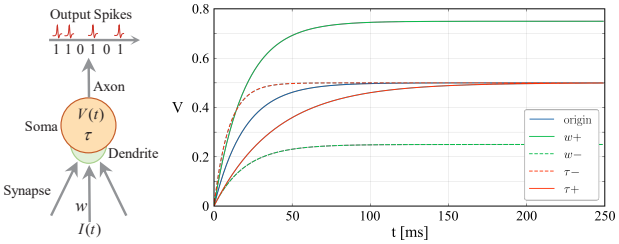
Incorporating Learnable Membrane Time Constant to Enhance Learning of Spiking Neural Networks

结合可学习的膜时间常数来增强脉冲神经网络的学习

方维 [1,2]、余肇飞 [1,2,3*]、陈彦骐 [1,2]、Timothée Masquelier[4]、黄铁军 [1,2,3]、田永鸿 [1,2*]
[1] 北京大学计算机科学与技术系
[2] 鹏城实验室，中国
[3] 北京大学人工智能研究院
[4] 认知研究中心 (CERCO)，UMR5549 CNRS - 图卢兹第三大学，法国
{fwei, yuzf12, chyq}@pku.edu.cn, timothee.masquelier@cnrs.fr, {tjhuang, yhtian}@pku.edu.cn

Abstract

脉冲神经网络（*SNN*）由于时间信息处理能力、低功耗和高生物学合理性而吸引了巨大的研究兴趣。然而，为 *SNN* 制定高效、高性能的学习算法仍然具有挑战性。大多数现有的学习方法仅学习权重，并且需要手动调整决定单个脉冲神经元动态的膜相关参数。这些参数通常选择对所有神经元都相同，这限制了神经元的多样性，从而限制了所得 *SNN* 的表达能力。在本文中，我们从不同大脑区域的膜相关参数不同的观察中得到启发，提出了一种训练算法，该算法不仅能够学习 *SNN* 的突触权重，还能够学习膜时间常数。我们证明，结合可学习的膜时间常数可以使网络对初始值不太敏感，并可以加快学习速度。此外，我们重新评估了 *SNN* 中的池化方法，发现最大池化不会导致显着的信息丢失，并且具有计算成本低和二进制兼容性的优点。我们在传统的静态图像分类任务上评估了所提出的方法 *MNIST*、*Fashion-MNIST*、*CIFAR-10* 数据集和神经形态 *N-MNIST*、*CIFAR10-DVS*、*DVS128* 手势数据集。实验结果表明，所提出的方法在几乎所有数据集上都优于最先进的精度，并且使用更少的时间步长。我们的代码可在 *https://github.com/fangwei123456/Parametric-Leaky-Integrate-and-Fire-Spiking-Neuron* 处获取。



(a) Spiking neu- (b) The membrane potential of a LIF neuron ron

Figure 1. (a) 具有膜电位 V、膜时间常数 τ、输入 $I(t)$ 和突触权重 w 的泄漏积分激发 (LIF) 神经元。(b) 接收恒定输入时 LIF 神经元的膜电位 V。增加或减少 τ 将在 t 方向拉伸 $v = f(t)$ 曲线，而增加或减少 w 将在 V 方向拉伸 $v = f(t)$ 曲线。

1. 介绍

脉冲神经网络（SNN）被视为第三代神经网络模型，它更接近于大脑中的生物神经元 [42]。与神经元和突触状态一起，SNN 中还考虑了脉冲时序的重要性。由于其独特的特性，例如时间信息处理能力、低功耗 [55]和高生物学合理性 [18]，近年来，SNN 越来越引起研究者的极大兴趣。然而，为 SNN 制定高效且高性能的学习算法仍然具有挑战性。

一般来说，SNN 的学习算法可分为无监督学习、监督学习、基于奖励的学习以及人工神经网络 (ANN) 到 SNN 的转换方法。不管怎样，我们发现大多数现有的学习方法只考虑学习突触相关的参数，如突触权重，并将膜相关的参数视为超参数。这些与膜相关的参数，如膜时间常数、决定单个脉冲神经元动力学的参数通常被选择为对所有神经元都相同。然而，请注意，大脑区域 [43, 9, 33] 的脉冲神经元存在不同的膜时间常数，这被证明对于工作记忆的表示和学习 [22, 59] 的制定至关重要。因此，简单地忽略 SNN 中的不同时间常数将限制神经元的异质性，从而限制最终 SNN 的表达能力。

在本文中，我们提出了一种训练算法，它不仅能够学习 SNN 的突触权重，还能够学习膜时间常数。如图1所

*通讯作者

示，我们发现突触权重和膜时间常数的调整对神经元动力学有不同的影响。我们证明，结合可学习的膜时间常数能够增强 SNN 的学习能力。

本文的主要贡献可概括如下：

1) 我们提出了基于反向传播的学习算法，使用具有可学习膜参数的脉冲神经元，称为参数泄漏积分激发（PLIF）脉冲神经元，它更好地表示神经元的异质性，从而增强 SNN 的表达能力。我们证明，由 PLIF 神经元组成的 SNN 对初始值更加稳健，并且比由具有固定时间常数的神经元组成的 SNN 学习得更快。
2) 我们重新评估了 SNN 中的池化方法，并驳斥了之前的结论，即最大池化会导致重大信息丢失。我们发现，与平均池化相比，最大池化能够更好地保留神经元放电的异步特性，并降低计算成本。我们的实验表明，最大池化的性能与平均池化相当。
3) 我们在 ANN 中广泛使用的传统静态 MNIST [35]、Fashion-MNIST [67]、CIFAR-10 [34] 数据集和神经形态 N-MNIST [50]、CIFAR10-DVS [39] 数据集上评估我们的方法，DVS128 Gesture [1] 数据集，重点验证网络的时间信息处理能力。所提出的方法使用更少的时间步长，在几乎所有测试的数据集上都超过了最先进的精度。

2. 相关作品

Unsupervised learning of SNNs SNN 的无监督学习方法基于生物学上合理的局部学习规则，例如 Hebbian 学习 [24] 和 Spike-Timing-Dependent Plasticity (STDP) [3]。现有方法利用了自组织原理 [62, 12, 31] 和基于 STDP 的期望最大化算法 [49, 19]。然而，这些方法仅适用于浅层 SNN，并且性能远低于最先进的 ANN 结果。

Reward-based learning of SNNs 基于奖励的 SNN 学习模仿人脑的学习方式，利用多巴胺能、血清素能、胆碱能或肾上腺素能神经元 [15, 6, 45] 诱导的奖励或惩罚信号。尽管强化学习中出现了策略梯度 [58, 30]、时差学习 [52, 16]和 Q 学习 [6]等方法，但最近提出了一些基于 STDP [17, 73]的启发式现象学模型。

ANN to SNN conversion ANN 到 SNN 的转换 (ANN2SNN) 通过使用每个脉冲神经元的放电率来近似模拟神经元 [26, 7, 56] 的相应 ReLU 激活，将经过训练的非脉冲 ANN 转换为 SNN。它可以像 ANN [57, 11] 一样获得近乎无损的推理结果，但在准确性和延迟之间需要权衡。为了提高准确性，需要更长的推理延迟 [21]。ANN2SNN 仅限于速率编码，从而失去了时态任务的处理能力。据我们所知，ANN2SNN 仅适用于静态数据集，不适用于神经形态数据集。

Supervised learning of SNNs SpikeProp [5] 是第一个基于反向传播的 SNN 监督学习方法，它使用线性逼近来克服 SNN 的不可微阈值触发机制。随后的工作包括 Tempotron [20]、ReSuMe [51] 和 SPAN [44]，但它们只能应用于单层 SNN。最近，代理梯度法被提出，为训练多层 SNN [38, 28, 75, 63, 60, 37, 29]提供了另一种解决方案。它利用代理导数来定义阈值触发触发机制的导数。因此，SNN 可以像 ANN 一样使用梯度下降算法进行优化。曾克等人。[74, 46] 系统地研究了替代梯度学习的显着鲁棒性，并表明通过替代梯度方法优化的 SNN 可以实现与 ANN 竞争的性能。与 ANN2SNN 相比，代理梯度法对模拟时间步没有限制，因为它不是基于速率编码 [64, 74]。

Spiking neurons and layers of deep SNNs 脉冲神经元和层模型在 SNN 中发挥着重要作用。程等人。[8]增加了相邻神经元之间的横向相互作用，获得了更好的精度和更强的噪声鲁棒性。齐默等人。[76]首次采用 LIF 神经元中可学习的时间常数进行语音识别任务。贝莱克等人。[2]提出了自适应阈值脉冲神经元来增强 SNN 的计算和学习能力，并通过具有可学习时间常数的 [69]进行了改进。拉蒂等人。[53] 建议使用可学习的膜泄漏和激发阈值来微调从 ANN 转换而来的 SNN。尽管如此，迄今为止尚未对学习膜时间常数对 SNN 的影响进行系统的研究，这正是本文的目的。吴等人。[64] 发现归一化层对于深度 SNN 也至关重要，并提出了神经元归一化 (NeuNorm) 来平衡每个神经元的放电率，以避免严重的信息丢失。莱迪瑙斯卡斯，E 等人。[36] 首先建议在深度 SNN 中使用 Batch Normalization [27] 来实现更快的收敛。

3. 方法

在本节中，我们首先简要回顾 3.1中的 Leaky Integrate-and-Fire 模型，并分析 3.2中突触权重和膜时间常数的影响。然后在 3.3 – Sec. 3.5 中介绍了参数化 Leaky Integrate-and-Fire 模型和 SNN 的网络结构。最后，我们在 3.6和 3.7中描述了峰值最大池化和 SNN 的学习算法。

3.1. 泄漏集成和激发模型

SNN 的基本计算单元是脉冲神经元。神经科学家建立了几种脉冲神经元模型来描述生物神经元输入和输出信号之间的准确关系。Leaky Integrate-and-Fire (LIF) 模型 [18] 是 SNN 中使用的最简单的脉冲神经元模型之一。LIF 神经元的阈下动力学定义为：

$$\tau \frac{\mathrm{d}V(t)}{\mathrm{d}t} = -(V(t) - V_{rest}) + X(t), \tag{1}$$

其中 $V(t)$ 表示神经元在时间 t 的膜电位，$X(t)$ 表示神经元在时间 t 的输入，τ 是膜时间常数，V_{rest} 是静息电位。当膜电位 $V(t)$ 在时间 t^f 超过某个阈值 V_{th} 时，神经元将引发脉冲，然后膜电位 $V(t)$ 返回到重置值 $V_{reset} < V_{th}$。LIF 神经元在计算成本和生物学合理性之间实现了平衡。我们在本文中设置为 $V_{rest} = V_{reset}$，并且在本文的其余部分中不会对它们进行区分。

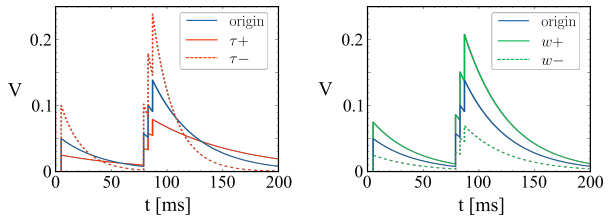


Figure 2. 当接收到 $t = 5, 80, 85, 90$ 的瞬时脉冲时，LIF 神经元的膜电位 V。

3.2. 突触重量与膜时间常数的函数比较

在大多数由 LIF 神经元组成的 SNN 的先前学习算法中，膜时间常数 τ 被视为超参数，并在学习之前选择对所有神经元相同。SNN 的学习只是为了优化突触权重。然而，不可忽视的是，给定输入的脉冲神经元的行为不仅取决于连接的突触的权重，还取决于由膜时间常数 τ 控制的神经元的固有动态。

为了比较突触权重和膜时间常数对神经元动力学的影响，我们考虑一个简单的情况，其中 LIF 神经元 z_i 从突触前神经元 z_j 接收加权输入 $X(t) = wI(t)$（图 1(a)）。其余电势 V_{rest} 设置为 0。当输入一定时，即 $I(t) = I$，LIF 神经元 z_i 的膜电位随时间的变化如图 1(b)（蓝色曲线）所示，这是根据方程 （1）计算的。增加或减少 w（如 $w+$ 和 $w-$ 曲线所示）将沿 V 方向拉伸 $v = f(t)$ 曲线。相反，增加或减少 τ 会沿 t 方向拉伸 $v = f(t)$ 曲线，并且不会像 $V(+\infty) = wI$ 那样改变神经元 z_i 的稳态电压。图 2说明了神经元 z_i 对 $t = \{5, 80, 85, 90\}$ms 时刻的瞬时输入脉冲的响应，即 $X(t) = w(\delta(t-5) + \delta(t-80) + \delta(t-85) + \delta(t-90))$ [1]。神经元对 $t = 5$ 处的瞬时输入脉冲的响应表明，较小的 τ（$\tau-$ 曲线）会导致更快地充电到稳态电压并更快地衰减到静止值，从而使 LIF 神经元对瞬时脉冲更加敏感。这种敏感性有助于神经元捕捉输入的即时变化。相反，较小的 w（$w-$ 曲线）会导致达到稳态电压的充电速度较慢，但不会影响衰减速度。当存在三个连续的输入脉冲时，具有较小 τ（$\tau-$ 曲线）的神经元的膜电位将以更快的速率达到更高的值，这使得它更容易放电。

在某种程度上，减少 τ 的效果与增加 w 的效果类似。然而，调整 τ 和 w 可以带来一些优越的额外好处。如上所述，同时改变 τ 和 w 可以在 t 方向和 V 方向拉伸 $v = f(t)$ 曲线，即神经元对给定输入的响应，从而赋予神经元更好的拟合能力。

3.3. 参数化泄漏积分与激发模型

我们提出了参数泄漏积分激发 (PLIF) 脉冲神经元模型来学习 SNN 的突触权重和膜时间常数。PLIF 神经元的动力学可以用方程 (1) 来描述。

具有 PLIF 神经元的 SNN 遵循三个规则：

[1] $\delta(t)$ 表示狄拉克 delta 函数。当 $x \neq 0$ 时，$\delta(t) = 0$。$\int_{-\infty}^{\infty} \delta(t)\, dt = 1$。

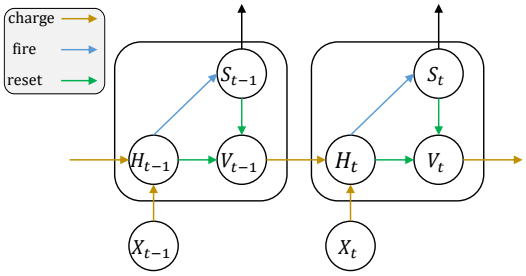


Figure 3. 一般离散脉冲神经元模型。

(1). 膜时间常数 τ 在训练过程中自动优化，而不是在训练前手动设置为超参数。

（2）。膜时间常数 τ 在 SNN 中同一层的神经元内共享，这在生物学上是合理的，因为相邻神经元具有相似的属性。

（3）。不同层神经元的膜时间常数 τ 不同，使得神经元的相频响应性不同。

事实上，所提出的规则能够增加神经元的异质性和所得 SNN 的表达能力，同时有效控制计算成本。

对于 SNN 中 PLIF 神经元的数值模拟，我们需要考虑时间离散的参数动态版本。具体来说，通过包括阈值触发的放电机制和放电后膜电位的重置，我们可以用以下方程描述各种脉冲神经元的动力学：

$$H_t = f(V_{t-1}, X_t), \tag{2}$$

$$S_t = \Theta(H_t - V_{th}), \tag{3}$$

$$V_t = H_t\,(1 - S_t) + V_{reset}\,S_t. \tag{4}$$

为了避免混淆，我们使用 H_t 和 V_t 分别表示神经元动力学后和时间步长 t 脉冲触发后的膜电位。X_t 表示外部输入，V_{th} 表示触发阈值。S_t 表示 t 时刻的输出脉冲，如果有脉冲则等于 1，否则等于 0。方程 (3) 描述了脉冲生成过程，其中 $\Theta(x)$ 是 Heaviside 阶跃函数，对于 $x \geq 0$ 由 $\Theta(x) = 1$ 定义，对于 $x < 0$ 由 $\Theta(x) = 0$ 定义。方程 (4) 说明在引发脉冲后膜电位返回到 V_{reset}，这被称为 *hard reset* 并广泛用于深度 SNN [36]。

如图 3所示，方程 (2) - (4) 建立了一个通用模型来描述离散脉冲神经元的动作：充电、激发和重置。具体来说，方程 (2) 描述了神经元动力学，不同的脉冲神经元模型具有不同的函数 $f(\cdot)$。例如，LIF 神经元和 PLIF 神经元的函数 $f(\cdot)$ 为

$$H_t = V_{t-1} + \frac{1}{\tau}(-(V_{t-1} - V_{reset}) + X_t). \tag{5}$$

对于 PLIF 神经元，直接优化方程 (5) 中的膜时间常数 τ 可能会导致数值不稳定，因为 τ 位于分母中。此外，方程 (5) 作为方程 (1) 的离散版本，只有当时间步长 $\mathrm{d}t$ 小于 τ，即 $\tau > 1$ 时，才是有效的近似，它被忽略 [53, 69]。为了避免上述问题，我们使用可训练参数 a 将方程 (5) 重新表述为以下方程：

$$H_t = V_{t-1} + k(a)(-(V_{t-1} - V_{reset}) + X_t). \tag{6}$$

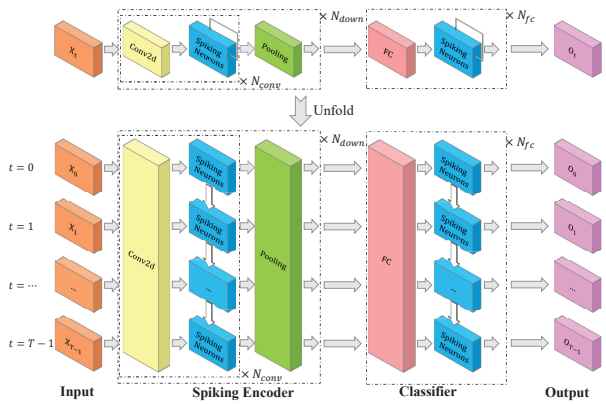


Figure 4. 我们网络的一般表述及其展开表述。$\times N_{conv}$ 表示有 N_{conv} *{Conv2d-Spiking Neurons}* 顺序连接。$\times N_{down}$ 和 $\times N_{fc}$ 含义相同。请注意，网络参数在所有时间步长都是共享的。

这里 $k(a)$ 表示钳位功能，$k(a) \in (0,1)$ 表示钳位功能，这确保了 $\tau = \frac{1}{k(a)} \in (1, +\infty)$。在我们的实验中，$k(a)$ 是 sigmoid 激活函数，即 $k(a) = \frac{1}{1+\exp(-a)}$。

3.4. LIF 和 PLIF 的类似 RNN 的表达

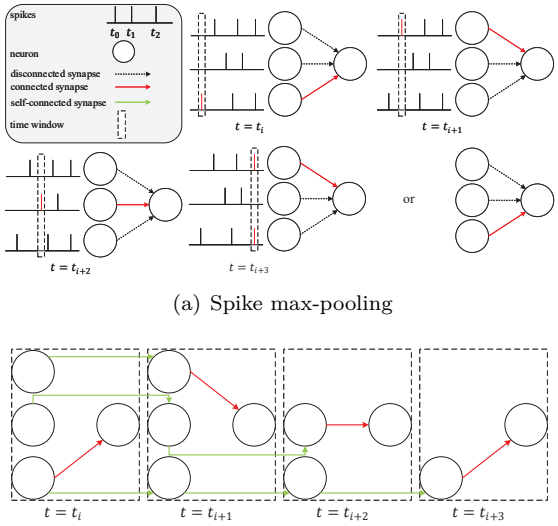
LIF 和 PLIF 神经元具有与循环神经网络类似的功能。具体来说，当 $V_{reset} = 0$ 时，LIF 神经元和 PLIF 神经元的神经元动力学（Eq. (5)）可以写为：

$$H_t = \left(1 - \frac{1}{\tau}\right) V_{t-1} + \frac{1}{\tau} X_t, \tag{7}$$

其中整合进度 $\frac{1}{\tau} X_t$ 使 LIF 和 PLIF 神经元能够记住当前输入信息，而泄漏进度 $(1 - \frac{1}{\tau}) V_{t-1}$ 可以被视为忘记了过去的一些信息。方程 (7) 表明，记忆和遗忘之间的平衡是由膜时间常数 τ 控制的，它起着与长短期记忆 (LSTM) 网络 [25]中的门类似的作用。

3.5. 网络制定

我们在本文中提出了构建 SNN 的通用公式，如图 4 所示。SNN 包括脉冲编码器网络和分类器网络。脉冲编码器网络由 N_{down} 下采样模块组成，每个模块包含 N_{conv} 重复的 *{Conv2d-Spiking Neurons}* 和池化层。脉冲编码器可以从输入中提取特征并将其转换为不同时间步长的放电脉冲。分类器网络由 N_{fc} 重复的 *{FC-Spiking Neurons}* 组成。这里 *Conv2d* 表示 2D 卷积层，*FC* 表示全连接层。之前的许多工作 [12, 37, 60, 75, 8, 21] 使用 Poisson 编码器将图像转换为脉冲作为输入，而 [56] 表明这种编码会给网络的发射带来可变性并损害其性能。与 [56, 64, 53] 类似，输入直接馈送到我们的网络，而无需首先转换为脉冲。在这种情况下，图像脉冲编码由第一个 *{Conv2d-Spiking Neurons}* 模块完成，该模块可以被视为可学习的编码器。请注意，突触连接（包括卷积层和全连接层）是无状态的，而脉冲神经元层在时域中具有自连接，如图 4所示的展开网络公式。所有参数在所有时间步共享。



(a) Spike max-pooling

(b) Unfolded computation graph

Figure 5. Spike max-pooling 动态调节连接。(a) 具有脉冲最大池化的三个突触前神经元和一个突触后神经元的示例。在每个时间步，只有发出脉冲的神经元才能连接到突触后神经元。当多个神经元同时放电时，会随机选择可以连接到突触后神经元的神经元。(b) (a) 的展开计算图

3.6. 峰值最大池化

池化层广泛用于减小特征图的大小并提取卷积 ANN 和 SNN 中的紧凑表示。之前的大多数研究 [57, 8, 54] 更喜欢在 SNN 中使用平均池化，因为他们发现 SNN 中的最大池化会导致显着的信息丢失。我们认为最大池化与 SNN 的时间信息处理能力一致，可以提高 SNN 在时间任务中的拟合能力并降低下一层的计算成本。

具体来说，在我们的模型中，最大池化层位于脉冲神经元层后面（图 4），并且最大池化操作是在脉冲上进行的。与平均池化窗口中所有神经元均等地将信息传输到下一层不同，只有在最大池化窗口中激发脉冲的神经元才能将信息传输到下一层。因此，最大池化层引入了赢家通吃机制，允许被激发的神经元与下一层进行通信，并忽略池化窗口中的其他神经元。另一个有吸引力的特性是最大池层将动态调节连接（图 5）。脉冲神经元的膜电位 V_t 在发射脉冲后将返回到 V_{reset}。由于充电需要时间，脉冲神经元很难再次放电。然而，如果最大池化窗口中的神经元异步激发，它们将依次连接到突触后神经元，这使得突触后神经元类似于连接连续激发的突触前神经元并且更容易激发。通过最大池化实现的空间域赢家通吃机制和时间域时变拓扑可以提高 SNN 在时间任务中的拟合能力，例如对 CIFAR10-DVS 数

据集进行分类。值得注意的是，最大池化层的输出仍然是二进制的，而平均池化层的输出是浮点型的。通过用 *logical AND* & 替换 *multiplication* $*$ 可以加速脉冲上的矩阵乘法和逐元素乘法运算，这也是 SNN 相对于 ANN 的优势。

3.7. 培训框架

在这里，我们结合神经元模型（图 3）和网络公式（图 4）来驱动 SNN 的反向传播训练算法。将模拟时间步表示为 T，类编号表示为 C，输出 $\boldsymbol{O} = [o_{t,i}]$ 是 $C \times T$ 张量。对于带有标签 l 的给定输入，我们鼓励代表 l 类的神经元具有最高的兴奋水平，而其他神经元应保持沉默。因此，目标输出由 $\boldsymbol{Y} = [y_{t,i}]$ 定义，其中 $i = l$ 为 $y_{t,i} = 1$，$i \neq l$ 为 $y_{t,i} = 0$。损失函数由均方误差 (MSE) $L = MSE(\boldsymbol{O}, \boldsymbol{Y}) = \frac{1}{T}\sum_{t=0}^{T-1} L_t = \frac{1}{T}\sum_{t=0}^{T-1}\frac{1}{C}\sum_{i=0}^{C-1}(o_{t,i} - y_{t,i})^2$ 定义。预测标签 l_p 被视为具有最大放电率 $l_p = \arg\max_i \frac{1}{T}\sum_{t=0}^{T-1} o_{t,i}$ 的神经元的索引。

这里我们假设 a^i 表示网络中第 i 层 PLIF 神经元的可学习参数。在时间步 t，向量 $\boldsymbol{H}_t^i$ 和 $\boldsymbol{V}_t^i$ 代表神经元动力学之后的膜电位，并且在复位之后，向量 $\boldsymbol{V}_{th}^i$ 和 $\boldsymbol{V}_{reset}^i$ 分别代表阈值和复位电位。上一层的加权输入是 $\boldsymbol{X}_t^i = \boldsymbol{W}^{i-1}\boldsymbol{I}_t^i$。$\boldsymbol{S}_t^i = [s_{t,j}^i]$ 表示时间步长 t 的输出脉冲，其中如果第 j 个神经元触发脉冲，则为 $s_{t,j}^i = 1$，否则为 $s_{t,j}^i = 0$。从下一层向后的梯度是 $\frac{\partial L_t}{\partial \boldsymbol{S}_t^i}$。根据图 3和图 4，我们可以递归地计算梯度：

$$\frac{\partial L}{\partial \boldsymbol{H}_t^i} = \frac{\partial L}{\partial \boldsymbol{H}_{t+1}^i}\frac{\partial \boldsymbol{H}_{t+1}^i}{\partial \boldsymbol{H}_t^i} + \frac{\partial L_t}{\partial \boldsymbol{H}_t^i} \tag{8}$$

$$\frac{\partial \boldsymbol{H}_{t+1}^i}{\partial \boldsymbol{H}_t^i} = \frac{\partial \boldsymbol{H}_{t+1}^i}{\partial \boldsymbol{V}_t^i}\frac{\partial \boldsymbol{V}_t^i}{\partial \boldsymbol{H}_t^i} \tag{9}$$

$$\frac{\partial L_t}{\partial \boldsymbol{H}_t^i} = \frac{\partial L_t}{\partial \boldsymbol{S}_t^i}\frac{\partial \boldsymbol{S}_t^i}{\partial \boldsymbol{H}_t^i} \tag{10}$$

根据式 (6)、式 (3)、式 (4) 可得

$$\frac{\partial \boldsymbol{H}_{t+1}^i}{\partial \boldsymbol{V}_t^i} = 1 - k(a^i) \tag{11}$$

$$\frac{\partial \boldsymbol{V}_t^i}{\partial \boldsymbol{H}_t^i} = 1 - \boldsymbol{S}_t + (\boldsymbol{V}_{reset}^i - \boldsymbol{H}_t^i)\frac{\partial \boldsymbol{S}_t^i}{\partial \boldsymbol{H}_t^i} \tag{12}$$

$$\frac{\partial \boldsymbol{S}_t^i}{\partial \boldsymbol{H}_t^i} = \Theta'(\boldsymbol{H}_t^i - \boldsymbol{V}_{th}^i) \tag{13}$$

$$\frac{\partial \boldsymbol{H}_t^i}{\partial \boldsymbol{X}_t^i} = k(a^i) \tag{14}$$

$$\begin{aligned}\frac{\partial \boldsymbol{H}_t^i}{\partial a^i} = {} & (-(\boldsymbol{V}_{t-1}^i - \boldsymbol{V}_{reset}^i) + \boldsymbol{X}_t^i)k'(a^i) \\ & + \frac{\partial \boldsymbol{H}_t^i}{\partial \boldsymbol{V}_{t-1}^i}\frac{\partial \boldsymbol{V}_{t-1}^i}{\partial \boldsymbol{H}_{t-1}^i}\frac{\partial \boldsymbol{H}_{t-1}^i}{\partial a^i}\end{aligned} \tag{15}$$

Dataset	N_{conv}	N_{down}	N_{fc}
*MNIST	1	2	2
CIFAR-10	3	2	2
CIFAR10-DVS	1	4	2
DVS128 手势	1	5	2

Table 1. 不同数据集的网络结构。N_{conv}、N_{down} 和 N_{fc} 的定义如图 4 所示。**MNIST* 表示 MNIST、Fashion-MNIST 和 N-MNIST 数据集。

最后，我们可以得到可学习参数的梯度：

$$\frac{\partial L}{\partial a^i} = \sum_{t=0}^{T-1}\frac{\partial L}{\partial \boldsymbol{H}_t^i}\frac{\partial \boldsymbol{H}_t^i}{\partial a^i} \tag{16}$$

$$\frac{\partial L}{\partial \boldsymbol{W}^{i-1}} = \sum_{t=0}^{T-1}\frac{\partial L}{\partial \boldsymbol{H}_t^i}\frac{\partial \boldsymbol{H}_t^i}{\partial \boldsymbol{X}_t^i}\boldsymbol{I}_t^i \tag{17}$$

注意当 $t \geq T$、$\boldsymbol{V}_{-1}^i = \boldsymbol{V}_{reset}^i$ 时为 $\frac{\partial *}{\partial \boldsymbol{S}_t^i} = 0$。我们使用代理函数 $\sigma(x)$ 的导数来定义脉冲函数 $\Theta(x)$ 的导数（参见补充）。$k(x)$ 为钳位功能。

4. 实验

我们评估了具有 PLIF 神经元和脉冲最大池化的 SNN 在传统静态 MNIST、Fashion-MNIST、CIFAR-10 数据集以及神经形态 N-MNIST、CIFAR10-DVS 和 DVS128 手势数据集上的分类任务的性能。有关培训的更多详细信息，请参阅补充材料。

4.1. 网络结构

不同数据集的 SNN 网络结构如表 1所示。我们为所有 *Conv2d* 层设置 *kernel size* $= 3$、*stride* $= 1$ 和 *padding* $= 1$。对于 CIFAR-10 数据集，*Conv2d* 图层的 *out channels* 为 256，对于所有其他数据集，*out channels* 图层为 128。在每个 *Conv2d* 层之后添加批量归一化 (*BN*) 层。由于 *BN* 层的参数可以吸收到其前面的 *Conv2d* 层 [56] 中，因此我们可以在 SNN 中删除 *BN* 进行推理。所有池化层均设置 *kernel size* $= 2$ 和 *stride* $= 2$。对于所有网络，第一个 *FC* 层的 *out features* 是 *in features* 的四分之一，第二个 *FC* 层的 *out features* 是 $M \cdot C$，其中 C 是类别编号，M 是要表示的群体的神经元一堂课。在每个 *FC* 层之前放置一个 dropout 层 [37]。输出脉冲神经元层之后的投票层用于提高分类的鲁棒性。投票层通过 *kernel size* $= M$ 和 *stride* $= M$ 的平均池化实现。我们为所有数据集设置 $M = 10$。我们采用平均池的方式实行民主投票，少数服从多数。使用最大池化投票可能会导致独裁，因为少数人不会参与计算图（见图 5），并且使用 M 神经元代表一类将退化为使用一个神经元。

Model	Method	Accuracy MNIST	Accuracy Fashion-MNIST	Accuracy CIFAR-10	Accuracy N-MNIST	Accuracy CIFAR10-DVS	Accuracy DVS128 Gesture
[26]	ANN2SNN	98.37%	-	82.95%	-	-	-
[56]	ANN2SNN	99.44%	-	88.82%	-	-	-
[57]	ANN2SNN	-	-	91.55%	-	-	-
[21]	ANN2SNN	-	-	**93.63%**	-	-	-
[38]	基于脉冲的 BP	99.31%	-	-	98.74%	-	-
[63]	基于脉冲的 BP	99.42%	-	-	98.78%	50.7%	-
[60]	基于脉冲的 BP	99.36%	-	-	99.2%	-	93.64%
[29]	基于脉冲的 BP	-	-	-	96%	-	**95.54%**
[28]	基于脉冲的 BP	99.49%	-	-	98.84%	-	-
[75]	基于脉冲的 BP	**99.62%**	90.13%	-	-	-	-
[64]	基于脉冲的 BP	-	-	90.53%	**99.53%**	**60.5%**	-
[37]	基于脉冲的 BP	99.59%	-	90.95%	99.09%	-	-
[8]	基于脉冲的 BP	99.5%	**92.07%**	-	99.45%	-	-
[40]	基于脉冲的 BP	-	-	-	96.3%	32.2%	-
[68]	基于脉冲的 BP	-	-	-	-	-	92.01%
[13]	基于脉冲的 BP	99.46%	-	-	99.39%	-	96.09% (10 classes)
[23]	基于脉冲的 BP	-	-	-	98.28%	-	93.40%
[53]	ANN2SNN and Spike-based BP	-	-	92.64%	-	-	-
[61]	帽子	-	-	-	99.1%	52.4%	-
[4]	GCN	-	-	-	99.0%	54.0%	-
我们的	基于脉冲的 BP	99.72%	94.38%	93.50%	99.61%	74.80%	97.57%

Table 2. 所提出的方法与最先进的方法在不同数据集上的性能比较。以前作品的最高准确度以粗体显示。

Dataset	SOTA	SOTA's T	ours T
MNIST	[75]	400	8
时尚-MNIST	[8]	20	8
CIFAR-10	[21]	2048	8
N-MNIST	[64]	59-64	10
CIFAR10-DVS	[64]	230-292	20
DVS128 手势	[29]	500(training) 1800(testing)	20

Table 3. 之前的 SOTA 和我们的 SOTA 在每个数据集上的时间步长均有效。

4.2. 与最先进的比较

Tab. 2 显示了所提出方法的准确性（具有 $\tau_0 = 2$ 的 PLIF 神经元，最大池化）以及传统静态 MNIST、Fashion-MNIST、CIFAR-10 数据集和神经形态 N-MNIST、CIFAR10-DVS、DVS128 手势数据集上的其他比较方法。我们为所有数据集设置相同的训练超参数（参见补充）。如 Tab. 2 所示，我们在除 CIFAR-10 之外的所有数据集上实现了最高准确度。CIFAR-10 上的精度略低于基于 ANN2SNN 转换的 [21]。然而，它们仅适用于静态图像，因为 ANN2SNN 不适合神经形态数据集。与它们不同的是，我们的方法也适用于神经形态数据集，并且优于基于脉冲的 BP SOTA 精度。

Tab. 3 比较了我们的方法和之前在每个数据集上实现最佳性能的方法的时间步数。可以发现，所提出的方法比所有其他方法花费更少的时间步。例如，与 ANN2SNN 转换 [21] 相比，我们的方法使用的推理时间步数最多为 256×。因此，我们的方法不仅可以减少内存消耗和训练时间，而且可以大大提高推理速度。

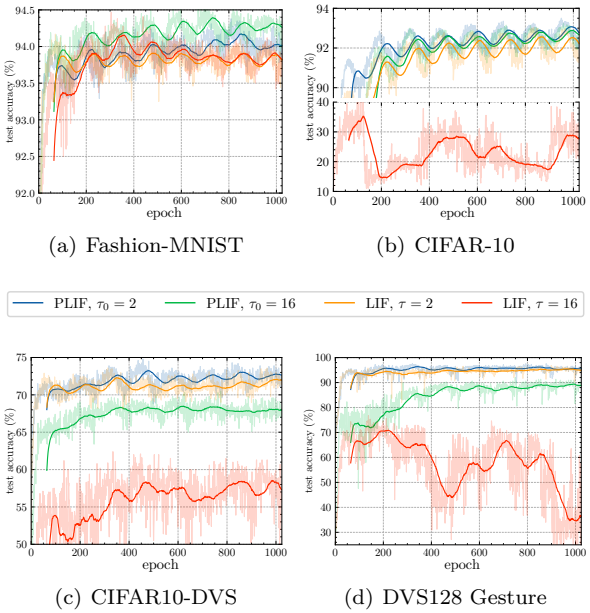
4.3. 消融研究

我们进行了广泛的消融研究，以评估 PLIF 神经元和四个具有挑战性的数据集上的最大池化。我们首先研究 PLIF 神经元的作用。在本实验中，我们分别用 PLIF 神经元和 LIF 神经元训练相同的 SNN，并比较测试精度。如表 4所示，如果将 PLIF 神经元的初始膜时间常数 τ_0 设置为等于 LIF 神经元的膜时间常数 τ，则带有 PLIF 神经元的 SNN 的测试精度始终高于带有 LIF 神经元的 SNN。这是由于不同层 PLIF 神经元在学习后的膜时间常数可以不同，这更好地代表了神经元的异质性。图 6 说明了训练期间 PLIF 与 LIF 神经元的测试准确性。可以看出，如果膜时间常数的初始值不合理（红色曲线），带有 LIF 神经元的 SNN 的精度和收敛速度会严重下降。相反，PLIF 神经元可以学习适当的膜时间常数并实现更好的性能（绿色曲线）。

为了分析初始值对 PLIF 神经元的影响，我们展示了每层神经元的膜时间常数在学习过程中相对于不同的初始值如何变化。如图 7所示，每层具有不同初始值的膜时间常数在训练过程中趋于聚集，这表明 PLIF 神经元对初始值具有鲁棒性。请注意，图 7(a) 中的 $\tau(6)$ 和图 7(b) 中的 $\tau(4)$ 趋于无穷大。这可以解释如下。两

Neuron	Fashion-MNIST	CIFAR-10	CIFAR10-DVS	DVS128 Gesture
PLIF($\tau_0 = 2$)	**94.38%**	**93.50%**	**74.80%**	**97.57%**
LIF($\tau = 2$)	94.17%	93.03%	73.60%	96.88%
PLIF($\tau_0 = 16$)	**94.65%**	**93.23%**	**70.50%**	**92.01%**
LIF($\tau = 16$)	94.47%	47.50%	62.40%	76.74%

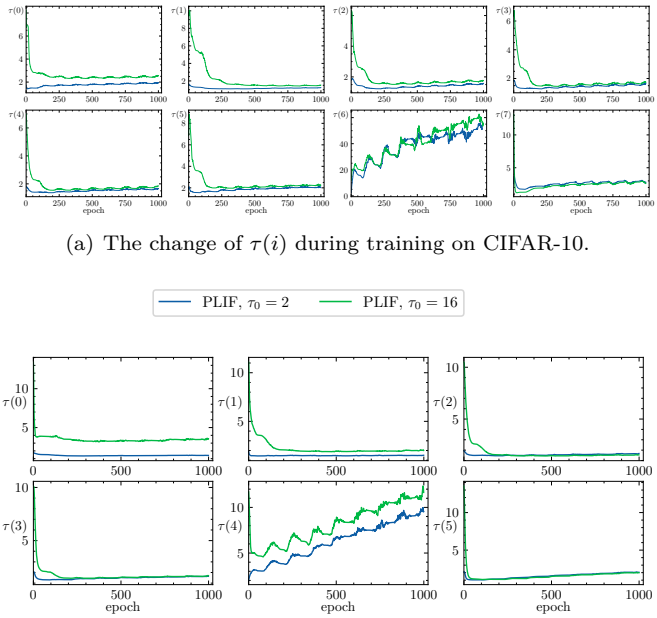
Table 4. 使用 PLIF/LIF 的准确性。



(a) Fashion-MNIST (b) CIFAR-10

(c) CIFAR10-DVS (d) DVS128 Gesture

Figure 6. 训练期间不同数据集上 PLIF 与 LIF 神经元的测试准确性。阴影曲线表示原始数据。实线是 64 纪元的移动平均线。



(a) The change of $\tau(i)$ during training on CIFAR-10.

(b) The change of $\tau(i)$ during training on CIFAR10-DVS.

Figure 7. 不同初始值训练时不同层膜时间常数的变化。$\tau(i)$ 表示第 i PLIF 神经元层的膜时间常数 τ。

Pooling	Fashion-MNIST	CIFAR-10	CIFAR10-DVS	DVS128 Gesture
平均值	**94.74%**	93.30%	72.70%	97.22%
最大	94.38%	**93.50%**	**74.80%**	**97.57%**

Table 5. 使用最大池化/平均池化的准确性。

个 SNN 中膜时间常数为 $\tau(4)$ 和 $\tau(6)$ 的 PLIF 神经元位于权重为 W_{fc} 的第一个 FC 层后面。我们检查训练日志，发现 $\frac{W_{fc}}{\tau}(\tau = \tau(4) \text{ or } \tau(6))$ 的分布、均值和方差在数十个 epoch 后收敛（见补充）。参考 PLIF 神经元的动力学 (Eq. (5)) 与 $X_t = W_{fc}I_t$ 和 $\frac{1}{\tau} \to 0$，我们可以找到 $H_t \to V_{t-1} + \frac{W_{fc}}{\tau}I_t$。这意味着第一个 FC 层之后的 PLIF 神经元正在学习成为非泄漏积分和激发神经元。

我们进一步研究最大池化的效果。Tab. 5 在四个具有挑战性的数据集上比较了所提出的 SNN 与最大池化/平均池化的准确性。最大池化的性能与平均池化的性能相似，这表明之前关于最大池化会导致 SNN 中显着信息丢失的结论是不合理的。值得注意的是，最大池化在 CIFAR-10、CIFAR10-DVS 和 DVS128 手势数据集上获得了略高的精度，显示了其在复杂任务中更好的拟合能力。

5. 结论

在这项工作中，我们提出了参数泄漏积分激发 (PLIF) 神经元，将可学习的膜时间参数合并到 SNN 中。我们证明，带有 PLIF 神经元的 SNN 在静态和神经形态数据集上的性能都优于最先进的比较方法。此外，我们还表明，由 PLIF 神经元组成的 SNN 对初始值更加稳健，并且比由 LIF 神经元组成的 SNN 学习得更快。我们还重新评估了 SNN 中最大池化和平均池化的性能，发现以前的工作低估了最大池化的性能。我们建议在 SNN 中使用最大池化，因为它具有更低的计算成本、更高的时间拟合能力，以及接收脉冲和输出脉冲的特性，而不是像平均池化那样使用浮动值。

6. 致谢

这项工作得到了国家自然科学基金委的资助，合同号为 62027804、61825101 和 62088102。

补充材料

A. 再现性

所有实验均由 SpikingJelly [14]实现。所有源代码、训练日志都可以在 GitHub 上找到。为了最大限度地提高再现性，我们在所有代码中使用相同的种子。

B. 网络结构详情

Tab. S1 说明了不同数据集的网络结构的详细信息。*c128k3s1* 表示与 *output channels* = 128、*kernel size* = 3 和 *stride* = 1 的卷积层。*BN* 是批量归一化。*MPk2s2* 是 *kernel size* = 2 和 *stride* = 2 的最大池化层。*PLIF* 是 PLIF 脉冲神经元层。*DP* 表示丢失层 [37]。*FC2048* 表示与 *output features* = 2048 的全连接层。符号 *{}** 表示重复结构。例如，*{c128k3s1-BN-PLIF-MPk2s2}*2* 表示有两个按顺序连接的 *{c128k3s1-BN-PLIF-MPk2s2}* 模块。最后一层 *APk10s10* 是投票层，由 *kernel size* = 10 和 *stride* = 10 的平均池化层实现。

C. 训练算法以适应目标输出

在定义了脉冲生成过程的导数后，可以像 ANN 中那样通过梯度下降算法来训练 SNN 的参数。分类是本文的任务，以及 ANN 和 SNN 的其他任务，可以看作是在给定特定输入时优化网络参数以适应目标输出。SNN 拟合目标输出的梯度下降算法在正文中导出（Eq. (16) 和 Eq. (17)），如下所示：

Dataset	Network Structure
*MNIST	{c128k3s1-BN-PLIF-MPk2s2}*2-DP-FC2048-PLIF-DP-FC100-PLIF-APk10s10
CIFAR-10	{{c256k3s1-BN-PLIF}*3-MPk2s2}*2-DP-FC2048-PLIF-DP-FC100-PLIF-APk10s10
CIFAR10-DVS	{c128k3s1-BN-PLIF-MPk2s2}*4-DP-FC512-PLIF-DP-FC100-PLIF-APk10s10
DVS128 手势	{c128k3s1-BN-PLIF-MPk2s2}*5-DP-FC512-PLIF-DP-FC110-PLIF-APk10s10

Table S1. 不同数据集的详细网络结构。**MNIST* 代表 MNIST、Fashion-MNIST 和 N-MNIST 数据集。

Algorithm S1 SNN 拟合目标输出的梯度下降算法

Require: 学习率 ϵ，网络参数 $\boldsymbol{\theta}$，总模拟时间步 T，输入 $\boldsymbol{X} = \{\boldsymbol{X}_0, \boldsymbol{X}_1, ..., \boldsymbol{X}_{T-1}\}$，目标输出 $\boldsymbol{Y} = \{\boldsymbol{Y}_0, \boldsymbol{Y}_1, ..., \boldsymbol{Y}_{T-1}\}$，损失函数 $L = \mathcal{L}(\boldsymbol{O}, \boldsymbol{Y})$
初始化 $\boldsymbol{\theta}$
创建一个空列表 $\boldsymbol{S} = \{\}$
for $t \leftarrow 0, 1, ...T-1$
  将 $\boldsymbol{X}_t$ 输入网络，得到输出脉冲 $\boldsymbol{S}_t$
  将 $\boldsymbol{S}_t$ 附加到 $\boldsymbol{S} = \{\boldsymbol{S}_0, \boldsymbol{S}_1, ..., \boldsymbol{S}_{t-1}\}$
计算损失 $L = \mathcal{L}(\boldsymbol{Y}, \boldsymbol{O})$
更新参数 $\boldsymbol{\theta} = \boldsymbol{\theta} - \epsilon \cdot \nabla_{\boldsymbol{\theta}} L$

这里的损失函数 $L = \mathcal{L}(\boldsymbol{O}, \boldsymbol{Y})$ 是 $\boldsymbol{Y}$ 和 $\boldsymbol{S}$ 之间的距离测量，例如正文中的均方误差（MSE）。

D. 数据集介绍

MNIST 手写数字的 MNIST 数据集包含 28×28 灰度图像，标记为 0 到 9。MNIST 数据集包含 60,000 张训练图像和 10,000 张测试图像。

时尚-MNIST 与 MNIST 数据集类似，Fashion-MNIST 数据集由 60,000 个示例的训练集和 10,000 个示例的测试集组成。Fashion-MNIST 数据集中的每个示例都是 28×28 灰度图像，标签为 0 到 9。

CIFAR-10 CIFAR-10 数据集由 10 个类别的 60,000 张自然图像组成，每个类别有 6,000 张图像。训练图像数量为 50,000 张，测试图像数量为 10,000 张。

N-MNIST Neuromorphic-MNIST (N-MNIST) 数据集是神经形态传感器记录的 MNIST 数据集的脉冲版本。它是从 MNIST 转换而来的，方法是将 ATIS 传感器安装在电动云台装置上，并移动传感器，同时在 LCD 监视器上记录 MNIST 示例。它由 60,000 个训练示例和 10,000 个测试示例组成。

CIFAR10-DVS CIFAR10-DVS 数据集是 CIFAR-10 数据集的神经形态版本。它由 10 个类别的 10,000 个示例组成，每个类别有 1000 个示例。由于 CIFAR10-DVS 数据集没有将数据分为训练集和测试集，因此在每个类别中，我们选择前 9000 个样本进行训练，其余 1000 个样本进行测试，这与 [64]类似。

DVS128 手势 DVS128 手势数据集由 DVS128 相机记录，包含 29 名受试者在 3 种光照条件下的 11 种手势。

E. 预处理

Static Datasets. 我们对所有静态数据集应用数据归一化，以确保输入图像的均值和单位方差为零。此外，在 MNIST 和 CIFAR-10 上进行随机水平翻转和裁剪以避免过度拟合。我们不在 Fashion-MNIST 上使用这些增强，因为该数据集中的图像很整洁。

Neuromorphic Datasets. 神经形态数据集中的数据通常采用地址事件表示（AER）$E(x_i, y_i, t_i, p_i)$（$i = 0, 1, ..., N-1$）的形式来表示异步流中的事件位置、时间戳和极性。由于事件数量很大，例如 CIFAR10-DVS 中超过一百万个事件，我们将事件分成 T 切片，每个切片中的事件数量几乎相同，并将事件集成到帧中。新的表示 $F(j, p, x, y)$ ($0 \leq j \leq T-1$) 是第 j 切片中事件数据的总和：

$$F(j,p,x,y) = \sum_{i=j_l}^{j_r-1} \mathcal{I}_{p,x,y}(p_i, x_i, y_i), \tag{S1}$$

其中 $\mathcal{I}_{p,x,y}(p_i, x_i, y_i)$ 是指示函数，仅当 $(p, x, y) = (p_i, x_i, y_i)$ 时才等于 1。j_l 和 j_r 是第 j 切片中的最小和最大时间戳索引。如果是 $j < T-1$，则为 $j_l = \lfloor \frac{N}{T} \rfloor \cdot j$、$j_r = \lfloor \frac{N}{T} \rfloor \cdot (j+1)$；如果是 $j = T-1$，则为 N。这里 $\lfloor \cdot \rfloor$ 是场内操作。请注意，T 也是我们实验中的时间步数。

用于预处理神经形态数据集的类似事件到帧集成方法广泛应用于 ANNs [23, 48, 47] 和 SNNs [64, 65, 8, 68, 23, 29, 38] 中。N-MNIST 和 CIFAR10-DVS 正文的 Tab. 3中的 [64]和 [64]是根据他们的论文手动计算的。具体来说，他们说明了通过在每 5 个 ms 内累积脉冲序列来降低时间分辨率，N-MNIST 和 CIFAR10-DVS 的时间范围（us）分别为 [290901, 315348] 和 [1149758, 1459301]。

F. 超参数

我们使用学习率为 0.001 的 Adam [32] 优化器和具有 $T_{schedule} = 64$ 的余弦退火学习率计划 [41]。*batch size* 设置为 16 以减少内存消耗。Dropout 层的丢弃概率 p 为 0.5。PLIF 神经元的钳位函数是 $k(a) = \frac{1}{1+e^{-a}}$，替代梯度函数是 $\sigma(x) = \frac{1}{\pi}\arctan(\pi x) + \frac{1}{2}$，因此是 $\sigma'(x) = \frac{1}{1+(\pi x)^2}$。我们为所有神经元设置 $V_{reset} = 0$ 和 $V_{th} = 1$。我们注意到之前的一些作品，例如 [63]、[64]，针对不同的任务对 V_{th} 进行了微调，这是不必要的。具体来说，由于 $\Theta(V - V_{th}) = \Theta(V_{th}(\frac{V}{V_{th}} - 1)) = \Theta(\frac{V}{V_{th}} - 1)$ 和 V 是直接受可训练权重影响的，因此设置 $V_{th} = 1$ 实现了权重的隐式归一化，可以缓解梯度爆炸和消失的问题。正如 Zenke 和 Vogels [74] 所发现的，通过将梯度从计算图中分离出来来计算梯度时忽略神经元重置可以提高性能，我们也在神经元重置中分离了 S_t。

Dataset	Without Validation	15% Validation
MNIST	99.72%	99.63%
时尚-MNIST	94.38%	93.85%
CIFAR-10	93.50%	92.58%
N-MNIST	99.61%	99.57%
CIFAR10-DVS	74.80%	69.00%
DVS128 手势	97.57%	96.53%

Table S2. 在不同数据集上有/没有验证集的情况下所提出的方法的准确性。

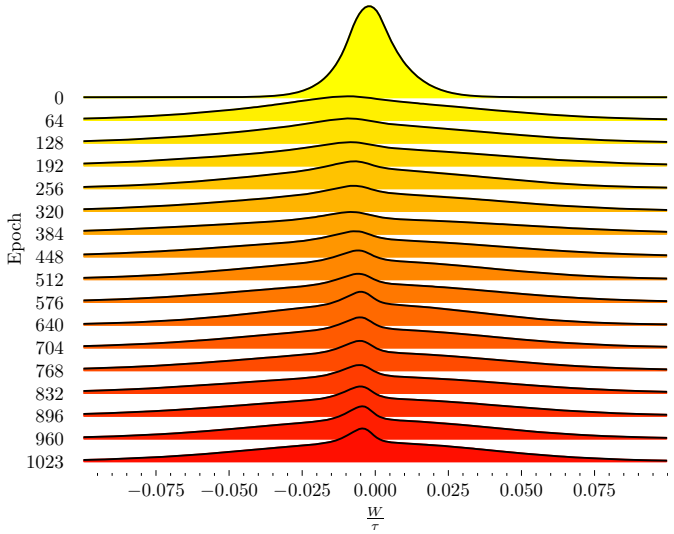


Figure S1. 在 CIFAR10-DVS 上训练 SNN 期间 $\frac{W_{fc}}{\tau}$ 的分布。

G. 验证集的准确性

正文 Tab. 2中的性能对比是通过在训练集上训练，在测试集上交替测试，记录最大测试精度得到的。本文和最先进的方法都使用这种方式来报告性能。然而，这种准确性被高估了。在这里，我们还报告了验证精度，这是通过将原始训练集拆分为新的训练集和验证集，在新训练集上进行训练，在验证集上交替进行测试，并使用达到最大验证精度的模型仅在测试集上记录一次测试精度来获得的。我们利用原始训练集中每个类别的 85 正文 Tab. 2 和 Tab. S2 中的实验结果表明，所提出的方法在几乎所有数据集上都优于最先进的精度。

H. 第一个 $\frac{W_{fc}}{\tau}$ 的分配

在正文的 Sec. 4.3 中，我们发现当 $\frac{1}{\tau} \to 0$ 和 $\frac{W_{fc}}{\tau}$ 收敛时，第一个 FC 层之后的 PLIF 神经元正在学习成为非泄漏积分和激发神经元。为了说明收敛性，我们在图 S1 中显示了在 CIFAR-10DVS 上训练 SNN 期间 $\frac{W_{fc}}{\tau}$ 的分布。$\frac{W_{fc}}{\tau}$ 在其他数据集上的分布以同样的方式收敛。

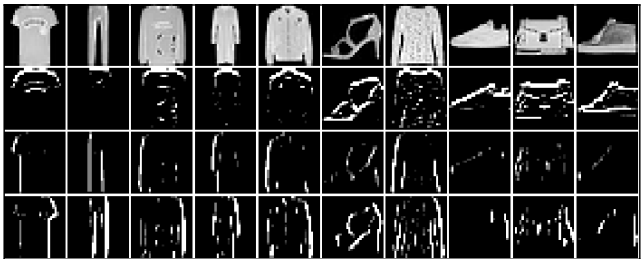


Figure S2. 第 1-4 行显示了来自 Fashion-MNIST 数据集的 10 个样本以及来自第一个 PLIF 神经元层的通道 45,75 和 76 ($c = 45, 75, 76$) 的相应放电率 $\boldsymbol{F}^2_{T_s=8}$。每列代表一个样本和相应的发射率。

I. 脉冲编码器的可视化

为了评估可学习编码器，我们将输入 $\boldsymbol{x}_t$ 提供给经过训练的网络，并显示来自 n 层中通道 c 的输出脉冲 $\boldsymbol{S}^n_t(c)$ 和发射率 $\boldsymbol{F}^n_{T_s}(c) = \frac{1}{T_s}\sum_{t=0}^{T_s-1}\boldsymbol{S}^n_t(c)$，这与 [10] 类似。尽管来自较深脉冲神经元层的输出脉冲包含更多语义特征，但它们更难以阅读和理解。因此，我们仅显示来自第一个脉冲神经元层（即 $n = 2$）的脉冲。

图 S2 显示了来自静态 Fashion-MNIST 数据集（第 1 行）的 10 个输入图像以及第一个 PLIF 神经元层（第 2、3 和 4 行）的三个典型通道（45、75 和 76）的放电率 $\boldsymbol{F}^2_{T_s=8}$。我们可以发现来自通道 45、75 和 76 的发射率检测输入图像的上、左、右边缘。图 S3(a) 显示了当给定标记为 *horse* 的输入样本时，3-D 张量 $\boldsymbol{S}^2_{t=0}(c = 0, 1, ..., 127)$ 跨通道的 2-D 网格变平，这说明了由 $t = 0$ 处的脉冲编码器提取的特征。由于 CIFAR10-DVS 数据集是从静态 CIFAR-10 数据集转换而来的，从脉冲累积的放电率可以重建经过卷积层过滤的图像。图 S3(b)说明了所有 128 个通道 ($c = 0, 1, ..., 127$) 的发射率 $\boldsymbol{F}^2_{T_s=19}$，其比图 S3(a)中的二进制输出脉冲具有更清晰的纹理。Fig. S4(a) 显示了输入 $\boldsymbol{x}_t$（第 1 行）以及 $t = 0, 1, ..., 19$ 处通道 40 和 103（第 2 行和第 3 行）的相应输出脉冲 $\boldsymbol{S}^2_t$，以及图 S4(b)显示了 $\boldsymbol{x}(T_s) = \frac{1}{T_s}\sum_{t=0}^{T_s-1}\boldsymbol{x}_t$（第 1 行）的平均输入以及 $T_s = 0, 1, ..., 19$ 处通道 40 和 103（第 2 行和第 3 行）的相应发射率 $\boldsymbol{F}^2_{T_s}$。可以发现，随着 T_s 的增加，由发射率 $\boldsymbol{F}^2_{T_s}$ 构建的纹理变得更加清晰，这与 Poisson 编码器的使用类似。

图 S5 可视化 DVS128 手势数据集中的三个输入样本 $\boldsymbol{x}_t$ 和输出脉冲 $\boldsymbol{S}^2_t(c = 59)$。图 S5 的第 1、3、5 行显示了 DVS128 手势数据集中在 $t = 0, 1, ..., 19$ 处标记为 *random other gestures*、*right hand clockwise*、*drums* 的三个样本。为了进行比较，第 2、4、6 行显示了第一个常规层中 PLIF 神经元通道 59 的相应输出脉冲。一个关键的区别是输出几乎仅包括手势的响应脉冲，这表明脉冲神经元对空间变化和时间变化的输入数据实现了高效且准确的过滤，保留了手势但丢弃了玩家。

J. 不同编码器之间的关系

Poisson 编码器是速率编码方法之一，广泛用于 SNN [12, 37, 60, 75, 8, 21] 将图像编码为脉冲。给定图像像素 $p \in [0, 1]$，时间步长 t 的编码脉冲 S_t 以概率 p 发射。因此，整个时间步长 T 期间脉冲数量的期望为 $E_{Poisson}(\Sigma_{t=0}^{T-1}S_t) = pT$。在我们提出的 SNN 中，输入直接馈送到网络，而无需首先转换为脉冲，并且图像脉冲编码由第一个 *{Conv2d-Spiking Neurons}* 模块（*BN* 被省略）完成，它可以被视为可学习的编码器。这里我们将该编码器表示为 ENC_l。如果我们将 *Conv2d* 设置为不可学习，$channels = kernel\ size = 1$ 为不可学习，核权重为常数 $w > 0$，*Spiking Neurons* 为具有阈值电位 V_{th} 和 $V_{reset} = 0$ 的非泄漏积分和激发神经元，则该模块的脉冲数期望为 $E(\Sigma_{t=0}^{T-1}S_t) = \lfloor \frac{T}{\lceil \frac{V_{th}}{wp} \rceil} \rfloor$，其中 $\lceil\ \rceil$ 表示上限操作。我们可以发现，当 $V_{th} = w = 1$ 时，$E(\Sigma_{t=0}^{T-1}S_t) \approx E_{Poisson}(\Sigma_{t=0}^{T-1}S_t)$，这表明 ENC_l 可以逼近 Possion 编码器在速率编码中的功能。

[66, 70, 71, 72]中使用的延迟编码器是一种代表性的时间编码方法。延迟编码器将图像像素 p 编码为时间步长 t_p 的脉冲。因此，输入信息被编码在脉冲的精确触发时间中。t_p 通常与输入强度 p 成反比，例如，$t_p = \lfloor (T_{max} - 1)(1 - p) \rfloor$ 和 T_{max} 是编码周期。我们还可以发现，对于给定输入 p，ENC_l 的第一次触发时间是 $\lceil \frac{V_{th}}{wp} \rceil$，这满足输入强度 p 越大，脉冲就越快。事实上，延迟编码器是一种极其简化的可学习编码器，具有直接输入的图像。在本文中，提出的可学习编码器具有可学习权重和更多通道，能够将图像编码为具有更多语义信息的复杂脉冲模式，例如，保留手势但丢弃来自 DVS128 手势数据集的样本的玩家（见图 S5）。

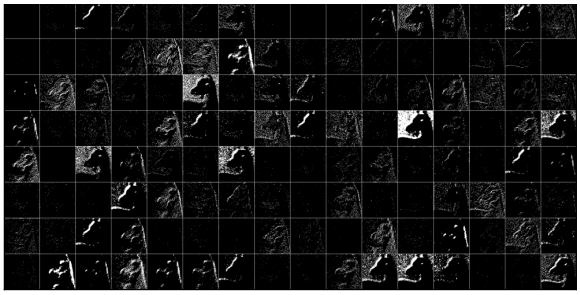
References

[1] Arnon Amir, Brian Taba, David Berg, Timothy Melano, Jeffrey McKinstry, Carmelo Di Nolfo, Tapan Nayak, Alexander Andreopoulos, Guillaume Garreau, Marcela Mendoza, et al. A low power, fully event-based gesture recognition system. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7243–7252, 2017. 2

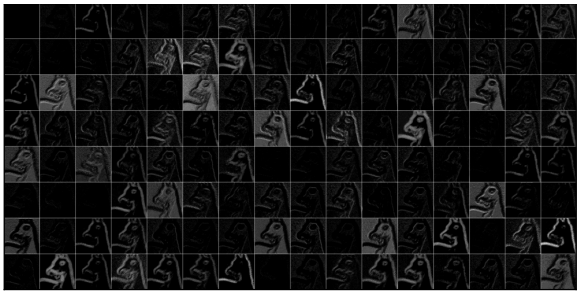
[2] Guillaume Bellec, Darjan Salaj, Anand Subramoney, Robert Legenstein, and Wolfgang Maass. Long short-term memory and learning-to-learn in networks of spiking neurons. In *Advances in Neural Information Processing Systems*, pages 787–797, 2018. 2

[3] Guo-qiang Bi and Mu-ming Poo. Synaptic modifications in cultured hippocampal neurons: dependence on spike timing, synaptic strength, and postsynaptic cell type. *Journal of Neuroscience*, 18(24):10464–10472, 1998. 2

[4] Yin Bi, Aaron Chadha, Alhabib Abbas, Eirina Bourtsoulatze, and Yiannis Andreopoulos. Graph-based object classification for neuromorphic vision sensing. In

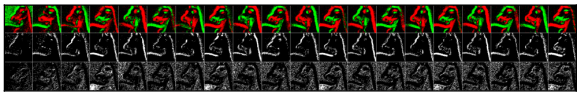


(a) $\boldsymbol{S}^2_{t=0}(c=0,1,...,127)$

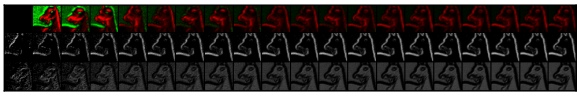


(b) $\boldsymbol{F}^2_{T_s=19}(c=0,1,...,127)$

Figure S3. 给定来自 CIFAR10-DVS 的标记为 *horse* 的样本，(a) 显示 $t=0$ 处第一个脉冲神经元层的所有 128 个通道的脉冲，(b) 显示 $T_s=19$ 处这些神经元的放电率。



(a) $\boldsymbol{x}_t$ and $\boldsymbol{S}^2_t(c=40,103)$ at $t=0,1,...,19$



(b) $\boldsymbol{x}(T_s)$ and $\boldsymbol{F}^2_{T_s}(c=40,103)$ at $T_s=0,1,...,19$

Figure S4. 给定如图 S3 的样本，每个时间步的通道 40 和 103 的输入数据和输出脉冲分别如第 1、2、3 行的 (a) 所示。(b) 显示了每个时间步的通道 40 和 103 的平均输入数据和发射率。

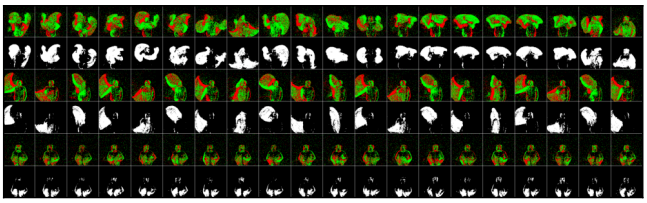


Figure S5. 第 1、3、5 行显示了 DVS128 手势数据集中标记为 *random other gestures*、*right hand clockwise*、*drums* 的三个样本。第 2、4、6 行显示了来自第一个 PLIF 神经元层的通道 59 的相应输出脉冲。

Proceedings of the IEEE International Conference on Computer Vision, pages 491–501, 2019. 6

[5] Sander M Bohte, Joost N Kok, and Han La Poutre. Error-backpropagation in temporally encoded networks of spiking neurons. *Neurocomputing*, 48(1-4):17–37, 2002. 2

[6] Matthew Botvinick, Jane X. Wang, Will Dabney, Kevin J. Miller, and Zeb Kurth-Nelson. Deep reinforcement learning and its neuroscientific implications. *Neuron*, 107(4):603–616, 2020. 2

[7] Yongqiang Cao, Yang Chen, and Deepak Khosla. Spiking deep convolutional neural networks for energy-efficient object recognition. *International Journal of Computer Vision*, 113(1):54–66, 2015. 2

[8] Xiang Cheng, Yunzhe Hao, Jiaming Xu, and Bo Xu. LISNN: Improving Spiking Neural Networks with Lateral Interactions for Robust Object Recognition. In *IJCAI*, pages 1519–1525. International Joint Conferences on Artificial Intelligence Organization, 7 2020. 2, 4, 6, 9, 10

[9] Gustavo Deco, Josephine Cruzat, and Morten L Kringelbach. Brain songs framework used for discovering the relevant timescale of the human brain. *Nature Communications*, 10(1):1–13, 2019. 1

[10] Lei Deng, Yujie Wu, Xing Hu, Ling Liang, Yufei Ding, Guoqi Li, Guangshe Zhao, Peng Li, and Yuan Xie. Rethinking the performance comparison between SNNS and ANNS. *Neural Networks*, 121:294–307, 2020. 10

[11] Shikuang Deng and Shi Gu. Optimal conversion of conventional artificial neural networks to spiking neural networks. In *International Conference on Learning Representations*, 2021. 2

[12] Peter U Diehl and Matthew Cook. Unsupervised learning of digit recognition using spike-timing-dependent plasticity. *Frontiers in Computational Neuroscience*, 9:99, 2015. 2, 4, 10

[13] Haowen Fang, Amar Shrestha, Ziyi Zhao, and Qinru Qiu. Exploiting Neuron and Synapse Filter Dynamics in Spatial Temporal Learning of Deep Spiking Neural Network. *arXiv preprint arXiv:2003.02944*, 2020. 6

[14] Wei Fang, Yanqi Chen, Jianhao Ding, Ding Chen, Zhaofei Yu, Huihui Zhou, Yonghong Tian, and other contributors. Spikingjelly. https://github.com/fangwei123456/spikingjelly, 2020. 8

[15] Nicolas Frémaux and Wulfram Gerstner. Neuromodulated spike-timing-dependent plasticity, and theory of three-factor learning rules. *Frontiers in Neural Circuits*, 9:85, 2016. 2

[16] Nicolas Frémaux, Henning Sprekeler, and Wulfram Gerstner. Reinforcement learning using a continuous time actor-critic framework with spiking neurons. *PLoS Computational Biology*, 9(4):e1003024, 2013. 2

[17] Johannes Friedrich, Robert Urbanczik, and Walter Senn. Spatio-temporal credit assignment in neuronal population learning. *PLoS Computational Biology*, 7(6):e1002092, 2011. 2

[18] Wulfram Gerstner, Werner M Kistler, Richard Naud, and Liam Paninski. *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press, 2014. 1, 2

[19] Shangqi Guo, Zhaofei Yu, Fei Deng, Xiaolin Hu, and Feng Chen. Hierarchical bayesian inference and learning in spiking neural networks. *IEEE Transactions on Cybernetics*, 49(1):133–145, 2017. 2

[20] Robert Gütig and Haim Sompolinsky. The tempotron: a neuron that learns spike timing–based decisions. *Nature Neuroscience*, 9(3):420–428, 2006. 2

[21] Bing Han, Gopalakrishnan Srinivasan, and Kaushik Roy. RMP-SNN: Residual Membrane Potential Neuron for Enabling Deeper High-Accuracy and Low-Latency Spiking Neural Network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 13558–13567, 2020. 2, 4, 6, 10

[22] Michael E Hasselmo and Chantal E Stern. Mechanisms underlying working memory for novel information. *Trends in Cognitive Sciences*, 10(11):487–493, 2006. 1

[23] Weihua He, YuJie Wu, Lei Deng, Guoqi Li, Haoyu Wang, Yang Tian, Wei Ding, Wenhui Wang, and Yuan Xie. Comparing SNNs and RNNs on Neuromorphic Vision Datasets: Similarities and Differences. *arXiv preprint arXiv:2005.02183*, 2020. 6, 9

[24] Donald Olding Hebb. *The organization of behavior: A neuropsychological theory*. Psychology Press, 2005. 2

[25] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. 4

[26] Eric Hunsberger and Chris Eliasmith. Spiking deep networks with LIF neurons. *arXiv preprint arXiv:1510.08829*, 2015. 2, 6

[27] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 2

[28] Yingyezhe Jin, Wenrui Zhang, and Peng Li. Hybrid macro/micro level backpropagation for training deep spiking neural networks. In *Advances in Neural Information Processing Systems*, pages 7005–7015, 2018. 2, 6

[29] Jacques Kaiser, Hesham Mostafa, and Emre Neftci. Synaptic Plasticity Dynamics for Deep Continuous Local Learning (DECOLLE). *Frontiers in Neuroscience*, 14:424, 2020. 2, 6, 9

[30] David Kappel, Robert Legenstein, Stefan Habenschuss, Michael Hsieh, and Wolfgang Maass. A dynamic connectome supports the emergence of stable computational function of neural circuits through reward-based learning. *eNeuro*, 5(2), 2018. 2

[31] Saeed Reza Kheradpisheh, Mohammad Ganjtabesh, Simon J Thorpe, and Timothée Masquelier. Stdp-based spiking deep convolutional neural networks for object recognition. *Neural Networks*, 99:56–67, 2018. 2

[32] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 9

[33] Christof Koch, Moshe Rapp, and Idan Segev. A brief history of time (constants). *Cerebral Cortex*, 6(2):93–101, 1996. 1

[34] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, 2009. 2

[35] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 2

[36] Eimantas Ledinauskas, Julius Ruseckas, Alfonsas Juršėnas, and Giedrius Buračas. Training Deep Spiking Neural Networks. *arXiv preprint arXiv:2006.04436*, 2020. 2, 3

[37] Chankyu Lee, Syed Shakib Sarwar, Priyadarshini Panda, Gopalakrishnan Srinivasan, and Kaushik Roy. Enabling spike-based backpropagation for training deep neural network architectures. *Frontiers in Neuroscience*, 14, 2020. 2, 4, 5, 6, 8, 10

[38] Jun Haeng Lee, Tobi Delbruck, and Michael Pfeiffer. Training deep spiking neural networks using backpropagation. *Frontiers in Neuroscience*, 10:508, 2016. 2, 6, 9

[39] Hongmin Li, Hanchao Liu, Xiangyang Ji, Guoqi Li, and Luping Shi. Cifar10-dvs: an event-stream dataset for object classification. *Frontiers in Neuroscience*, 11:309, 2017. 2

[40] Qianhui Liu, Haibo Ruan, Dong Xing, Huajin Tang, and Gang Pan. Effective AER Object Classification Using Segmented Probability-Maximization Learning in Spiking Neural Networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 1308–1315, 2020. 6

[41] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 9

[42] Wolfgang Maass. Networks of spiking neurons: the third generation of neural network models. *Neural Networks*, 10(9):1659–1671, 1997. 1

[43] Maurizio Mattia and Paolo Del Giudice. Population dynamics of interacting spiking neurons. *Physical Review E*, 66(5):051917, 2002. 1

[44] Ammar Mohemmed, Stefan Schliebs, Satoshi Matsuda, and Nikola Kasabov. Span: Spike pattern association neuron for learning spatio-temporal spike patterns. *International Journal of Neural Systems*, 22(04):1250012, 2012. 2

[45] Milad Mozafari, Mohammad Ganjtabesh, Abbas Nowzari-Dalini, Simon J Thorpe, and Timothée Masquelier. Bio-inspired digit recognition using reward-modulated spike-timing-dependent plasticity in deep convolutional networks. *Pattern Recognition*, 94:87–95, 2019. 2

[46] Emre O Neftci, Hesham Mostafa, and Friedemann Zenke. Surrogate gradient learning in spiking neural networks. *IEEE Signal Processing Magazine*, 36:61–63, 2019. 2

[47] Daniel Neil and Shih-Chii Liu. Effective sensor fusion with event-based sensors and deep network architectures. In *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 2282–2285. IEEE, 2016. 9

[48] Daniel Neil, Michael Pfeiffer, and Shih-Chii Liu. Phased lstm: Accelerating recurrent network training for long or event-based sequences, 2016. 9

[49] Bernhard Nessler, Michael Pfeiffer, Lars Buesing, and Wolfgang Maass. Bayesian computation emerges in generic cortical microcircuits through spike-timing-dependent plasticity. *PLoS Computational Biology*, 9(4):e1003037, 2013. 2

[50] Garrick Orchard, Ajinkya Jayawant, Gregory K Cohen, and Nitish Thakor. Converting static image datasets to spiking neuromorphic datasets using saccades. *Frontiers in Neuroscience*, 9:437, 2015. 2

[51] Filip Ponulak and Andrzej Kasiński. Supervised learning in spiking neural networks with ReSuMe: sequence learning, classification, and spike shifting. *Neural Computation*, 22(2):467–510, 2010. 2

[52] Wiebke Potjans, Markus Diesmann, and Abigail Morrison. An imperfect dopaminergic error signal can drive temporal-difference learning. *PLoS Computational Biology*, 7(5):e1001133, 2011. 2

[53] Nitin Rathi and Kaushik Roy. DIET-SNN: Direct Input Encoding With Leakage and Threshold Optimization in Deep Spiking Neural Networks. *arXiv preprint arXiv:2008.03658*, 2020. 2, 3, 4, 6

[54] Nitin Rathi, Gopalakrishnan Srinivasan, Priyadarshini Panda, and Kaushik Roy. Enabling deep spiking neural networks with hybrid conversion and spike timing dependent backpropagation. *arXiv preprint arXiv:2005.01807*, 2020. 4

[55] Kaushik Roy, Akhilesh Jaiswal, and Priyadarshini Panda. Towards spike-based machine intelligence with neuromorphic computing. *Nature*, 575(7784):607–617, 2019. 1

[56] Bodo Rueckauer, Iulia-Alexandra Lungu, Yuhuang Hu, Michael Pfeiffer, and Shih-Chii Liu. Conversion of continuous-valued deep networks to efficient event-driven networks for image classification. *Frontiers in Neuroscience*, 11:682, 2017. 2, 4, 5, 6

[57] Abhronil Sengupta, Yuting Ye, Robert Wang, Chiao Liu, and Kaushik Roy. Going deeper in spiking neural networks: Vgg and residual architectures. *Frontiers in Neuroscience*, 13:95, 2019. 2, 4, 6

[58] H Sebastian Seung. Learning in spiking neural networks by reinforcement of stochastic synaptic transmission. *Neuron*, 40(6):1063–1073, 2003. 2

[59] Karthik H Shankar and Marc W Howard. A scale-invariant internal representation of time. *Neural Computation*, 24(1):134–193, 2012. 1

[60] Sumit B Shrestha and Garrick Orchard. Slayer: Spike layer error reassignment in time. *Advances in Neural Information Processing Systems*, 31:1412–1421, 2018. 2, 4, 6, 10

[61] Amos Sironi, Manuele Brambilla, Nicolas Bourdis, Xavier Lagorce, and Ryad Benosman. HATS: Histograms of averaged time surfaces for robust event-based object classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1731–1740, 2018. 6

[62] Narayan Srinivasa and Youngkwan Cho. Self-organizing spiking neural model for learning fault-tolerant spatio-motor transformations. *IEEE Transactions on Neural Networks and Learning Systems*, 23(10):1526–1538, 2012. 2

[63] Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, and Luping Shi. Spatio-temporal backpropagation for training high-performance spiking neural networks. *Frontiers in Neuroscience*, 12:331, 2018. 2, 6, 9

[64] Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, Yuan Xie, and Luping Shi. Direct training for spiking neural networks: Faster, larger, better. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 1311–1318, 2019. 2, 4, 6, 8, 9

[65] Yujie Wu, Rong Zhao, Jun Zhu, Feng Chen, Mingkun Xu, Guoqi Li, Sen Song, Lei Deng, Guanrui Wang, Hao Zheng, et al. Brain-inspired global-local hybrid learning towards human-like intelligence. *arXiv preprint arXiv:2006.03226*, 2020. 9

[66] Simei Gomes Wysoski, Lubica Benuskova, and Nikola Kasabov. Fast and adaptive network of spiking neurons for multi-view visual pattern recognition. *Neurocomputing*, 71(13):2563–2575, 2008. Artificial Neural Networks (ICANN 2006) / Engineering of Intelligent Systems (ICEIS 2006). 10

[67] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017. 2

[68] Yannan Xing, Gaetano Di Caterina, and John Soraghan. A new spiking convolutional recurrent neural network (scrnn) with applications to event-based

hand gesture recognition. *Frontiers in Neuroscience*, 14:1143, 2020. 6, 9

[69] Bojian Yin, Federico Corradi, and Sander M Bohté. Effective and Efficient Computation with Multiple-timescale Spiking Recurrent Neural Networks. *arXiv preprint arXiv:2005.11633*, 2020. 2, 3

[70] Qiang Yu, Kay Chen Tan, and Huajin Tang. Pattern recognition computation in a spiking neural network with temporal encoding and learning. *The 2012 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7, 2012. 10

[71] Qiang Yu, Huajin Tang, Kay Chen Tan, and Haizhou Li. Rapid feedforward computation by temporal encoding and learning with spiking neurons. *IEEE Transactions on Neural Networks and Learning Systems*, 24(10):1539–1552, 2013. 10

[72] Qiang Yu, Huajin Tang, Kay Chen Tan, and Haoyong Yu. A brain-inspired spiking neural network model with temporal encoding and learning. *Neurocomputing*, 138:3–13, 2014. 10

[73] Mengwen Yuan, Xi Wu, Rui Yan, and Huajin Tang. Reinforcement Learning in Spiking Neural Networks with Stochastic and Deterministic Synapses. *Neural Computation*, 31(12):2368–2389, 2019. 2

[74] Friedemann Zenke and Tim P Vogels. The remarkable robustness of surrogate gradient learning for instilling complex function in spiking neural networks. *BioRxiv*, 2020. 2, 9

[75] Wenrui Zhang and Peng Li. Spike-train level backpropagation for training deep recurrent spiking neural networks. In *Advances in Neural Information Processing Systems*, pages 7802–7813, 2019. 2, 4, 6, 10

[76] Romain Zimmer, Thomas Pellegrini, Srisht Fateh Singh, and Timothée Masquelier. Technical report: supervised training of convolutional spiking neural networks with PyTorch. *arXiv preprint arXiv:1911.10124*, 2019. 2