通过故障感知阈值电压优化提高脉冲神经网络的可靠性

Ayesha Siddique, Khaza AnuarulHoque密苏里大学
电气工程与计算机科学系，美国密苏里州哥伦比亚
ayesha.siddique@mail.missouri.edu, hoquek@missouri.edu

arXiv:2301.05266v1 [cs.NE] 12 Jan 2023

摘要—**脉冲神经网络借助神经形态硬件，在计算机视觉领域取得了突破。然而，神经形态硬件缺乏并行性，因此限制了边缘设备上脉冲神经网络的吞吐量和硬件加速能力。为解决这一问题，近期提出了许多脉动阵列脉冲神经网络加速器（脉动式脉冲神经网络），但其可靠性仍是一个主要问题。本文首先深入分析永久性故障对脉动式脉冲神经网络的影响。随后，我们提出一种新颖的故障缓解方法，即在重新训练过程中进行故障感知阈值电压优化（FalVolt）。FalVolt 在重新训练时优化每一层的阈值电压，使网络在存在故障的情况下仍能达到接近基准的分类准确率。为展示所提缓解方法的有效性，我们在带有固定型故障的 256×256 脉动式脉冲神经网络上，对静态数据集（即 MNIST）和神经形态数据集（即 N-MNIST 和 DVS手势数据集）进行分类。实验表明，即使故障率极低（低至 0.012%），脉动式脉冲神经网络的分类准确率也会显著下降。我们提出的 FalVolt 缓解方法使脉动式脉冲神经网络能够在故障率高达 60% 的情况下运行，同时分类准确率仅有微不足道的下降（低至 0.1%），从而提升其性能。结果表明，与人工神经网络（ANN）中常见的其他先进技术相比，例如不进行阈值电压优化的故障感知剪枝与再训练，FalVolt 的速度快 2 倍。**

索引词—**脉冲神经网络、固定型故障、脉动阵列、故障缓解措施。**

I. 引言

脉冲神经网络（SNN）是一种前景广阔的第三代神经网络，能够以低功耗实现高算法性能。其硬件加速需要专用架构，例如 SpiNNaker [1], 和 TrueNorth [2]。然而，这些架构的每个核心都缺乏并行性，也缺少能够最大限度复用权重数据的高效数据流。这限制了它们在资源受限设备（如电池驱动的自动驾驶汽车）中的可实现吞吐量和鲁棒性。为此，在脉动阵列等大规模并行硬件加速器上利用脉冲神经网络，已被证明是一种高效的解决方案 [3], [4], [5], [6], [7]。脉动阵列脉冲神经网络加速器（systolicSNN）借鉴了其他先进的硬件加速器 [8]，这类加速器支持人工神经网络（ANN）的完全并行执行。这些加速器采用 NxN 的互连处理单元（PE）密集网格，能够凭借较高的时空局部性实现高效并行处理。与人工神经网络不同，脉冲神经网络及其硬件加速器仍处于相对早期的应用阶段 [9] ，因此，确保脉动式脉冲神经网络的可靠性仍是一项重大的研究挑战。

脉动式脉冲神经网络硬件芯片采用纳米级互补金属氧化物半导体技术制造 [10], ，这需要高度复杂的制造工艺。该工艺中的缺陷会导致各种制造缺陷，从工艺偏差到固定型故障等永久性故障。固定型故障会在每个执行周期影响脉动式脉冲神经网络的输出，因此会造成显著的准确率损失，正如本文所述。此外，脉冲神经网络中突触失效故障等大规模故障的影响已得到充分研究 [11], [12]。然而，要在硬件中分析此类故障，需要采用更高抽象层次的故障模型，以便对仿真进行追踪。Guo 等人利用突触固定型故障模型，研究了采用不同编码方案训练的脉冲神经网络的故障韧性 [13]。El-Sayed 等人分析了这些故障对漏积分发放（LIF）神经元晶体管级设计的影响 [14]。其他先进研究则关注权重存储器中的位翻转 [15], [16], [17], [18]。相较之下，固定型故障对脉动式脉冲神经网络的影响尚未得到研究。

固定型故障通常通过制造后测试检测，以剔除有缺陷的制造芯片。然而，如果制造的芯片中有大量存在缺陷，剔除这些芯片会大幅降低良率。一种潜在的解决方案是采用冗余执行（重新执行）来确保输出正确，但这会导致显著的延迟和能耗开销 [17]。在当前资源受限的纳米级硬件范式中，为满足终端用户的鲁棒性要求，处理单元数量已大幅增加，因此亟须借助高效且容错的 systolicSNN 最大限度地提高良率。近期，Mehul 等人针对脉冲神经网络中固定为 0 的权重提出了一种星形胶质细胞自修复机制 [19]。其他研究要么专注于缓解脉冲神经网络中的瞬态故障 [16], [19]，要么探讨人工神经网络加速器中的永久性故障缓解 [20], [21], [22], [23]。然而，针对脉动式脉冲神经网络中永久性故障影响的缓解，仍存在相当大的研究空白。

创新贡献：本文对固定型故障的脆弱性进行了广泛分析，并提出了一种新颖的故障缓解方法，即故障感知的阈值电压优化再训练（*FalVolt*。FalVolt 首先将仅映射到故障处理单元的权重置零，然后重新训练映射到无故障处理单元的权重，同时优化每一层的阈值电压，使分类准确率恢复至接近基线水平。优化后的阈值电压不同于初始训练中使用的

实际阈值电压。为证明所提出的 FalVolt 缓解方法的有效性，我们使用了静态MNIST [24], 、神经形态N-MNIST [25]和 DVS128 手势 [26] 数据集。结果表明，FalVolt 在故障率高达 60% 时仍能运行，且对分类准确率的影响可忽略不计，与基线相比几乎没有下降。我们的实证结果表明，FalVolt 所需的重新训练轮次减少了 2 倍，因此与故障感知剪枝与再训练等其他先进技术相比，恢复基线准确率的速度提高了 2 倍。值得注意的是，故障感知剪枝与再训练以及阈值电压优化传统上用于人工神经网络故障缓解 [21], [22]和加快 SNN 收敛。据我们所知，这是首项采用故障感知阈值电压优化来缓解脉冲神经网络故障的研究。

本文其余部分安排如下：第 II 节介绍脉冲神经网络和 systolicSNNs 的预备知识。第 III 节和第 IV 节分别介绍一个用于说明问题的案例研究，以及为 systolicSNNs 提出的 FalVolt 缓解方法。第 V 节讨论故障脆弱性和缓解措施的结果。最后，第 VI 节总结全文。

II. 背景

本节简要概述当前先进的脉冲神经网络和 systolicSNNs，以便更好地理解相关内容。

脉冲神经网络：脉冲神经网络是一种仿生人工神经网络。其工作原理可通过标准的 LIF模型说明如下：当突触前神经元的膜电位V_t 在时间 t超过特定阈值电压时，便会发放一个突触后脉冲，随后 V_t 以时间常数 τ回落至静息状态（V_{rest} < 阈值电压）。 V_t 在静息状态下维持不应期 t_{ref} ，之后才对接收到的脉冲作出响应。基于LIF的脉冲神经网络能够学习突触前权重，但训练时需要手动调整时间常数。此外，通常所有神经元采用相同的时间常数，这限制了神经元的多样性，进而限制了基于LIF的脉冲神经网络的表达能力。最近，Fang 等人提出了一种先进的 LIF模型，即参数化漏积分发放（PLIF），可在训练权重的同时训练时间常数 [27]。通过基于PLIF的脉冲神经网络引入可学习时间常数，可降低网络对初始值的敏感性，并

可减少训练时间。

脉动阵列 SNN 加速器：脉动式脉冲神经网络利用空间和时间并行性，使二进制脉冲输入（逻辑 1 或 0）沿垂直方向在脉动阵列中传播。如图 1 所示，脉冲输入首先被划分为多个时间步，然后每个时间步的所有输入值都映射到脉动阵列的一行上。二进制脉冲输入以时钟同步的方式穿过由相互连接的处理单元组成的密集 NxN 网格。滤波器数据映射并预先存储在处理单元中。图 3a 展示了脉动式脉冲神经网络中标准处理单元的设计。该处理单元在使能信号作用下，根据 1 位二进制脉冲累加 32 位权重输入。脉动式脉冲神经网络中的累加操作所需的加法器，比脉动阵列人工神经网络加速器中乘加（MAC）单元所需的乘法器成本更低 。

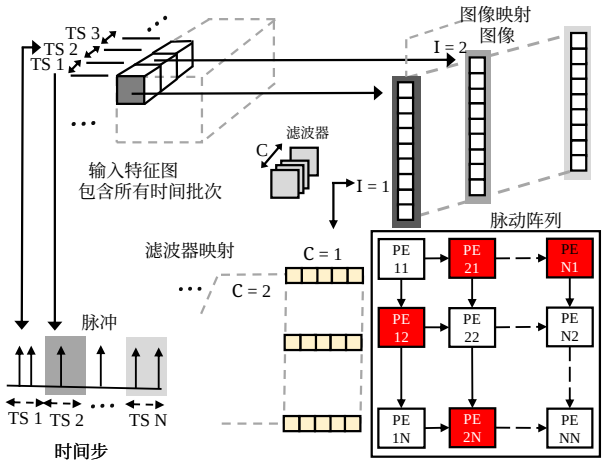
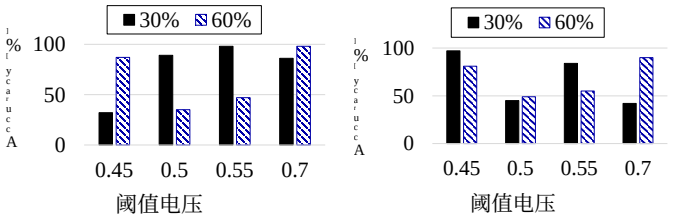


图 1：包含故障处理单元（PEs，以红色表示）和无故障处理单元（以白色表示）的 systolicSNN

脉动式脉冲神经网络不使用乘法器，因此与脉动阵列人工神经网络加速器相比能效更高。处理单元还采用加减选择单元来处理有符号权重。此外，内部计数器有助于统计推理阶段的脉冲数量。

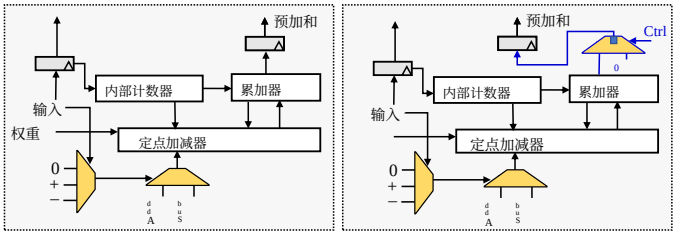
III. 动机案例研究

为说明所提出的 FalVolt 缓解方法，我们首先通过实证分析不同阈值电压对存在故障的 systolicSNN 分类准确率的影响。为此，我们先使用 MNIST 和 DVS128 Gesture 数据集训练一个 PLIF 脉冲神经网络。随后，在一个 256×256 脉动式脉冲神经网络中，针对 30% 和 60% 的处理单元，使用不同的故障图注入固定型故障。接着，我们并行运行采用不同阈值电压的重新训练仿真。如图 2 所示，我们观察到，在重新训练中将阈值电压从 1.0 调整为 0.55 和 0.7 后，即使 systolicSNN 中分别有 30% 和 60% 的处理单元存在故障，使用 MNIST 数据集时仍可达到 99% 的分类准确率。然而，当 30% 和 60% 的处理单元分别存在故障时，使用 0.45 和 0.5 的阈值电压重新训练同一模型，会分别导致约 73% 和 60% 的准确率损失。此外，对于处理单元中分别存在 30% 和 60% 故障的 systolicSNN，0.45 和 0.7 的阈值电压最适合对 DVS128 Gesture 数据集进行分类。然而，当 30% 和 60% 的处理单元分别存在故障时，使用 0.7 和 0.5 的阈值电压重新训练同一模型，会分别导致约 60% 和 55% 的准确率损失。因此，要使 systolicSNN 在重新训练后获得较高的分类准确率，选择合适的阈值电压至关重要。尽管如此，找到合适的阈值电压需要进行大量重新训练仿真，这可能耗费大量时间。受此启发，我们提出一种新颖的、用于故障缓解的重新训练故障感知阈值电压优化技术。



(a) MNIST分类　　(b) DVS128手势分类

图2：使用不同阈值电压进行固定型故障缓解（V_{th}）；在256x256 脉动阵列脉冲神经网络加速器（systolicSNN）中，30% 和 60% 的处理单元存在故障



(a) 实际处理单元　　(b) 旁路处理单元

图3：包含实际电路和旁路电路的处理单元 IV. 提出的故障感知阈值电压优化（FalVolt）

我们提出的 FalVolt 缓解方法通过首先将映射到故障处理单元的输入预训练权重设为零，来提高 systolicSNN 的可靠性。故障位置通过对 systolicSNN芯片进行制造后测试确定。如图 3b 所示，这一初始步骤类似于在硬件层使用多路复用器旁路处理单元。启用旁路路径后，故障处理单元对列和的贡献将被跳过。然而，由于 systolicSNN 在数据处理过程中会重复使用，旁路单个故障处理单元可能导致多个预训练权重被剪枝。因此，FalVolt 接下来会重新训练未剪枝权重，同时为每一层优化阈值电压。

阈值电压优化通过消除穷举搜索合适阈值电压的需要，节省了重新训练时间。它降低了脉冲神经网络对初始值的敏感性，并增强和加快了学习过程。优化后的阈值电压用于某一层中的所有神经元，以减少可再训练参数和所需时间。FalVolt 在初始训练和重新训练期间，均通过递归梯度计算来优化权重。每个再训练轮次结束时，映射到故障处理单元的权重都会被设为零。然而，如下文所述，仅在重新训练期间针对每一层优化阈值电压：

令 $\mathbf{r}$ 表示膜电位v 与阈值电压 $\mathbb{V}$的比值。当 v 超过 $\mathbb{V}$ 时，神经元会发出输出脉冲 $\mathbf{o}$。数学上可写为：

$$\mathbf{z}_l^t = \mathbf{r}_l^t - 1 \quad and \quad \mathbf{o}_l^t = \begin{cases} 1, & \text{if } \mathbf{z}_l^t > 0. \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

这里，符号$\mathbf{x}_l^t$ 表示网络第 l 层在时间步 t 的脉冲神经网络参数。不连续梯度 $\partial \mathbf{o} / \partial \mathbf{z}_l^t$ 在

算法 1： FalVolt缓解算法

输入： (i) 预训练权重：wts；(ii) 训练数据：trData；(iii) 测试数据：tsData；(iv) 故障映射：fmaps；(v) 时间步：T；(vi) 最大重新训练轮次：trEpochs；(vii) 学习率：η；

输出： 准确率：acc；

```
ind =FindPrunedWeightsIndices (fmaps, wts)
//从故障映射中查找剪枝权重的索引
pWts =将剪枝权重设为零(ind, wts)
//将上述索引处的剪枝权重设为零
(pVth, θ) = 参数初始化()
//初始化 θ 和阈值电压参数
对于 训练轮次 = 0 : trEpochs - 1执行
    对于t = 0 : T - 1执行
        对于l = 0 : L - 1 执行
            (nWts) = 更新权重 (pWts, ts, trData)
            //使用反向传播更新权重
            (nVth) =更新电压阈值 (pVth, ts, trData)
            //使用反向传播更新阈值电压
        结束循环
        L =计算损失(trData)
        // 计算交叉熵损失
        θ = θ - η ΔL
        //更新网络参数 θ
    结束循环
    nWts = 将更新后的权重设为零(nWts, ind)
    //使用步骤 1 中的索引将所有剪枝权重设为零
结束循环
准确率 =检查使用新权重的推理准确率(nWts, tsData)
//使用新权重检查推理准确率
返回 (nWts, nVth, acc);
```

重新训练中的误差反向传播期间，使用代理函数进行近似，与初始训练类似。术语 $\partial \mathbf{o} / \partial \mathbf{z}_l^t$ 的数学表达式为：

$$\frac{\partial \mathbf{o}_l^t}{\partial \mathbf{z}_l^t} = \gamma \max(0, 1 - |\mathbf{z}_l^t|) \tag{2}$$

其中，γ 是表示替代函数最大值的常数。在反向传播过程中，第 l 层的阈值电压$\mathbb{V}$ 按如下方式更新：

$$\overline{\mathbb{V}}_l = \overline{\mathbb{V}}_{l-1} - \eta \, \Delta\overline{\mathbb{V}} \tag{3}$$

其中，η 表示学习率。这里，第 l 层阈值电压$\Delta\mathbb{V}$ 的梯度可计算如下：

$$\Delta\overline{\mathbb{V}}_l = \frac{\partial L}{\partial \overline{\mathbb{V}}_l} = \sum_{t=0}^{T-1} \frac{\partial L}{\partial \mathbf{o}_l^t} \frac{\partial \mathbf{o}}{\partial \mathbf{z}_l^t} \frac{\partial \mathbf{z}}{\partial \overline{\mathbb{V}}_l} = \sum_{t=0}^{T-1} \frac{\partial L}{\partial \mathbf{o}_l^t} \frac{\partial \mathbf{o}}{\partial \mathbf{z}_l^t} \left(\frac{-\overline{\mathbb{V}}_l \mathbf{o}_l^{t-1} - v_l^t}{\overline{\mathbb{V}}_l^2} \right) \tag{4}$$

其中，L 表示由均方误差定义的交叉熵损失函数。算法 1 描述了所提出的 FalVolt 缓解方法。第 1–2 行对映射到脉动式脉冲神经网络中故障处理单元的预训练权重进行剪枝。第 3 行初始化重阶跃函数 θ 和 $\mathbb{V}$。第 4–5 行通过反向传播，经过多个训练轮次计算未剪枝权重和 $\mathbb{V}$。在 PLIF 脉冲神经网络的每个时间步中，针对每一层优化未剪枝权重和 $\mathbb{V}$，同时在第 10–11 行计算损失函数梯度（ΔL）。第 13 行在每个训练轮次结束时，将映射到故障处理单元的权重置为零。值得注意的是，将重新训练轮次设为零后，FalVolt

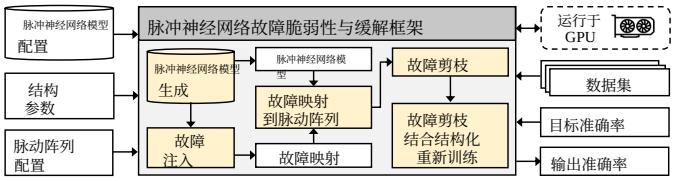


图4：实验设置与工具流程

等同于简单的故障感知剪枝（FaP）。FalVolt 会为每一层的未剪枝权重（或重新训练后的模型）返回新的优化值 $\mathbb{V}$，并提高分类准确率。请注意，所提出的缓解方法只需针对制造完成的芯片，依据其独特的故障图执行一次，因此有助于避免芯片的重新制造成本。

V. 结果与讨论

本节讨论对脉动式脉冲神经网络进行故障脆弱性分析和缓解分析所得的结果。

A. 数据集与网络架构

本文采用了一个静态MNIST [24], 数据集，以及两个神经形态数据集——N-MNIST [25] 和 DVS128 手势 [26] 。请注意，脉冲神经网络研究社区广泛使用这些数据集来评估脉冲神经网络的性能 [29], [16]。对于 N-MNIST 和 MNIST 数据集的分类，我们使用基于 PLIF 的脉冲神经网络，其中包含两组重复两次的卷积层、批量归一化、脉冲神经元和池化层，以及两组重复两次的丢弃层、全连接层和脉冲神经元层。在 DVS128 手势数据集的分类器中，前一组采用相同的架构配置重复五次。此外，受 [30], 启发，我们在这些架构中额外使用一组卷积层和脉冲神经元层，对输入图像进行脉冲编码。我们采用 [27] 中的初始化参数，在推理阶段进行故障注入之前，达到基线准确率：MNIST [24]和 N-MNIST [25]数据集为 99%，DVS128 手势 [26] 数据集为 97%。对于 systolicSNN 推理，我们使用 VHDL 开发了一个 256×256 的处理单元网格，并配备旁路电路，面积开销仅为 8%。

B. 仿真方法

图 4 展示了本文用于故障脆弱性分析和故障缓解分析的工具流程。首先，训练脉冲神经网络模型，使其达到基准准确率。接下来，使用不同的故障图，将固定型故障注入处理单元的累加器输出中。然后，将映射到故障处理单元的权重设为零，以实施故障剪枝。最后，使用算法 1，通过重新训练并进行逐层阈值电压优化来缓解故障。所有仿真均在搭载 Intel Core i9-10900kF（运行频率为 3.06 GHz、内存为 32 GB）的系统上，使用 NVIDIA GeForce RTX 2080 Ti 图形处理器进行。

C. 故障脆弱性分析

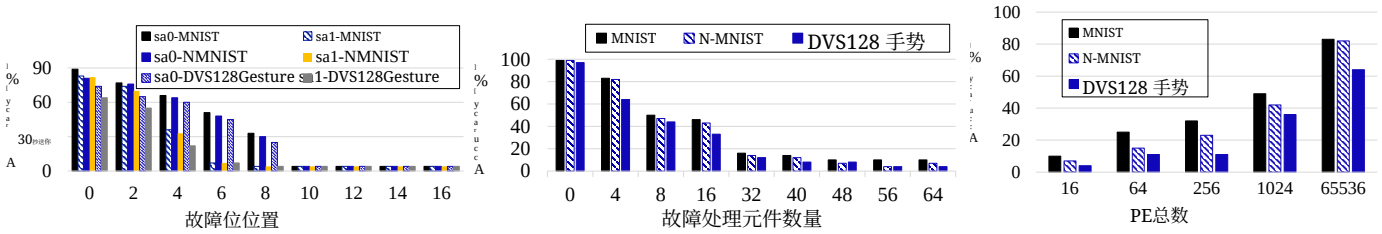
为研究脉动式脉冲神经网络中的固定型故障脆弱性，我们通过改变故障位的位置、故障处理单元的数量以及脉动阵列的规模，全面分析其影响，具体如下。

改变故障位的位置：在进行大规模故障缓解模拟之前，我们首先确定 256×256 脉动式脉冲神经网络中处理单元最容易受到固定型故障影响的位。为此，我们生成故障图，将固定为 0 和固定为 1 的故障注入处理单元内累加器的不同输出位位置。请注意，使用故障图进行故障注入是分析脉动阵列故障脆弱性的常见做法 [31]。在实际场景中，可以通过制造后测试生成故障图。值得一提的是，我们将故障注入累加器的输出中；累加器是处理单元的主要算术组件。如图 5a 所示，我们的分析表明，最高有效位（MSB）中的固定型故障对分类准确率的影响，大于最低有效位（LSB）中的固定型故障。这是因为脉动阵列会在不同层间重复使用；因此，某一层中处理单元里未被屏蔽的单个故障会影响后续层中的所有连接节点，从而降低整体分类准确率。我们还观察到，在对 MNIST、N-MNIST 和 DVS128 手势数据集进行分类时，最高有效位中的固定为 1 故障会导致近 80% 的准确率损失，高于最低有效位中的同类故障。值得注意的是，与脉动阵列人工神经网络加速器类似{systolic array ANN accelerators}，固定为 1 的故障在 systolicSNN 中造成的扰动比固定为 0 的故障更大 [20]。

改变故障处理单元数量：接下来，我们在 256×256 脉动式脉冲神经网络中，考虑固定型故障的随机分布，并据此进行故障仿真。通过改变每次实验中的故障处理单元数量来调整故障率，并将每次实验运行 8 次。每次实验的所有迭代中，故障处理单元数量保持不变。此外，每次迭代均使用不同的故障图。在下一节中，我们将故障注入处理单元累加器输出的高位比特（即最高有效位）中，以进行最坏情况分析。此外，我们记录每次实验中所有迭代的平均分类准确率。如图 5b 所示，结果表明，即使只有 *8* 个故障处理单元（即占处理单元总数的 *0.012%*），也会使 MNIST、N-MNIST 和 DVS128 手势分类的准确率分别从 99% 降至 50%、从 99% 降至 47%、以及从 97% 降至 44%。因此，静态数据集和神经形态数据集的分类都容易受到固定型故障的影响。

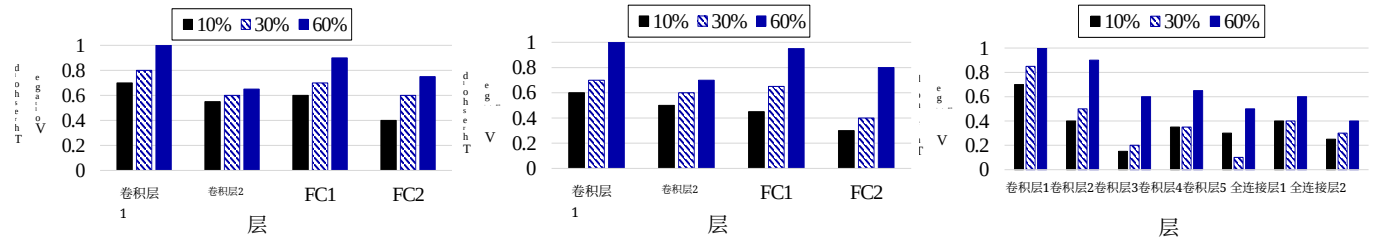
脉动阵列尺寸变化：为了开展更深入的故障脆弱性研究，我们分析了固定型故障对不同尺寸的*N*x*N* 脉动阵列的影响，即 4x4、8x8、16x16、32x32 和 64x64。如图 5c 所示，我们的分析表明，与大尺寸脉动阵列相比，小尺寸脉动阵列中的固定型故障会造成更大的准确率损失。例如，在一个 8x8 脉动阵列（含有 16 个处理单元）中，4 个处理单元发生故障，会使 MNIST、N-MNIST 和 DVS128 手势分类的准确率分别损失 89%、92% 和 93%。然而，对于采用 256x256 脉动阵列的脉冲神经网络分类任务，在故障配置相同的情况下，准确率损失仅约为 16%、17% 和 33%。这是因为脉动阵列尺寸越小，可复用性越高，因此永久性故障在每个执行周期中重复出现的可能性也越大。

我们的分析表明，DVS128 手势更容易受到



(a) 准确率与故障位位置　　(b) 准确率与故障处理单元数量　　(c) 准确率与脉动阵列规模

图 5：256x256 基于脉动阵列的脉冲神经网络加速器（systolicSNN）的固定型故障脆弱性分析。



（a）MNIST [24]分类　　（b）N-MNIST [25]分类　　（c）DVS128 手势 [26] 分类

图 6：在 256x256 脉动阵列脉冲神经网络加速器（systolicSNN）中，当总处理单元的 0%、10%、30% 和 60% 存在故障时，使用 FalVolt 优化隐藏卷积层和全连接层的阈值电压

故障的影响，相较于 MNIST 和 N-MNIST 数据集而言，尽管它们的基准准确率相同。如图 5b 所示，在存在固定型故障时，DVS128 手势的分类准确率仍低于其他数据集。此外，图 5c 显示，DVS128 手势数据集的准确率损失也高于其他数据集。然而，在所有情况下，固定型故障数量过多都可能导致无法接受的性能损失。

D. 故障缓解分析

在本节中，我们研究 FalVolt 的性能，并将其与人工神经网络中常用的最先进技术进行比较。具体而言，我们将 FalVolt 与故障感知剪枝（FAP）以及不进行阈值电压优化的带重新训练的故障感知剪枝（FaPIT）进行比较。

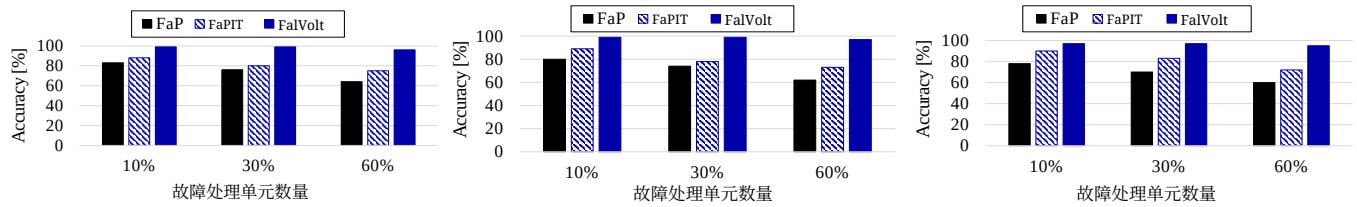
分类准确率与故障率：在故障缓解分析中，我们使用不同的故障图，在 256×256 脉动式脉冲神经网络中 10%、30% 和 60% 的处理单元上注入固定型故障，并运行并行重新训练仿真。对于 256×256 脉动式脉冲神经网络中 10%、30% 和 60% 的处理单元，我们采用算法 1 所述的 FalVolt 缓解方法。分析表明，对每个隐藏卷积层和全连接层进行阈值电压优化，有助于达到基线准确率。图 6 显示了针对 MNIST、NMNIST 和 DVS128 Gesture 数据集，为达到基线准确率，FalVolt 缓解方法为每个隐藏层求得的优化阈值电压。对于所有这些数据集，初始脉冲卷积层和脉冲全连接层的优化阈值电压均高于其他层，以确保冗余脉冲不会传递到输出层。

图 7 对比了 FalVolt 缓解方法与 FaP 和 FaPIT。我们观察到，故障率升高会导致 FaP 的准确率迅速下降。FaPIT 和 FalVolt 有助于提高分类准确率。然而，只有 FalVolt 在 MNIST、N-MNIST 和即使有 60% 的处理单元发生故障，DVS128 手势分类仍能达到基准分类准确率。这验证了 FalVolt 同时适用于静态数据集和神经形态数据集。

分类准确率与训练轮次的关系：FalVolt 以增加 FaP 所需的重新训练轮次为代价，提高了分类准确率；不过，与脉动式脉冲神经网络的生命周期相比，这些轮次微不足道。如图 8 所示，FaPVolt 的速度比 FaPIT 快 2 倍。例如，使用 FaPIT 训练 20 轮后，MNIST 的分类准确率可高达 80%，并在约 25 轮时收敛至基线准确率。然而，如图 8a 所示，同一数据集使用 FalVolt 仅需 10 轮便可达到基线准确率。同样，如图 8b 所示，与 FaPIT 相比，FalVolt 达到 NMNIST 分类基线准确率所需的训练轮次减少了一半。此外，如图 8c 所示，使用 FaPIT 训练 40 轮后，DVS128 手势的分类准确率可高达 83%，并在约 50 轮时收敛至基线准确率。然而，同一数据集使用 FalVolt 约 25 轮便可达到 97% 的准确率。由于基线准确率的微小变化可能会在安全关键型应用中引发灾难性问题，因此，初始训练、FaPIT 和 FalVolt 算法所采用的训练轮次较多，以使分类准确率接近基线。请注意，训练大型脉冲神经网络本身就需要很长时间（或较多训练轮次）。

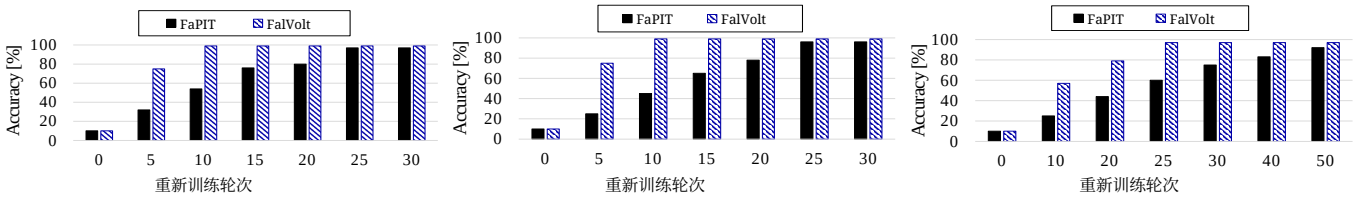
六、结论

本文全面分析了脉动式脉冲神经网络的固定型故障脆弱性，并提出了一种新颖的故障缓解技术“通过阈值电压优化进行故障感知再训练（*FalVolt*）”。FalVolt 使用经过优化的阈值电压和不同于初始训练的时间步，使分类准确率接近基准。为展示 FalVolt 的有效性，我们在 256×256 脉动式脉冲神经网络上对 MNIST、N-MNIST 和 DVS128 Gesture 数据集进行分类



(a) MNIST [24]分类　(b) N-MNIST [25]分类　(c) DVS128 手势 [26] 分类

图 7：在 256x256 脉动阵列脉冲神经网络加速器（systolicSNN）中，当总处理单元中有 0%、10%、30% 和 60% 存在故障时，使用 FaP、FaPIT（阈值电压设为 1.0）和 FalVolt 进行固定型故障缓解



(a) MNIST [24]分类　(b) N-MNIST [25]分类　(c) DVS128 手势 [26] 分类

图 8：在 256x256 脉动阵列脉冲神经网络加速器（systolicSNN）中，当 30% 的处理单元发生故障时，FaPIT 和 FalVolt 在不同训练轮次下的性能（systolicSNN）脉冲神经网络加速器

同时以不同故障率注入故障。结果表明，脉动式脉冲神经网络中即使只有 0.012% 的故障处理单元，也会导致显著的准确率损失。然而，FalVolt 使脉动式脉冲神经网络能够在高达 60% 的故障率下运行，并将分类准确率的下降幅度控制在可忽略的水平（最低仅为 0.1%），从而提升其性能。此外，结果还表明，与不进行阈值电压优化的故障感知剪枝等最先进技术相比，FalVolt 的速度快 2 倍。

参考文献

[1] E. Painkras 等人，“SpiNNaker：用于大规模并行神经网络仿真的 1 W、18 核片上系统”，*IEEE Journal of Solid- StateCircuits*，第 48 卷，第 8 期，第 1943–1953 页，2013 年。[2] F. Akopyan 等人，“TrueNorth：一款 65 mW、100 万神经元可编程神经突触芯片的设计与工具流程”，*IEEE transactions on computer-aided designof integrated circuitsand systems*，第 34 卷，第 10 期，第 1537–1557 页，2015 年。[3] J. J. Lee 等人，“脉动阵列加速器上时空脉冲神经计算的可重构数据流优化”，载于 ***ICCD***。IEEE，2020 年，第 57–64 页。[4] S. Guo 等人，“脉动式脉冲神经网络推理加速器及其协同优化软件框架”，载于 *Proceedings of the 2019 on Great Lakes Sym- posium on VLSI*，2019 年，第 63–68 页。[5] P. Y. Chuang 等人，“用于实时目标分类的 90 nm、103.14 TOPS/W 二进制权重脉冲神经网络 CMOS专用集成电路”，载于 ***DAC***。IEEE，2020 年，第 1–6 页。[6] P. Y. Tan 等人，“用于实时目标分类的高能效二进制权重脉冲神经网络架构”，*arXiv preprint arXiv:2003.06310*，2020 年。[7] J. J. Lee 等人，“并行时间分批：稀疏脉冲神经计算的脉动阵列加速”，***HPCA***，第 317–330 页，2022 年。[8] H. T. Kung 等人，“为高效脉动阵列实现打包稀疏卷积神经网络：联合优化下的列合并”，载于 ***ASPLOS***，2019 年，第 821–834 页。[9] R. e. a. El-Allami，“通过固有结构参数保护深度脉冲神经网络免受对抗攻击”，*arXiv preprint arXiv:2012.05321*，2020 年。[10] J. J. Lee 等人，“具有动态异构电压调节的脉动阵列脉冲神经网络加速器”，载于 ***IJCNN***。IEEE，2021 年，第 1–7 页。[11] C. D. Schuman 等人，“脉冲神经网络在神经形态系统中的弹性与鲁棒性”，载于 ***IJCNN***。IEEE，2020 年，第 1–10 页。[12] E. I. Vatajelu 等人，“特别专题：硬件实现的脉冲神经网络（SNN）的可靠性”，载于 ***VTS***。IEEE，2019 年，第 1–8 页。[13] W. Guo 等人，“脉冲神经网络中的神经编码：面向鲁棒神经形态系统的比较研究”，*Frontiers in Neuroscience*，第 15 卷，第 212 页，2021 年。[14] S. A. El-Sayed，“脉冲神经元硬件级故障建模”，载于 ***IOLTS***。IEEE，2020 年，第 1–4 页。[15] T. Spyrou 等人，“脉冲神经网络硬件加速器的可靠性分析”，载于 ***DATE***，2022 年。[16] R. V. W. Putra 等人，“Respawn：考虑不可靠存储器的脉冲神经网络节能容错”，载于 ***ICCAD***。IEEE，2021 年，第 1–9 页。[17] ——，“Softsnn：软错误条件下脉冲神经网络加速器的低成本容错”，*arXiv preprint arXiv:2203.05523*，2022 年。[18] V. Venceslai 等人，“Neuroattack：通过外部触发的位翻转破坏脉冲神经网络安全”，载于 ***IJCNN***。IEEE，2020 年，第 1–8 页。[19] M. Rastogi 等人，“星形胶质细胞在启用脉冲时序依赖可塑性的无监督脉冲神经网络中的自修复作用探究”，*Frontiers in Neuroscience*，第 14 卷，第 1351 页，2021 年。[20] A. Siddique 等人，“近似深度神经网络硬件加速器中的故障能耗权衡探究”，载于 ***ISQED***。IEEE，2021 年，第 343–348 页。[21] J. J. Zhang 等人，“分析并缓解永久性故障对基于脉动阵列的神经网络加速器的影响”，载于 ***VTS***。IEEE，2018 年，第 1–6 页。[22] M. A. Hanif 等人，“Salvagednn：通过显著性驱动的故障感知映射，利用永久性故障挽救深度神经网络加速器”，*Philosophical Transactions of the Royal Society A*，第 378 卷，第 2164 期，第 20190164 页，2020 年。[23] S. Kundu 等人，“深度神经网络加速器中制造故障的高层建模”，载于 ***IOLTS***。IEEE，2020 年，第 1–4 页。[24] Y. LeCun 等人，“MNIST 手写数字数据库”，*ATT 实验室 [Online]. Available: http://yann.lecun.com/exdb/mnist*，第 2 卷，2010 年。[25] G. Orchard、A. Jayawant、G. K. Cohen 和 N. Thakor，“使用扫视将静态图像数据集转换为脉冲神经形态数据集”，*Frontiers in neuroscience*，第 9 卷，第 437 页，2015 年。[26] A. Amir 等人，“低功耗、完全基于事件的手势识别系统”，载于 ***CVPR***，2017 年，第 7243–7252 页。[27] W. Fang 等人，“引入可学习膜时间常数以增强脉冲神经网络的学习能力”，载于 ***ICCV***，2021 年，第 2661–2671 页。[28] S. Q. Wang 等人，“Sies：在现场可编程门阵列上实现脉冲卷积神经网络推理引擎的新方法”，*Journal of Computer Scienceand Technology*，第 35 卷，489，2020 2，第 475no. pp. – 。[29] J. Morris 等人，“Hyperspike：利用超维计算实现更高效、更鲁棒的脉冲神经网络”，载于 ***DATE***。IEEE，2022 年，第 664–669 页。[30] C. Lee、S. S. Sarwar、P. Panda、G. Srinivasan 和 K. Roy，“实现基于脉冲的反向传播，以训练深度神经网络架构”，*Frontiers inneuroscience*，第 119 页，2020 年。[31] M. A. Hanif 等人，“可信深度学习：面向深度神经网络加速器抵御软错误和永久性故障的高成本效益韧性”，载于***IOLTS***。IEEE，2020 年，第 1–4 页。