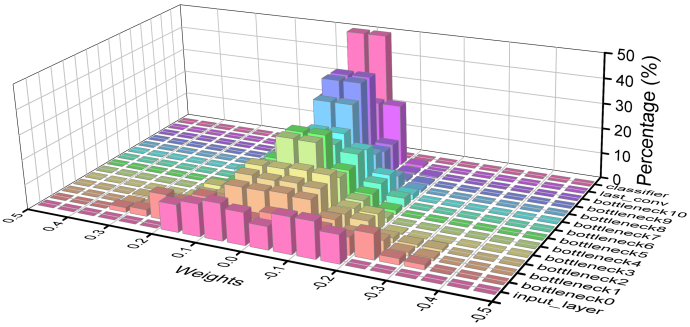

图 9. 忆阻器权重的分布。

V. 结论

本文提出的基于忆阻器的 MobileNetV3 在准确率、延迟和能耗方面表现出色，突显了这一计算范式作为解决当前机器学习中面临的计算资源限制问题的有前景解决方案的潜力。此外，展示了一个用于构建和验证基于忆阻器计算范式的自动化框架，有望加速该领域的研究。我们的结果表明，我们提出的方法不仅在与传统的 CPU 和 GPU 实现相比具有优势，而且也优于其他新型计算范式，从而为将基于忆阻器的计算范式应用于机器学习领域铺平了道路。

参考文献

[1] H. A. Gonzalez, J. Huang, F. Kelber, K. K. Nazeer, T. Langer, C. Liu, M. Lohrmann, A. Rostami, M. Schöne, B. Vogginger *et al.*, "Spinnaker2: A large-scale neuromorphic system for event-based and asynchronous machine learning," *arXiv preprint arXiv:2401.04491*, 2024.

[2] J. Lu, W.-K. Chow, C.-W. Sham, and E. F. Y. Young, "A dual-mst approach for clock network synthesis," *2010 15th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 467–473, 2010.

[3] J. Lu, W.-K. Chow, and C.-W. Sham, "A new clock network synthesizer for modern vlsi designs," *Integr.*, vol. 45, pp. 121–131, 2012.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[5] S. Woźniak, A. Pantazi, T. Bohnstingl, and E. Eleftheriou, "Deep learning incorporating biologically inspired neural dynamics and in-memory computing," *Nature Machine Intelligence*, vol. 2, no. 6, pp. 325–336, 2020.

[6] C. Y. Lo, C.-W. Sham, and L. Ma, "A novel iris verification framework using machine learning algorithm on embedded systems," *2020 IEEE 9th Global Conference on Consumer Electronics (GCCE)*, pp. 173–175, 2020.

[7] Z. Yue, D. Yan, R. Wu, L. Ma, and C.-W. Sham, "Mtst: A multi-task scheduling transformer accelerator for edge computing," *2024 IEEE 13th Global Conference on Consumer Electronics (GCCE)*, pp. 1394–1395, 2024.

[8] Y. Fu, D. Yan, J. Li, S. L. Ma, C.-W. Sham, and H.-f. Chou, "An efficient fpga-based edge ai system for railway fault detection," *IEEE Consumer Electronics Magazine*, 2025.

[9] J. Li, Y. Fu, D. Yan, S. L. Ma, and C.-W. Sham, "An edge ai system based on fpga platform for railway fault detection," in *2024 IEEE 13th Global Conference on Consumer Electronics (GCCE)*. IEEE, 2024, pp. 1387–1389.

[10] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan *et al.*, "Searching for mobilenetv3," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1314–1324.

[11] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "Shufflenet v2: Practical guidelines for efficient cnn architecture design," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 116–131.

[12] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size," *arXiv preprint arXiv:1602.07360*, 2016.

[13] R. Desislavov, F. Martínez-Plumed, and J. Hernández-Orallo, "Trends in ai inference energy consumption: Beyond the performance-vs-parameter laws of deep learning," *Sustainable Computing: Informatics and Systems*, vol. 38, p. 100857, 2023.

[14] C. Y. Lo and C.-W. Sham, "Energy efficient fixed-point inference system of convolutional neural network," *2020 IEEE 63rd International Midwest Symposium on Circuits and Systems (MWSCAS)*, pp. 403–406, 2020.

[15] C. Y. Lo, F. C.-M. Lau, and C.-W. Sham, "Fixed-point implementation of convolutional neural networks for image classification," *2018 International Conference on Advanced Technologies for Communications (ATC)*, pp. 105–109, 2018.

[16] C. Y. Lo, C.-W. Sham, and C. Fu, "Novel cnn accelerator design with dual benes network architecture," *IEEE Access*, vol. 11, pp. 59 524–59 529, 2023.

[17] F. Yan, A. Koch, and O. Sinnen, "A survey on fpga-based accelerator for ml models," *arXiv preprint arXiv:2412.15666*, 2024.

[18] O. Mutlu, S. Ghose, J. Gómez-Luna, and R. Ausavarungnirun, "A modern primer on processing in memory," in *Emerging Computing: From Devices to Systems: Looking Beyond Moore and Von Neumann*. Springer, 2022, pp. 171–243.

[19] W. Zhang, P. Yao, B. Gao, Q. Liu, D. Wu, Q. Zhang, Y. Li, Q. Qin, J. Li, Z. Zhu *et al.*, "Edge learning using a fully integrated neuro-inspired memristor chip," *Science*, vol. 381, no. 6663, pp. 1205–1211, 2023.

[20] C. Li, M. Hu, Y. Li, H. Jiang, N. Ge, E. Montgomery, J. Zhang, W. Song, N. Dávila, C. E. Graves *et al.*, "Analogue signal and image processing with large memristor crossbars," *Nature electronics*, vol. 1, no. 1, pp. 52–59, 2018.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[22] C. Wang, G.-J. Ruan, Z.-Z. Yang, X.-J. Yangdong, Y. Li, L. Wu, Y. Ge, Y. Zhao, C. Pan, W. Wei *et al.*, "Parallel in-memory wireless computing," *Nature Electronics*, pp. 1–9, 2023.

[23] P. Coussy, D. D. Gajski, M. Meredith, and A. Takach, "An introduction to high-level synthesis," *IEEE Design & Test of Computers*, vol. 26, no. 4, pp. 8–17, 2009.

[24] B. Li and G. Shi, "A cmos rectified linear unit operating in weak inversion for memristive neuromorphic circuits," *Integration*, vol. 87, pp. 24–28, 2022.

[25] Y. Zhang, M. Cui, L. Shen, and Z. Zeng, "Memristive quantized neural networks: A novel approach to accelerate deep learning on-chip," *IEEE transactions on cybernetics*, vol. 51, no. 4, pp. 1875–1887, 2019.

[26] D. Nikishov and A. Antonov, "Automated generation of spice models of memristor-based neural networks from python models," in *2023 International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM)*. IEEE, 2023, pp. 895–899.

[27] P. Yao, H. Wu, B. Gao, J. Tang, Q. Zhang, W. Zhang, J. J. Yang, and H. Qian, "Fully hardware-implemented memristor convolutional neural network," *Nature*, vol. 577, no. 7792, pp. 641–646, 2020.

[28] Y. N. Joglekar and S. J. Wolf, "The elusive memristor: properties of basic electrical circuits," *European Journal of physics*, vol. 30, no. 4, p. 661, 2009.

[29] Q. Xia and J. J. Yang, "Memristive crossbar arrays for brain-inspired computing," *Nature materials*, vol. 18, no. 4, pp. 309–323, 2019.

[30] L. Chua, "Memristor-the missing circuit element," *IEEE Transactions on circuit theory*, vol. 18, no. 5, pp. 507–519, 1971.

[31] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, "The missing memristor found," *nature*, vol. 453, no. 7191, pp. 80–83, 2008.

[32] C. Yakopcic, M. Z. Alom, and T. M. Taha, "Memristor crossbar deep network implementation based on a convolutional neural network," in *2016 International joint conference on neural networks (IJCNN)*. IEEE, 2016, pp. 963–970.

[33] ——, "Extremely parallel memristor crossbar architecture for convolutional neural network implementation," in *2017 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2017, pp. 1696–1703.

[34] S. Wen, J. Chen, Y. Wu, Z. Yan, Y. Cao, Y. Yang, and T. Huang, "Ckfo: Convolution kernel first operated algorithm with applications in memristor-based convolutional neural network," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 40, no. 8, pp. 1640–1647, 2020.

[35] L. Xia, B. Li, T. Tang, P. Gu, P.-Y. Chen, S. Yu, Y. Cao, Y. Wang, Y. Xie, and H. Yang, "Mnsim: Simulation platform for memristor-based neuromorphic computing system," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 5, pp. 1009–1022, 2017.

[36] S. Wen, H. Wei, Z. Yan, Z. Guo, Y. Yang, T. Huang, and Y. Chen, "Memristor-based design of sparse compact convolutional neural network," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 3, pp. 1431–1440, 2019.

[37] H. Ran, S. Wen, S. Wang, Y. Cao, P. Zhou, and T. Huang, "Memristor-based edge computing of shufflenetv2 for image classification," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 40, no. 8, pp. 1701–1710, 2020.

[38] C. Yang, X. Wang, and Z. Zeng, "Full-circuit implementation of transformer network based on memristor," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 69, no. 4, pp. 1395–1407, 2022.

[39] R. Chen, W. Zeng, W. Fan, F. Lai, Y. Chen, X. Lin, L. Tang, W. Ouyang, Z. Liu, and X. Luo, "Automatic recognition of ocular surface diseases on smartphone images using densely connected convolutional networks," in *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*. IEEE, 2021, pp. 2786–2789.

[40] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.

[41] B.-S. Hua, M.-K. Tran, and S.-K. Yeung, "Pointwise convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 984–993.

[42] B. Li and G. Shi, "A native spice implementation of memristor models for simulation of neuromorphic analog signal processing circuits," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 27, no. 1, pp. 1–24, 2021.

[43] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. pmlr, 2015, pp. 448–456.

[44] P. Priyanka, G. Nisarga, and S. Raghuram, "Cmos implementations of rectified linear activation function," in *VLSI Design and Test: 22nd International Symposium, VDAT 2018, Madurai, India, June 28-30, 2018, Revised Selected Papers 22*. Springer, 2019, pp. 121–129.

[45] J. Li, Y. Fu, C.-W. Sham, and S. L. Ma, "A framework for mapping convolutional neural network onto memristor crossbars," in *2024 IEEE 13th Global Conference on Consumer Electronics (GCCE)*. IEEE, 2024, pp. 1390–1393.

[46] X. Sun, S. Yin, X. Peng, R. Liu, J.-s. Seo, and S. Yu, "Xnor-rram: A scalable and parallel resistive synaptic architecture for binary neural networks," in *2018 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2018, pp. 1423–1428.

[47] S. Xie, C. Ni, A. Sayal, P. Jain, F. Hamzaoglu, and J. P. Kulkarni, "16.2 edram-cim: Compute-in-memory design with reconfigurable embedded-dynamic-memory array realizing adaptive data converters and charge-domain computing," in *2021 IEEE International Solid-State Circuits Conference (ISSCC)*, vol. 64. IEEE, 2021, pp. 248–250.

[48] Y. Li, J. Chen, L. Wang, W. Zhang, Z. Guo, J. Wang, Y. Han, Z. Li, F. Wang, C. Dou *et al.*, "An adc-less rram-based computing-in-memory macro with binary cnn for efficient edge ai," *IEEE Transactions on Circuits and Systems II: Express Briefs*, 2023.

[49] H. Xiao, X. Hu, T. Gao, Y. Zhou, S. Duan, and Y. Chen, "Efficient low-bit neural network with memristor-based reconfigurable circuits," *IEEE Transactions on Circuits and Systems II: Express Briefs*, 2023.

[50] B. Hosticka, R. Klinke, and H.-J. Pfleiderer, "A very high slew-rate cmos operational amplifier," 1989.