

神经网络：一种具有混合数据事件执行和即时注意力数据流的弹性 神经元同构 A 架构

Yuehai Chen, and Farhad Merchant

Bernoulli Institute and CogniGron, University of Groningen, The Netherlands

Email: {yuehai.chen, f.a.merchant}@rug.nl

摘要—尖峰神经网络 (SNNs) 作为人工神经网络 (ANNs) 的一种有前景的替代方案，通过利用稀疏和事件驱动的计算提供了改进的能量效率。然而，现有的 SNN 硬件实现仍然受到固有的脉冲稀疏性和多步时序执行的影响，这显著增加了延迟并降低了能量效率。本研究介绍了 NEURAL，一种基于混合数据-事件执行范式的新型神经形态架构，通过将感知稀疏性的处理与神经元计算解耦，并使用弹性先入先出 (FIFO) 实现。NEURAL 支持在基线计算流程中嵌入尖峰 QKFormer 的操作以即时执行，而无需专用硬件单元。它还集成了一个窗口到时间到首次脉冲 (W2TTFS) 机制来替代平均池化并启用全脉冲执行。此外，我们引入了一种基于知识蒸馏 (KD) 的训练框架来构建具有竞争力准确性的单步时序 SNN 模型。NEURAL 在 Xilinx Virtex-7 FPGA 上实现，并使用 ResNet-11、QKResNet-11 和 VGG-11 进行评估。实验结果表明，在算法层面，使用 KD 训练的 VGG-11 模型在 CIFAR-10 上的准确率提高了 3.20%，在 CIFAR-100 上提高了 5.13%。在架构层面，与现有的 SNN 加速器相比，NEURAL 实现了资源利用率减少 50% 和能效提升 1.97 倍。

Index Terms—尖峰神经网络，弹性计算，稀疏感知，知识蒸馏，尖峰变压器

I. 介绍

最近，越来越多的边缘设备获得了执行智能处理的能力。尽管当前主流的人工神经网络 (ANNs) 表现出色，但由于其复杂的计算和高能耗，在资源受限的边缘设备上部署仍具挑战性。相比之下，脉冲神经网络 (SNNs) 以二进制尖峰的形式传递信息，这本身具有低功耗和事件驱动计算的特点。然而，由于多步推理导致的高延迟、限制反向传播的二进制信号以及通用硬件难以高效支持事件驱动机制等问题仍然存在。为了解决这些问题，算法和硬件协同设计成为提高 SNN 执行效率的关键。图 1 展示了 ANNs 与 SNNs 硬件实现之间的差异。与 ANNs 相比，SNNs 消除了对乘法器和复杂激活

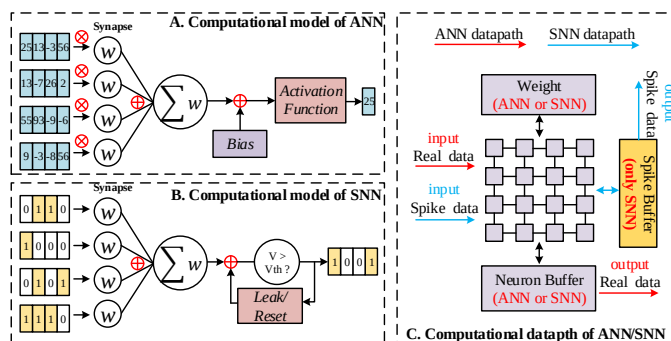


图 1: 人工神经网络和脉冲神经网络计算模型之间的差异。

函数的需求，而是依赖于加法和比较器，但需要额外的内存来存储膜电位和尖峰数据以进行时间处理。研究人员提出了 3D 架构 [1] [2] 以并行处理多个时间步长为代价，同时稀疏感知架构 [3] 利用尖峰的稀疏性优化了计算。在模型优化方面，替代梯度方法 [4] 在过去几年中显著减少了时间步骤的数量，同时保持准确性。知识蒸馏 (KD) [5] [6] 也被广泛用于训练低时间步 SNN，通过 ANN 教师网络引导学生网络学习更好的表示，实现了低延迟和高精度之间的良好平衡。同时，基于脉冲的变压器机制 [7] [8] 进一步提高了 SNN 模型的识别准确性。然而，大多数现有方法仍停留在算法层面，仅构建类似脉冲计算图而没有真正实现脉冲执行。这突出了算法-硬件协同设计的重要性，强调优化不仅应在算法层面上进行，还应从架构层面上进行，以充分利用神经形态计算的效率潜力。为进一步探索高效的 SNN 硬件架构，我们专注于建立一个具有全脉冲计算和稀疏感知能力且能在单个时间步内执行推理的协同优化架构。在算法层面实现高精度单时间步 SNN 消除了对多时间步调度的需求，从而降低了推理延迟并减少了控制逻辑复杂度。此外，识别那些保持非脉冲的操作并将它们转

化为基于脉冲的对应操作可以进一步提高事件驱动计算的能效。根据这些观察，在本文中，我们提出了 NEURAL，一种支持弹性连接和实时注意力数据流的混合数据-事件执行神经形态计算架构。主要贡献如下。

- 1) 训练框架结合了知识蒸馏和定点量化，使单时间步 SNN 能够达到与多时间步模型相当的准确性
- 2) 窗口到首次尖峰时间 (W2TTFS) 机制，用于在保持准确性的同时将非尖峰平均池化转换为基于尖峰的计算
- 3) 使用弹性 FIFO 调度的混合数据-事件执行，以实现数据驱动控制和事件驱动神经元计算，同时支持在飞 spike 的 QKFormer [8] 而无需专用硬件单元
- 4) FPGA 实现的 NEURAL 架构部署了三个深度 SNN，即 VGG-11、ResNet-11 和 QKFormer-11，在基于知识蒸馏训练的情况下，分别在 CIFAR-10 和 CIFAR-100 上达到了高达 93.46% 和 72.1% 的准确率。在单时间步执行范式下，NEURAL 显著优于现有的 STI-SNN [9] 架构，计算效率提高了近 3.9 倍。

本文其余部分组织如下。第二节总结了相关的神经形态算法和架构。第三节介绍了 W2TTFS 机制以及基于知识蒸馏的训练框架。第四节描述了 NEURAL 架构的详细设计。第五节评估并分析了 NEURAL 的性能。最后，第六节对本文进行了总结，并讨论了未来方向。

II. 相关工作

神经形态计算在算法和架构方面都在不断发展。在算法层面，尖峰模型及训练方法的发展使得 SNN 能够接近 ANN 的性能。在架构层面，利用 SNN 固有的事件驱动计算和稀疏性来提高能效。因此，我们从这两个方面总结相关工作。

神经形态算法。随着尖峰计算理论的发展，网络结构已经从最初的简单尖峰多层感知器 (SMLP) 扩展到深层尖峰卷积神经网络 (DSCNN)。训练方法经历了从生物启发的尖峰时间依赖可塑性 [10](STDP) 和 ANN-to-SNN [11] 到基于替代梯度的监督训练方法 [4] 的转变。为了进一步利用深度学习中 ANN 模型的表现力，一些研究探索了使用 KD 来改进 SNN 的训练。最近的工作表明，通过 KD 训练的 ResNet-19 SNN 模型可以达到 96.65% 和 81. 在 CIFAR-10 和 CIFAR-100 数据集上分别实现了 47% 的准确性，仅使用了 2 个时间步骤 [6]，显著提高了低步长 SNN 的实际价值。

神经形态架构。与通用处理器相比，专用神经拟态架构更适合事件驱动和稀疏计算模型的 SNN，从而提高能效和吞吐量。现有的 SNN 加速器研究可以归类为以下方向：一类工作专注于高效实现脉冲 MLP [12] [13]；另一类设计了感知稀疏性的执行机制以利用固有的激活稀疏性。感知稀疏性的执行机制 [3] [14] 可以动态跳过零值计算并减少冗余访问，从而显著提高计算效率。此外，为了减轻 SNN 多时间步特性引起的延迟和计算开销，各种研究探索了时间步压缩技术和时间并行计算架构 [1] [2] [15]。然而，这些方法在节省延迟的同时往往引入更复杂的资源调度和更高的片上开销，使其难以部署在资源受限的边缘环境中。NEURAL 弥合了算法改进与架构支持之间的差距，从而解决了之前 SNN 硬件的主要局限性。它通过协同优化训练和弹性硬件设计实现高精度、低延迟、完全基于脉冲的执行，并且面积更小，同时支持如 QKFormer 等新兴 SNN 模块在实时执行。这使得 NEURAL 成为下一代神经形态计算系统的一种实用且可扩展的解决方案。

III. 全尖刺 SNN 模型

SNN 通过脉冲传输信息，具有固有的计算和能效优势。然而，当前主流模型难以实现完全基于脉冲的计算路径，尤其是在下采样阶段使用了平均池化。尽管平均池化在稀疏输入下提高了训练稳定性，但它引入了连续值，破坏了基于脉冲执行的一致性并增加了计算和能耗开销。为了解决这个问题，我们提出了 W2TTFS 机制，在推理过程中将脉冲窗口转换为跨多个时间步的首次脉冲到达时间表示。通过保持分类器的完全脉冲输入，该机制平衡了分类准确性和硬件效率。

A. 窗口到首次尖峰时间 (W2TTFS)

如图 2 (a) 所示，为了提升模型性能，我们通过添加 QKFormer Blocks [8] 来增强传统的 ResNet-11 [16] 主干网络，并将其命名为 QKFormer-11 以集成注意力机制。标准的平均池化 (AP) 操作将尖峰信号转换为连续值，导致分类器接收非尖峰输入。为了解决这个问题，我们引入了 W2TTFS 方法，如图 2 (a) 中推理模块所示，该方法确保分类器接收到基于尖峰的输入。W2TTFS 的具体转换流程显示在算法 1 中。第 4 和 5 行计算 AP 操作的感受野大小 (表示为 $window_size$)，并根据最终卷积层生成的特征图维度初始化一个具有 $window_size^2$ 时间步长的零矩阵。第 8 - 16 行通过通

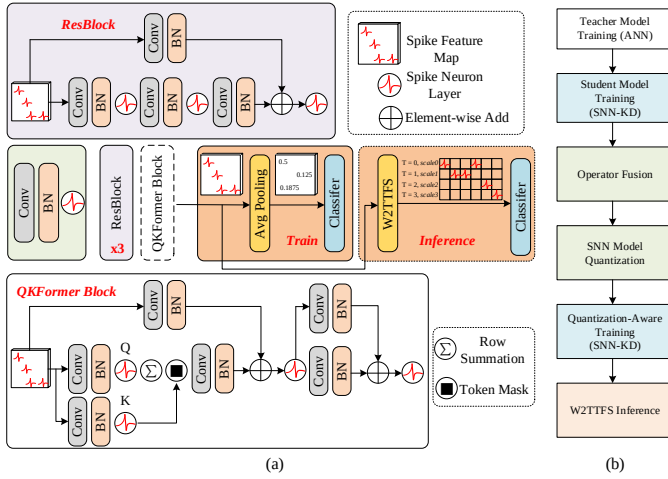


图 2: QKFRNet-11 模型及训练框架概述。(a) QKFRNet-11。(b) 基于知识蒸馏的 SNN 模型训练流程。

道和空间位置遍历输出特征图，通过计数每个池化窗口内的脉冲数量来识别第一次有效的脉冲时刻，如在第 11-13 行中执行的操作所示。最后，第 17 - 20 行计算时间依赖的权重缩放因子。例如，对于 $window_size = 4$ ，如果首次发出的有效脉冲是在时间步长 $t = 3$ 时，则对应的缩放因子为 $3/16$ 。该因子随后用于在全连接 (FC) 计算过程中对权重进行缩放。

B. 基于知识蒸馏的单时间步 SNN

在本文中，我们介绍了一个用于 SNN 的基于知识蒸馏 (KD) 的训练框架，如图 2(b) 所示。我们首先训练一个高精度的 ANN 作为教师模型。然后，构建一个 SNN 作为学生模型并通过知识蒸馏进行训练。如图 2(a) 所示，典型的 SNN 模型通常包含批归一化 (BN) 等层，这些对硬件部署提出了挑战。为了解决这个问题，我们应用操作融合和定点量化来减少模型复杂度和硬件资源使用。由于量化可能会降低精度，我们进一步采用基于知识蒸馏的感知量化训练 (QAT) 以减轻精度损失。最后，在推理过程中，我们将 AP 层替换为所提出的 W2TTFs 模块以实现全脉冲执行。

IV. 神经架构

如图 3 所示，NEURAL 架构包括三个关键模块：弹性处理元件阵列 (EPA)、流水线稀疏检测阵列 (PipeSDA) 和基于 W2TTFs 的全连接计算核心 (WTFC)。权重通过弹性 W-FIFO 流进 EPA，这些权重来源于权重管理单元 (WMU)。WMU 根据当前计算状

Algorithm 1 窗口到首次尖峰时间 (W2TTFs)

```

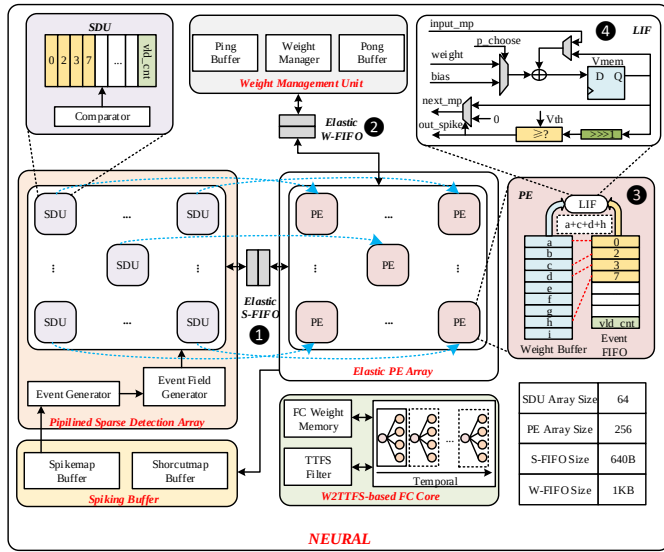
1: 给定: spike_map 表示来自下采样卷积层的输出。
2: 给定: spike_cnt 是获取有效脉冲数量的函数。
3: 给定:  $T$ ,  $C$ ,  $H_i$ ,  $W_i$ ,  $H_o$  和  $W_o$  分别表示时间步长、通道数、输入高度、输入宽度、输出高度和输出宽度。
4: 窗口大小  $\leftarrow H_i // H_o$ 
5: spike_array_fc  $\leftarrow$  零点 ( $window\_size^2, C, H_o * W_o$ )
6: for  $t = 0$  to  $T$  do
7:   spike_array_fc.reset()
8:   for channel = 0 to  $C$  do
9:     for  $h = 0$  to  $H_o$  do
10:      for  $w = 0$  to  $W_o$  do
11:        // 窗口
12:        池化窗口  $\leftarrow$  spike_map.get( $h, w$ )
13:        有效计数  $\leftarrow$  池化窗口.spike_cnt()
14:        // 首次尖峰时间
15:        spike_array_fc[vld_cnt, channel,  $h*w$ ] = 1
16:      end for
17:    end for
18:  end for
19:  for  $tt = 0$  to window_size do
20:    scale =  $tt/window\_size^2$ 
21:    spike_array_fc[tt].展平.分类器(尺度)
22:  end for
23: end for

```

态从片外内存动态调度所需的权重并将其送入 FIFO。输入脉冲以类似的方式处理，当有效的脉冲阵列在弹性 S-FIFO 中被缓冲时，EPA 读取它们并与权重进行并行计算。PipeSDA 模块基于其坐标识别每个输入脉冲的事件感受野，并将其映射到适当的稀疏检测单元 (SDU)。SDU 生成局部卷积窗口，然后转发给 EPA 进一步处理。WTFC 模块执行由 W2TTFs 层定义的计算，在分类器阶段实现完全尖峰推理。这种端到端的尖峰设计使得 SNN 模型的所有计算层都能在 NEURAL 架构中高效执行，从而提高系统级集成和执行效率。

A. 混合数据事件执行数据流

本研究提出了一种基于弹性 FIFO 的数据-事件混合执行机制，在架构层面采用数据驱动的控制流，而在单个神经元计算粒度上切换到事件驱动的执行。**数据执**



行。如图 3 所示，EPA 的左侧输入是尖峰序列 ①，而上方则是相应的权重矩阵 ②，使得系统在两端的数据都可用时即可触发计算，无需依赖集中式控制。**事件执行。**如图 3 所示，每个 PE 包含一个专用事件 FIFO，其末尾寄存器存储当前有效事件的数量，vld_cnt ③。在计算过程中，PE 根据 vld_cnt 的顺序从 FIFO 中读取事件索引，并获取相应的权重并将其发送到 LIF 单元，实现膜电位 (MP) 更新计算。LIF 单元使用相应权重更新 MP，并进行阈值比较以确定是否发出尖峰 ④。它是完全基于事件驱动的，从而避免了无尖峰期间的冗余更新。

B. 流水线稀疏检测阵列设计

PipeSDA 的主要阶段包括索引生成 (IG)、中心位置 (CP) 生成以及将 CP 映射到 SDU 阵列 (CP Map) 以进行事件检测。**索引生成:** 如图 4 所示, 系统首先从输入尖峰图像中生成所有有效尖峰的索引, 并将其存储在缓冲区中。**CP 生成:** CP 生成: 每个脉冲事件对应的接受野的 CP 基于生成的索引进行计算。**CP 映射:** 这些 CP 被映射到 SDA 中的特定 SDU。由于某些 CP 的坐标可能是负数, 因此在 SDA 架构中预先定义了虚拟 SDU 以支持负索引映射。一旦映射完成, 位于 CP 位置的 SDU 向其邻近单元广播扩散信号 (如图 4 右侧所示), 表明对应的 SDU 位于活跃区域内。接收到信号的每个 SDU 随后根据映射结果更新其内部事件 FIFO, 为后续处理中的卷积窗口构建做好准备。

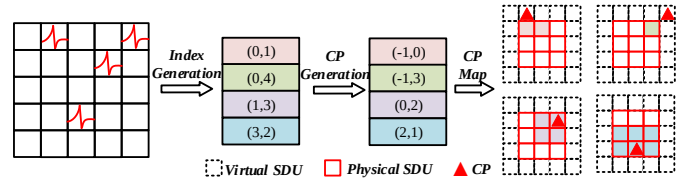


图 4: 流水线稀疏检测数据流。

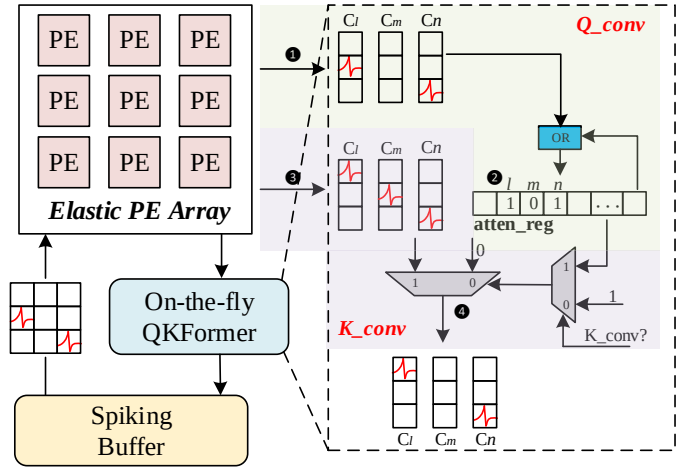


图 5: 实时计算数据流的 QKFormer。

C. 实时 QKFormer 计算

提出的 NEURAL 架构支持在基线数据流中即时计算 QKFormer，无需单独的尖峰转换器单元。

如图 5 所示, QKFormer 操作直接嵌入从 EPA (尖峰由相应的 PE 生成) 到尖峰缓冲区的写回路径中。

如图 5 所示, 在计算出 Q 矩阵 ❶ 之后, 我们通过通道间的按位 OR 运算使用注意力寄存器 (atten_reg 记录) 生成注意力激活状态 ❷, 这对应于图 2 中沿 Q 路径的行求和操作。随后, 计算 K 矩阵 ❸ 并写回到尖峰缓冲区。在此过程中, atten_reg 用于确定每个通道的激活状态 (0/1), 然后将其应用为 QK 标记掩码 ❹, 与图 2 中的标记掩码对齐。

D. 基于 $W2TFS$ 的 FC 核心

图 6 显示了基于 W2TTFS 的全连接核心架构，该架构由两个核心模块组成：TTFS 滤波器和全连接计算单元 (FCU)。输入的脉冲特征映射根据通道顺序依次进入 TTFS 滤波器，而 TTFS 滤波器的主要功能是根据池化窗口的大小统计每个窗口中的有效脉冲数量（标记为 `vld_cnt`），并生成相应的权重缩放因子。然而，如算法 1 所示，比例值可能包含非移位友好的小数（例如 $3/16$ ）。NEURAL 执行了对尺度生成策略的细粒度优化：比例不再依赖于特定的脉冲位置，而是统一设置

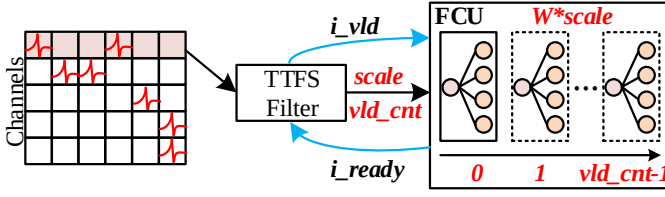


图 6: 基于 W2TTFS 的 FC 核心。

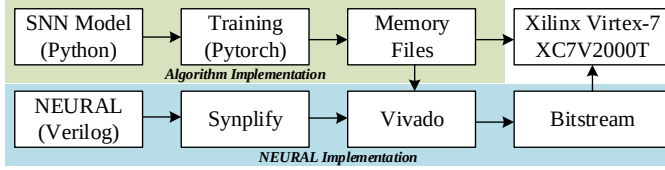


图 7: 算法和硬件实现的设计流程。

为池化窗口的逆单位（例如 1/16）。随后，复杂的比例通过时间复用策略进行近似处理。例如，当算法需要执行 3/16 的比例操作时，系统重复三次单元求和来完成相应的膜电位更新，避免了乘法和高精度除法运算。

V. 实验评估

A. 实验设置

算法实现。我们实现了四个 SNN 模型：VGG-11、ResNet-11、QKResNet-11 和基于 PyTorch 和 Spiking-Jelly 的 ResNet-19 [17]。LIF 神经元衰减参数 τ 为 0.5，时间步长设置为 1。这些模型在 CIFAR-10 和 CIFAR-100 数据集上使用 NVIDIA RTX 2080TI 进行了训练和测试。训练使用基于 logit 的 KD 框架 [6] 进行，教师模型是 ResNet-34，使用带有动量 0.9、批量大小 128 和 300 个 epoch 的 SGD 优化器。

硬件实现。获取量化模型后，生成内存文件用于硬件实现，如图 7 所示。NEURAL 用 Verilog HDL 实现，使用 Synplify 进行综合，并使用 Xilinx Vivado 进行布局布线。该设计在 Xilinx Virtex-7 XC7V2000T FPGA 上以 200MHz 运行。

B. 算法分析

如图 8 所示，它包含四种模型类型：KDT：使用 KD 训练的全精度模型；F & Q：应用算子融合和定点量化简化的模型；KD-QAT：基于 KD 的 QAT 模型；W2TTFS：本文提出的一种硬件友好的模型。如图 8(a) 所示，我们用 KD 训练的单时间步长 VGG-11 在全精度设置下，在 CIFAR-10 上实现了 94.06% 的准确率，比 [2] 高出 3.01%，后者是使用 4 个时间步骤评估的。量化后，KD-QAT VGG-11 仅损失了 0.17% 的准确率，这比 [2]

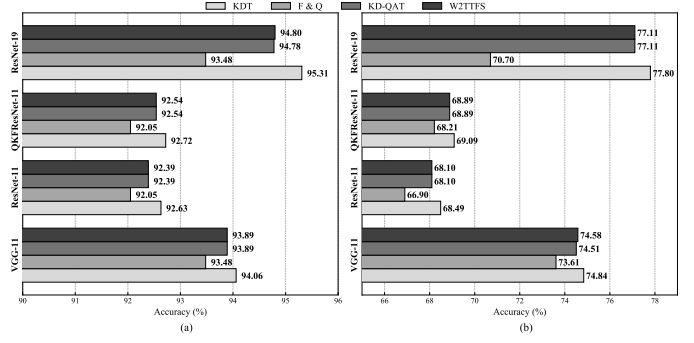


图 8: CIFAR-10/CIFAR-100 基于不同模型的准确性。(a) CIFAR-10 的准确性。(b) CIFAR-100 的准确性。

表 I: 硬件资源成本的神经网络

资源	管道 SDA	环保局	世贸组织	总计
LUTs	9K (12%)	33K (45%)	1K (1%)	74K
Registers	10K (16%)	15K (24%)	0.7K (1%)	63K
BRAM	3 (2%)	64 (47%)	25 (18%)	137.5

小 0.34%。在 CIFAR-100 上，改进更为显著，分别达到了 5.77% 和 1.15%。此外，KD-QAT 在保持高准确性方面有明显优势。例如，如图 8(b) 所示，ResNet-19 在 F & Q 后准确率下降了近 7%，而在 KD-QAT 微调后的准确率损失仅为 0.69%。具体而言，QKResNet-11 通过结合 QKFormer Block，在 CIFAR-100 上将 ResNet-11 的准确性提高了 0.79%。因此，基于 KD 的模型训练和量化方法可以有效地提高单时间步长 SNN 模型的准确率，减少由于量化导致的表现损失，并增强模型的鲁棒性。

C. 计算资源与能源分析

资源分析：我们在 NEURAL 上部署了与 SiBrain [2] 和 SCPU [16] 中相同的 VGG-11 和 ResNet-11 网络，并在 CIFAR-10 和 CIFAR-100 数据集上对其进行了评估。如图 9 所示，得益于单时间步执行范式，NEURAL 仅消耗约 70K LUTs，比其他架构少大约 50%，并且 RAM 使用量也减少了近 50%。在识别准确率方面，在 NEURAL 部署的 VGG-11 模型分别在 CIFAR-10 和 CIFAR-100 上实现了 93.45% 和 72.1%，这是所有比较架构中的最高值。同样，如图 9 所示，ResNet-11 部署的结果也是一致的。表 I 还报告了每个关键组件的资源情况。EPA 几乎占用了总资源的一半，因为它是最主要的计算引擎。

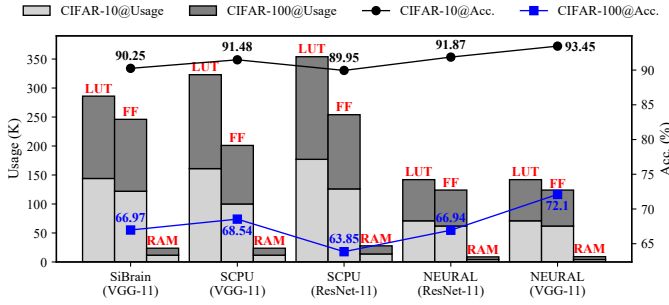


图 9: VGG-11/ResNet-11 在不同平台上的执行结果: 资源和准确性, 使用 CIFAR-10/100 数据集。

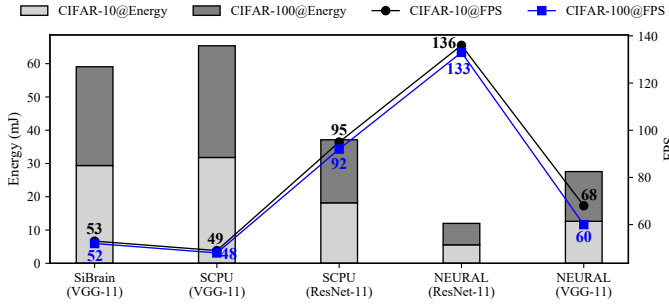


图 10: VGG-11/ResNet-11 在不同平台上使用 CIFAR-10/100 的执行结果: 计算能耗和吞吐量 (FPS)。

稀疏处理引起的硬件开销很小, 因为其内部逻辑非常简单。对于提出的 WTFc, 它使用 1K LUTs 和 0.7K 寄存器, 使其非常适合边缘设备。

能量分析: 在边缘计算中, 能耗直接影响系统的部署能力和生命周期。如图 10 所示, 在单张图像推理过程中, 与基线架构 [2] [16] 相比, NEURAL 的能耗降低了近 50%, 并且每秒处理帧数 (FPS) 也有所提高。具体来说, 在 VGG-11 (CIFAR-10) 任务中, NEURAL 实现了 68 FPS 的帧率, 而单张图像的能耗仅为约 10 毫焦耳; 在 ResNet-11 (CIFAR-10) 任务中, 帧率增加到 136 FPS, 同时能耗保持在 10 毫焦耳以下。结果表明, 在保证识别准确性的前提下, NEURAL 能够实现更高的能效比。

D. 残差网络-11 对比 QKF 残差网络-11

表 II 描述了在 NEURAL 上部署 ResNet-11 和 QKFResNet-11 的对比。如表 II 所示, QKFResNet-11 中注意力层的整合提高了两个数据集上的分类精度, 在 CIFAR-100 上获得了显著的 1.59% 增益。由于网络深度增加, QKFResNet-11 引入了大约额外 2 毫秒的延迟。总脉冲 (TS) 表示推理过程中生成的总脉冲数量。如表 II 所示, QKFormer 的整合可以在 CIFAR-10 上

表 II: ResNet-11 与 QKFResNet-11 在 CIFAR-10/100 上的比较

数据	模型	总尖峰	准确率 (%)	延迟 (毫秒)	能量 (毫焦)
CIFAR-10	ResNet-11	76K	91.87	7.3	5.56
	QKFResNet-11	72K	92.01 (+0.14)	9.7	8.14
CIFAR-100	ResNet-11	83K	66.94	7.5	6.44
	QKFResNet-11	84K	68.53 (+1.59)	9.9	8.26

减少 TS 以实现高效的脉冲抑制, 同时在 CIFAR-100 上增加 TS 以适应更高的任务复杂性。重要的是, 与 [16] 相比, 我们的 QKFResNet-11 实现了 4.68% 的精度提升, 同时减少了 10 毫焦的能量消耗。

E. 与先前神经拟态架构的比较

每瓦特千兆突触运算次数每秒 (GSOPS/W) 是评估 SNN 硬件架构能效的一个常见指标。如表 III 所示, 当在 CIFAR-10 数据集上使用 FP8 精度部署 ResNet-11 模型时, NEURAL 实现了 91.87% 的准确率、136 FPS 的帧速率、仅 0.758 W 的功耗以及 46.65 GSOPS/W 的能效。此外, 在同一数据集上使用 VGG-11 模型时, 准确率提高到 93.45%, 帧速率为 68 FPS, 功耗为 0.792 W, 并且能效进一步提升至 52.37 GSOPS/W。与其它最先进的平台相比, NEURAL 的功耗远低于 SiBrain [2] (1.56 W) 和 STI-SNN [9] (1.34 W), 同时实现了更高的准确率和效率。与此同时, 在相同单时间步设置下, 与 STI-SNN [9] 相比, NEURAL 的计算效率提高了高达 3.9 倍。为了公平比较, 我们采用每个千个查找表的能效 (GSOPS/W/kLUTs) 作为评估指标。如表 III 所示, NEURAL 实现了最高的归一化能效 0.73。特别是与 [3] 相比, NEURAL 在 CIFAR-10 上仅损失了 0.03% 的准确率, 同时显著提升了 FPS 46, 并将归一化能效提高了 1.97 倍。

VI. 结论

本文提出了 NEURAL, 一种具有弹性互连的混合数据-事件执行神经拟态架构, 该架构能够实现高效的 SNNs 执行, 并支持实时尖峰 QKFormer 计算。我们引入了一种基于知识蒸馏 (KD) 的训练框架, 以在单时间步长执行中达到竞争性精度。为了硬件友好部署, 训练

表 III: 与现有 SNN 加速器在 CIFAR-10 上的比较

平台	[2]	[3]	[9]	[18]	神经的	
设备	V. 7	Z. 7	Z. U	V. U	V. 7	
Fmax (兆赫兹)	200	200	200	100	200	
模型	VGG-11	MobileNet	SCNN5	VGG-9	ResNet-11	VGG-11
精度	FP8	N/A	INT8	IN4	浮点 8	
准确率 (%)	90.25	91.90	90.31	86.6	91.87	93.45
每秒帧数	53	90	397	120	136	68
功率 (瓦)	1.56	1.4	1.53	0.73	0.76	0.79
效率 (全局最优搜索程序/书)	84.16	31.6	13.46	64.11	46.65	52.37
范数. 效率 (全局信号优化与选择/写入 kLUTs)	0.60	0.37	0.52	0.58	0.65	0.73

好的 SNN 模型进行了操作融合、量化和基于 KD 的知识蒸馏量化 (KD-QAT)。此外，我们提出了 W2TTFs 机制作为 AP 层的替代方案，实现了全尖峰执行。实验结果表明，NEURAL 在显著减少硬件资源消耗的情况下达到了高精度和高性能。未来，我们计划探索更多基于 NEURAL 的应用，包括图像分割 [19] 和尖峰大型语言模型 [20]，以进一步推动节能神经拟态计算的实际应用。

参考文献

- [1] C. Fang, Z. Shen, Z. Wang, C. Wang, S. Zhao, F. Tian, J. Yang, and M. Sawan, "An energy-efficient unstructured sparsity-aware deep SNN accelerator with 3-D computation array," *IEEE Journal of Solid-State Circuits*, vol. 60, pp. 977–989, Mar. 2025.
- [2] Y. Chen, W. Ye, Y. Liu, and H. Zhou, "SiBrain: A sparse spatiotemporal parallel neuromorphic architecture for accelerating spiking convolution neural networks with low latency," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 71, pp. 6482–6494, Dec. 2024.
- [3] Q. Chen, C. Gao, and Y. Fu, "Cerebron: A Reconfigurable Architecture for Spatiotemporal Sparse Spiking Neural Networks," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 30, pp. 1425–1437, Oct. 2022.
- [4] Y. Wu, L. Deng, G. Li, J. Zhu, and L. Shi, "Spatio-temporal back-propagation for training high-performance spiking neural networks," *Frontiers in Neuroscience*, vol. 12, May 2018.
- [5] X. Liang, G. Chao, M. Li, and Y. Zhao, "Knowledge distill for spiking neural networks," in *2024 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, June 2024.
- [6] C. Yu, X. Zhao, L. Liu, S. Yang, G. Wang, E. Li, and A. Wang, "Efficient logit-based knowledge distillation of deep spiking neural networks for full-range timestep deployment," May 2025.
- [7] M. Yao, J. Hu, Z. Zhou, L. Yuan, Y. Tian, B. Xu, and G. Li, "Spike-driven transformer," *Advances in Neural Information Processing Systems*, vol. 36, pp. 64043–64058, Dec. 2023.
- [8] C. Zhou, H. Zhang, Z. Zhou, L. Yu, L. Huang, X. Fan, L. Yuan, Z. Ma, H. Zhou, and Y. Tian, "QKFormer: Hierarchical spiking transformer using Q-K attention," *Advances in Neural Information Processing Systems*, vol. 37, pp. 13074–13098, Dec. 2024.
- [9] K. Wang, C. Yang, C. Yu, Y. S. Ang, B. Wang, and A. Wang, "STI-SNN: A 0.14 GOPS/W/PE single-timestep inference FPGA-based SNN accelerator with algorithm and hardware co-design," June 2025.
- [10] P. U. Diehl and M. Cook, "Unsupervised learning of digit recognition using spike-timing-dependent plasticity," *Frontiers in Computational Neuroscience*, vol. 9, Aug. 2015.
- [11] B. Rueckauer, I.-A. Lungu, Y. Hu, M. Pfeiffer, and S.-C. Liu, "Conversion of continuous-valued deep networks to efficient event-driven networks for image classification," *Frontiers in Neuroscience*, vol. 11, Dec. 2017.
- [12] H. Chu, Y. Yan, L. Gan, H. Jia, L. Qian, Y. Huan, L. Zheng, and Z. Zou, "A neuromorphic processing system with spike-driven SNN processor for wearable ECG classification," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 16, pp. 511–523, Aug. 2022.
- [13] D.-A. Nguyen, X.-T. Tran, K. N. Dang, and F. Iacopi, "A low-power, high-accuracy with fully on-chip ternary weight hardware architecture for deep spiking neural networks," *Microprocessors and Microsystems*, vol. 90, p. 104458, Apr. 2022.
- [14] I. Aliyev and T. Adegbiya, "PULSE: Parametric hardware units for low-power sparsity-aware convolution engine," in *2024 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–5, May 2024.
- [15] J.-J. Lee, W. Zhang, and P. Li, "Parallel time batching: Systolic-array acceleration of sparse spiking neural computation," in *2022 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, pp. 317–330, 2022.
- [16] Y. Chen, Y. Liu, W. Ye, and C.-C. Chang, "The high-performance design of a general spiking convolution computation unit for supporting neuromorphic hardware acceleration," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 70, pp. 3634–3638, Sept. 2023.
- [17] W. Fang, Y. Chen, J. Ding, Z. Yu, T. Masquelier, D. Chen, L. Huang, H. Zhou, G. Li, and Y. Tian, "SpikingJelly: An open-source machine learning infrastructure platform for spike-based intelligence," *Science Advances*, vol. 9, p. eadi1480, Oct. 2023.
- [18] I. Aliyev, J. Lopez, and T. Adegbiya, "Exploring the sparsity-quantization interplay on a novel hybrid SNN event-driven architecture," in *2025 Design, Automation & Test in Europe Conference (DATE)*, pp. 1–7, Mar. 2025.

- [19] W. Ye, S. Chen, H. Liu, Y. Liu, Y. Chen, Y. Cui, and W. Lin, “The architecture design and training optimization of spiking neural network with low-latency and high-performance for classification and segmentation,” *Neural Networks*, vol. 191, p. 107790, 2025.
- [20] H. Zhao, H. Wu, D. Yang, A. Zou, and J. Hong, “Brillm: Brain-inspired large language model,” *arXiv preprint arXiv:2503.11299*, 2025.