

基于低时延和高精度脉冲神经网络的目标检测^①



明晓钰, 李翔宇

(南京理工大学 计算机科学与工程学院, 南京 210094)

通信作者: 李翔宇, E-mail: wl@njust.edu.cn

摘要: 神经网络 (artificial neural network, ANN) 在众多领域取得了显著进展, 但其对计算资源和能耗的高需求限制了其在硬件端的部署和应用. 脉冲神经网络 (spiking neural network, SNN) 因其低功耗和快速推理的特性, 在神经形态硬件上表现出色. 然而, SNN 的神经元动态和脉冲发放机制导致其训练过程复杂, 目前主要研究集中在图像分类任务上, 本文尝试将 SNN 应用于更为复杂的计算机视觉任务. 本文以 YOLOv3-tiny 网络为基础, 提出了 Spiking YOLOv3 模型, 其符合 SNN 特性的网络模型, 在检测任务上实现了更高的准确度, 并将平均推理时间减少至约原来工作的 1/4. 此外, 我们还分析了 ANN-SNN 转换过程中产生的转换误差, 并采用量化激活函数对 Spiking YOLOv3 模型进行了优化以减小转换误差. 优化后的模型平均推理时间减少至约原来的 1/2, 并在 VOC 与 UAV 数据集上实现在 ANN-SNN 无损转换, 显著提升了基于该模型的目标检测效率.

关键词: 脉冲神经网络; 神经网络-脉冲神经网络 (ANN-SNN) 转换; 目标检测; 低时延和高精度

引用格式: 明晓钰, 李翔宇. 基于低时延和高精度脉冲神经网络的目标检测. 计算机系统应用, 2024, 33(7): 170-179. <http://www.c-s-a.org.cn/1003-3254/9566.html>

Target Detection Based on Low Latency and High Accuracy Spiking Neural Network

MING Xiao-Yu, LI Xiang-Yu

(School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China)

Abstract: Artificial neural network (ANN) has made significant progress in many fields, but its high demand for computing resources and energy consumption limits its deployment and application on the hardware side. Spiking neural network (SNN) performs well on neural morphology hardware due to its low power consumption and fast inference characteristics. However, the neural dynamics and pulse propagation mechanism of SNN make its training process complex. Current research primarily focuses on image classification tasks. This study attempts to apply SNN to more complex computer vision tasks. This study is based on the YOLOv3 tiny network and proposes the spiking YOLOv3 model, which conforms to the SNN characteristics of the network model. It achieves higher accuracy in detection tasks and reduces the average inference time to about 1/4 of the original work. In addition, this study also analyzes the conversion errors generated during the ANN-SNN conversion process and optimizes the Spiking YOLOv3 model using a quantization activation function to reduce conversion errors. The optimized model reduces the average inference time to about half of the original and achieves lossless conversion on the VOC and UAV datasets in ANN-SNN, significantly improving the detection efficiency based on this model.

Key words: spiking neural network (SNN); artificial neural network-spiking neural network (ANN-SNN) conversion; target detection; low latency and high accuracy

① 收稿时间: 2024-01-11; 修改时间: 2024-02-07; 采用时间: 2024-03-11; csa 在线出版时间: 2024-05-31

CNKI 网络首发时间: 2024-06-04

ANN 在众多领域表现优异^[1,2], 然而其较高的计算开销和能耗需求对硬件部署不利. 随着类脑研究的深入, 具有仿生特性的 SNN 逐渐受到关注^[3]. SNN 由离散脉冲时间序列构成的神经元模型, 模拟了实际生物神经网络中的独特机制, 能够通过脉冲序列表达和传递信息, 其更符合生物学上的合理性. 与 ANN 相比, SNN 模拟了生物神经元的工作方式, 只有在需要时才发放脉冲, 因此能够实现更高的能量效率降低网络计算量, 更适用于部署在硬件至上以降低能耗. 且 SNN 对于处理时间相关的信息具有更好的性能, 其适用于处理时间序列数据、事件驱动的场景、需要低能耗的嵌入式系统等场合. 因此, 采用 SNN 实现复杂的目标检测任务是一个理想的选择.

然而, 由于 SNN 的神经元复杂动力学和不可微操作^[4], 传统反向传播算法无法直接应用于 SNN. 近年来, 有学者提出用近似连续函数替代脉冲函数或其导数, 从而产生基于脉冲的反向传播算法^[5,6]. 但此方法局限于浅层 SNN 和简单任务, 且需要大量计算和存储能力. ANN-SNN 转换方法通过建立激活输出和平均发射速率之间的映射关系, 避免了直接训练 SNN. 转换方法通常需利用较大时间步长克服转换误差, 但也有许多工作从不同角度优化该方法, 如可训练的阈值^[7], 软重置机制^[8], 阈值偏移^[9]等. 现有研究在较小时间步长下难以消除 ANN 到 SNN 的转换误差. Spiking YOLO^[10]是深度 SNN 的首个目标检测模型, 但是该模型需要 8000 个时间步长才能到达与原始 ANN 相当的结果. 本文的主要贡献可以总结如下.

1) 提出了 Spiking YOLOv3 模型, 该模型基于 YOLOv3-tiny 网络模型, 经过优化以适应后续的脉冲神经网络转换. 此外, 本文采用了细粒度通道归一化方法, 使得神经元能够获得适当的放电率, 从而提高了模型的精度.

2) 在理论上, 对人工神经网络到脉冲神经网络的转换过程产生的误差进行了分析, 并进一步通过量化激活函数对 Spiking YOLOv3 模型进行了优化.

3) 在 PASCAL VOC 和 MS COCO 数据集上对 Spiking YOLOv3 模型进行测试. 实验发现, 我们所提出的 Spiking YOLOv3 模型的平均推理时间约为先前工作的 1/4, 同时准确率分别提升了约 4.65% 和 9.32%. 经过优化后, Spiking YOLOv3-FC 模型在 UAV, PASCAL VOC 和 MS COCO 数据集上的平均推理时间约需优

化前模型的 1/2, 且在 PASCAL VOC 和 UAV 数据集上实现了无损转换, 在 COCO 数据集上的转换误差降低到了 0.93%.

1 相关工作

对于 ANN 到 SNN 的转换研究, 最早由 Cao 等人^[11]发起的, Diehl^[12]提出了基于数据和基于模型的归一化方法, 将 3 层的 ANN 转换为 SNN, Sengupta 等人^[7]又对阈值进行缩放以减小精度的损失. 为了解决信息的丢失问题, Han 等人^[8]提出使用“减法重置”神经元. 为了消除 ANN-SNN 的转换误差, Rueckauer^[13]提出使用 p -Norm 的方法, 用 99.9% 的激活百分位数作为尺度因子, 避免在激活分布中挑选异常值. Ho 等人^[14]在 ANN 的 ReLU 层后面添加约束, 降低激活值, 从而减小截断误差与量化误差. 此外, Han 等人^[8]重新调整了 SNN 阈值, 以避免脉冲神经元的当激活. Deng 等人^[9]引入额外的偏差以最小化分层误差, Li 等人^[15]进一步提出逐层修正误差的校正算法, 通过校准 SNN 中的参数来匹配转换后的激活, 从而缩小源神经网络与校准后 SNN 之间的激活分布差距.

Spiking YOLO 首次将 SNN 应用到目标检测上面, 采用通道归一化与 IBT 平衡神经元的放电率, 在深度 SNN 中实现快速、准确的信息传递. 该工作需要 8000 个时间步长才能达到与原始 ANN 相当的准确率. 在其后续工作中^[16], 使用贝叶斯优化的阈值平衡方法, 找到最优阈值电压, 但仍需 5000 个时间步长才能达到与 ANN 相当的准确率.

2 本文方法

2.1 SNN 神经元

本文在 SNN 的神经元模型选择上, 采用了 IF 神经元模型. 为降低信息损失, 采用了文献^[8]提出的“减法重置”机制, 神经元放电的瞬间, 超过阈值的部分将被保留. 因此, IF 神经元的动力学方程可以表示为:

$$V_{\text{temp}}^l(t) = V^l(t-1) + W^l x^{l-1}(t) \quad (1)$$

$$V^l(t) = V_{\text{temp}}^l(t) - s^l(t)\theta^l \quad (2)$$

其中, $V_{\text{temp}}^l(t)$ 为脉冲神经网络中第 t 个时间步长突触前膜电位, $V^l(t)$ 表示脉冲神经网络中第 t 个时间步长突触后膜电位, W^l 表示第 $l-1$ 层和第 l 层之间的突触权重,

x^{l-1} 表示第 l 层神经元接收来自上一层的输入, s^l 表示第 l 层的神经元所发放的脉冲, θ^l 是第 l 层神经元的激发阈值, 脉冲的激发规则可以用如下公式来表示:

$$s^l(t) = \begin{cases} 1, & V_{\text{temp}}^l \geq \theta^l \\ 0, & \text{else} \end{cases} \quad (3)$$

对于一个脉冲神经元, 其膜电位超过激发阈值, 则该神经元将发放脉冲, 如果第 $l-1$ 层的突触前神经元发出脉冲, 则第 l 层的突触后神经元接收未加权的突触后电位 θ^l , 其为输入值 $x^l(t)$, 即:

$$x^l(t) = s^l(t)\theta^l \quad (4)$$

2.2 ANN-SNN 转换

本文中 ANN-SNN 转换的主要思想是用 SNN 的脉冲发射率 (平均突触后电位) 来近似 ANN 中的激活值. 将式 (1) 与式 (2) 结合得:

$$V^l(t) = V^l(t-1) + W^l x^{l-1}(t) - s^l(t)\theta^l \quad (5)$$

式 (5) 对时间步长 $1-T$ 求和得:

$$\frac{V^l(T) - V^l(0)}{T} = W^l \frac{\sum_{t=1}^T x^{l-1}(t)}{T} - \frac{\sum_{t=1}^T s^l(t)\theta^l}{T} \quad (6)$$

其中, T 表示模拟的总时间步长 (总周期), 用 $\phi^l(T)$ 来表示平均突触后电位, 并结合式 (4) 与式 (6) 得:

$$\phi^l(T) = \frac{\sum_{t=1}^T s^l(t)\theta^l}{T} \quad (7)$$

$$\phi^l(T) = W^l \phi^{l-1}(T) + \left(-\frac{V^l(T) - V^l(0)}{T} \right) \quad (8)$$

式 (8) 描述了脉冲神经网络中两个相邻层之间神经元的平均突触后电位的关系.

2.3 Spiking YOLOv3 模型

目标检测是在目标识别的基础上, 确定图像中物体的精确坐标. YOLOv3 算法在目标检测领域表现优异, 其借鉴了特征金字塔的思想, 不仅在每个特征图上进行独立的预测, 同时通过上采样小特征图至大特征图尺寸, 再与大特征图拼接以进一步预测. YOLOv3-tiny 是 YOLOv3 的轻量级版本, 其在 YOLOv3 的基础上减少了一些特征层, 仅保留了两个独立预测分支, 速度更快. 因此, 我们选择 YOLOv3-tiny 作为基础网络模型.

然而, 将原始的 ANN 转换成 SNN 时, 通常会会出现不可接受的精度损失. 其主要损失来源包括:

1) ANN 中的负输出值在 SNN 中很难精确的表示, 使用双曲线正切 (tanh) 或归一化指数函数 (Softmax) 后, 非线性神经元的输出值可以得正也可以得负, 而脉冲神经元的速率只能是正值. 因此, 负值总被丢弃, 导致转换后的 SNN 的精度下降.

2) SNN 无法直接应用 ANN 中的批量归一化 (batch normalization, BN) 层, 用在 SNN 中, 通常会被舍弃.

3) SNN 的信息传递是脉冲形式, 导致最大池化层在转换过程中精度损失, 增加 SNN 的复杂度.

针对上述问题, 我们对基于 YOLOv3-tiny 的基础网络模型进行如下改进:

- 1) 使用 ReLU 激活函数替代 Leaky ReLU 激活函数.
- 2) 将步长为 2 的最大池化层替换为步长为 2 的卷积 (convolution, Conv) 层, 并移除步长为 1 的最大池化层.
- 3) 将上采样层替换为反卷积层.
- 4) 将 BN 层权值折叠进 Conv 层权值.

我们采用 ReLU 激活函数替代 YOLOv3-tiny 中的 Leaky ReLU 激活函数的原因在于, ANN 中 ReLU 神经元的非线性激活与 SNN 中 IF 神经元的脉冲发放率之间具有较强相关性. 如图 1, IF 神经元脉冲发放频率与输入的曲线与 ReLU 曲线几乎一致. 借助这一特性, 我们可以解决负输出问题, 提高转换后的 SNN 精度.

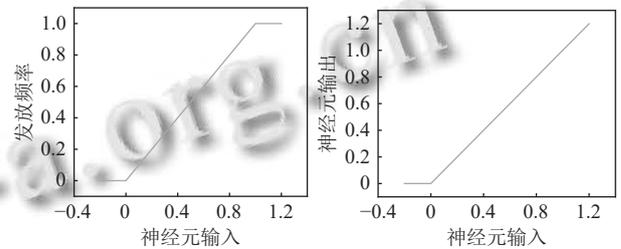


图 1 IF 神经元脉冲发放频率和输入 x_i 和 $\text{ReLU}(x_i)$ 的比较

我们采取步长为 2 的卷积层取代原有步长为 2 的最大池化层, 同时移除了步长为 1 的最大池化层. 这一调针对上述提到的精度损失来源第 3 点. 由于 ANN 中的最大池化层转化为 SNN 中的对应层时会导致精度下降, 之前有研究选择用平均池化层来替代, 或直接移除池化层. 然而, Na 等人^[17]的研究指出用平均层替换最大池化层会对模型的精度和能量效率产生负面影响. 同时, 完全移除池化层会丧失降维和去除冗余信息的功能. 卷积层通过调整步长大小同样可以缩减特征图的尺寸并增大感受野. 因此, 我们选采步长为 2 的最大池化层来替代步长为 2 的卷积层, 并移除步长

为1的最大池化层。

对于 YOLOv3-tiny 模型中的上采样层, 我们采用反卷积层进行替换. 因上采样层不具备权重, 在转化为 SNN 结构时存在困难, 而反卷积层与上采样层功能相似, 能够更顺畅地完成这一转换. 反卷积操作如图 2 所示, 其在四周添加 0 元素后再进行卷积操作.

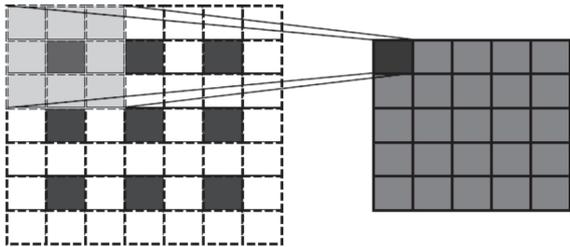


图 2 反卷积操作图 (在输入 3×3 特征图四周填充 0 元素, 采用 3×3 卷积核进行卷积操作, 输出 5×5 特征图)

在 SNN 中, 由于神经元的活动具有脉冲性, BN 层的传统应用方式不再适用. BN 层在提高模型训练速度的同时, 对模型输出精度有着积极影响, 直接去除 BN 层可能会导致网络精度的损失. 为解决这一问题, 我们采取将 BN 层折叠进前一卷积层权值中. 这种融合确保了在 SNN 中也能保留 BN 层的优势, 而不会牺牲网络性能. BN 层的公式如下:

$$\mu_B = \frac{1}{m} \sum_{i=1}^m x_i \quad (9)$$

$$\sigma_B^2 = \frac{1}{m} \sum_{i=1}^m (x_i - \mu_B)^2 \quad (10)$$

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 - \varepsilon}} \quad (11)$$

$$y_i = \gamma \hat{x}_i + \beta \quad (12)$$

式 (9)、式 (10) 为计算 1 个批次输入数据的平均值与方差, 式 (9) 对数据进行标准化, 式 (10) 为归一化的数据. 其中 x_i 是某一个批次即将进入激活函数的输入的第 i 个数据, μ_B 是这一批次所有数据的平均值, m 是这批次数据的个数, σ_B 是这一批次所有数据的方差, ε 是为了防止分母为 0 的 1 个较小的数, y_i 是经过归一化后的数据, γ 是缩放因子, β 是偏移. 由式 (9) 和式 (10) 将 BN 层折叠进 Conv 层公式如下:

$$\tilde{W}_{i,j}^l = \frac{\gamma_i^l}{\sigma_i^l} W_{i,j}^l \quad (13)$$

$$\tilde{b}_i^l = \frac{\gamma_i^l}{\sigma_i^l} (b_i^l - \mu_i^l) + \beta_i^l \quad (14)$$

其中, $W_{i,j}^l$ 是第 l 层神经元 i 第 j 个突触的权重, γ_i^l 是第 l 层神经元 i 的缩放因子, σ_i^l 是一批次数据的方差, b_i^l 是 l 层神经元 i 的偏置, μ_i^l 是一批次数据的平均值, β_i^l 是 l 层神经元 i 的偏移.

经过上述网络模型调整后, 得 Spiking YOLOv3 网络模型, 如图 3 所示.

Spiking YOLOv3 网络模型在尽可能地减少精度损失的同时, 还能在较低的时延下实现较高的检测精度. 最终的 Spiking YOLOv3 网络结构详细参数如表 1 所示.

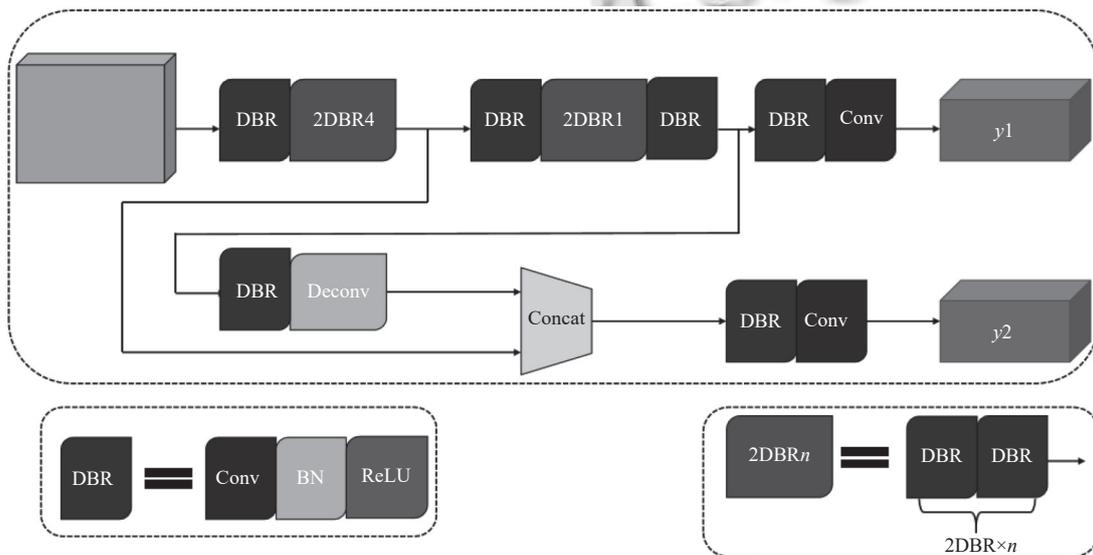


图 3 Spiking YOLOv3 网络模型图 (DBR 模块为卷积+批处理归一化+ReLU 激活函数组成, 2DBR 模块为两个 DBR 模块)

表1 Spiking YOLOv3 的网络结构表

层	滤波器数量	卷积核大小	步长	输出
Conv1	16	3	1	416×416×16
Conv2	16	3	2	208×208×16
Conv3	32	3	1	208×208×32
Conv4	32	3	2	104×104×32
Conv5	64	3	1	104×104×64
Conv6	64	3	2	52×52×64
Conv7	128	3	1	52×52×128
Conv8	128	3	2	26×26×128
Conv9	256	3	1	26×26×256
Conv10	256	3	2	13×13×256
Conv11	512	3	1	13×13×512
Conv12	1024	3	1	13×13×1024
Conv13	256	1	1	13×13×256
Conv14	512	3	1	13×13×512
Conv15	255	1	1	13×13×255
Conv16	128	1	1	13×13×128
ConvTranspose1	128	3	2	26×26×128
Conv17	256	3	1	26×26×256
Conv18	255	1	1	26×26×255

在给定的时间步长下,当利用 SNN 进行目标检测时,神经元的激活不足或过渡激活都可能导致信息的丢失.为确保神经元的有效性,我们采用细粒度的归一化方法.该方法在每层的通道维度上,使用第 99.9 个百分位激活值对权值进行归一化,图 4 为通道归一化具体操作示意图.

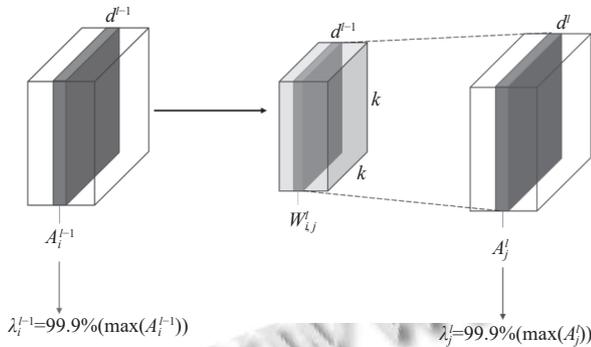


图4 通道归一化

通道归一化的具体公式如下:

$$\tilde{W}_{i,j}^l = W_{i,j}^l \frac{\lambda_i^{l-1}}{\lambda_j^l} \text{ and } \tilde{b}_i^l = \frac{b_i^l}{\lambda_j^l} \quad (15)$$

其中, i 和 j 是通道的索引, l 层权值 W 通过在每个通道使用第 99.9 个百分位激活值 λ_j^l 进行归一化. 激活值是由训练数据集计算. 对于非首层, 归一化的激活值须乘 λ_i^{l-1} 来将输入还原为上一层归一化前的值, 再进行本层的归一化. 通道归一化后神经元可以获得更高但是合适的放电率, 从而在短时间内实现准确信息传输.

2.4 ANN-SNN 转换误差

对于 ANN, 前向传播的过程可以描述为一个线性变换过程和一个 ReLU 激活函数的非线性映射:

$$a^l = f(W^l a^{l-1}), l = 1, 2, 3, \dots, L \quad (16)$$

其中, $f(\cdot)$ 为 ReLU 激活函数, 当时间步长趋于无穷大, 式 (8) 中 $(V^l(T) - V^l(0))/T$ 可忽略, 从而有 $\phi^l = a^l$, 实现无损的 ANN-SNN 转换. 然而, 在短时间步长的情况下, 转换后的 SNN 性能下降严重.

假设 $\phi^{l-1} = a^{l-1}$, 即 ANN 和 SNN 从 $l-1$ 层接收相同的输入, 则误差可用如下公式来表示:

$$E^l = \phi^l - a^l = W^l \phi^{l-1} - \frac{V^l(T) - V^l(0)}{T} - f(W^l a^l) \quad (17)$$

因接收相同的输入 $\phi^{l-1} = a^{l-1}$, 进一步简化式 (17), 用 $z^l = W^l a^{l-1} = W^l \phi^{l-1}$ 来表示加权输入, 则可得:

$$E^l = z^l - \frac{V^l(T) - V^l(0)}{T} - \text{ReLU}(z^l) \quad (18)$$

式 (18) 可看出转换误差非零, ANN-SNN 转换中有 3 个主要误差, 截断误差, 量化误差, 不均匀误差.

截断误差由 ANN 与 SNN 的不同的取值范围造成. 对于 ANN, 其 l 层输出的值的区间为 $[0, a_{\max}^l]$, 其中 a_{\max}^l 为 l 层的最大激活值. 对于 SNN, 其 l 层的输出, 为有限的离散集合 $S_T = \{\theta_i^l/T \mid t \in [0, T]\}$, 将 λ^l 设置为 ANN 中的实际最大值, 用其映射到 SNN 中 S_T 中的最大值 θ^l , 则 ANN 中的激活值 a^l 可通过式 (19) 来映射到 SNN 中的平均突触后膜电位 $\phi^l(T)$ 中:

$$\phi^l(T) = \text{clip}\left(\frac{\theta^l}{T} \left\lfloor \frac{a^l T}{\lambda^l} \right\rfloor, 0, \theta^l\right) \quad (19)$$

其中, $\lfloor \cdot \rfloor$ 表示 floor 函数, 当 ANN 中的激活值 $a^l \in [\lambda^l, a_{\max}^l]$ 时, 都会映射到 SNN 相同的值 θ^l , 如图 5 所示.

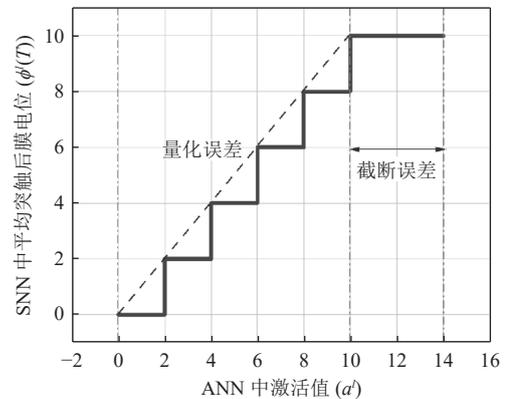


图5 截断误差与量化误差 (阈值: 10, 时间步长: 5)

量化误差是因为在 ANN 中其激活值 a^l 是连续的, 但 SNN 中平均突触后膜电位 $\phi^l(T)$ 是离散的. 当 ANN 的激活值 a^l 在 $[3\lambda^l/T, 4\lambda^l/T]$ 区间, 都会被映射为 SNN 中相同值 $3\theta^l/T$. 如图 5 中, ANN 中激活值为 6~8 区间的值, 都会被映射为 SNN 中相同的值 6.

不均匀误差是由于脉冲到达激活层的时间序列不同. 在理想情况下, 期望来自前一层接受脉冲的时间是均匀的. 但在现实情况中, 脉冲传递到深层时间不均匀, 这将会导致比预期产生较多或者较少的脉冲. 设现在在 $l-1$ 层的两个突触前神经元连接到一个 l 层的一个突触后神经元, 且有 $W^l = [2 \ -1]$, $a^{l-1} = \phi^{l-1} = [0.5 \ 0.5]^T$, $\theta^l = \lambda^l = 1, T = 6$. 对于 ANN, 由式 (16) 可得到 ANN 的输出 0.5. 对于 SNN, 如图 6.

假设两个脉冲神经元在 6 个时间步长内都分别发射 3 个脉冲, 分为以下 3 种情况.

1) 两个突触前神经元分别在 $t=1, 3, 5$ 和 $t=2, 4,$

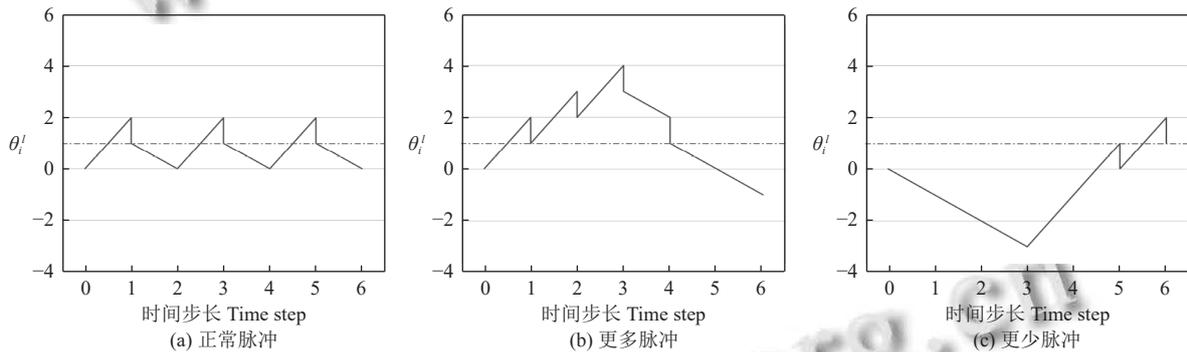


图 6 不均匀误差

$clip$ 函数通常用于限制数值的范围, 式 (20) 中的 $clip$ 函数中括号内为接受的 3 个参数, 第 1 个参数为待限制的数值, 第 2 个参数为函数的下界, 第 3 个参数为函数的上界. $clip$ 函数会返回待限制的数值, 但会确保其落在函数指定的下界和上界之间. 如果待限制的数值小于下界, 则函数返回下界; 如果待限制的数值大于上界, 则函数返回上界; 否则返回待限制的数值本身.

估计的转换误差为:

$$\bar{E}^l = \theta^l clip\left(\frac{1}{T} \left[\frac{z^l T + V^l(0)}{\theta^l} \right], 0, 1\right) - g(z^l) \quad (21)$$

2.5 Spiking YOLOv3 模型优化

在优化 Spiking YOLOv3 模型中, 我们采用了量化步长^[18]的方法. 优化后的模型为 Spiking YOLOv3-FC. 将 ANN 中的 ReLU 激活函数替换为带偏移项和量化

6 处发出脉冲, 如图 6(a) 所示, 则 l 层突触后神经元在 $t=1, 3, 5$ 时发出脉冲, 则 SNN 的输出 $\phi^l(T) = 0.5 = a^l$, SNN 与 ANN 的输出相同.

2) 两个突触前神经元分别在 $t=1, 2, 3$ 和 $t=4, 5, 6$ 处发出脉冲, 如图 6(b) 所示, 则 l 层突触后神经元在 $t=1, 2, 3, 4$ 时发出脉冲, SNN 的输出 $\phi^l(T) = 0.6 > a^l$, SNN 大于 ANN 的输出.

3) 两个突触前神经元分别在 $t=4, 5, 6$ 和 $t=1, 2, 3$ 处发出脉冲, 如图 6(c) 所示, 则 l 层突触后神经元在 $t=5, 6$ 时发出脉冲, SNN 的输出 $\phi^l(T) = 0.3 < a^l$, SNN 小于 ANN 的输出.

上述的这 3 种误差存在着相互依存的关系, 若 $V^l(T) \in [0, \theta^l]$, 不均匀误差将退化为量化误差. 假设 $V^l(T) \in [0, \theta^l]$, 则可以得到:

$$\phi^l(T) \approx \theta^l clip\left(\frac{1}{T} \left[\frac{z^l T + V^l(0)}{\theta^l} \right], 0, 1\right) \quad (20)$$

步长 L 的激活函数, 其公式如式 (22) 所示:

$$a^l = g(z^l) = \lambda^l clip\left(\frac{1}{L} \left[\frac{z^l L}{\lambda^l} + \frac{1}{2} \right], 0, 1\right) \quad (22)$$

在式 (22) 中, 超参数 L 是 ANN 中的量化步长, λ^l 是可训练的参数其决定了 ANN 中 a^l 的最大值, 该参数映射到 SNN 中 $\phi^l(T)$ 的最大值, 即 SNN 中的阈值, 从而能在 SNN 中实现动态阈值的效果, 让脉冲神经元有更加合适的发射率以提高目标检测任务的精度.

当 ANN 中的 L 与 SNN 中的 T 相等, 且初始膜电位是阈值的一半时, 将式 (22) 代入式 (21) 中, 其转化误差为 0, 理论上可以实现 ANN 到 SNN 的无损转换.

对于用量化步长函数的训练过程, 对该函数的导数使用直通估计器^[19], 即式 (23), 使用式 (24)、式 (25) 的总体推导规则后可以用随机梯度下降算法来训练 Spiking

YOLOv3-FC 模型.

$$\frac{d[x]}{dx} = 1 \tag{23}$$

$$\frac{\partial \hat{g}_i(z^l)}{\partial z_i^l} = \begin{cases} 1, & -\frac{\lambda^l}{2L} < z_i^l < \lambda^l - \frac{\lambda^l}{2L} \\ 0, & \text{else} \end{cases} \tag{24}$$

$$\frac{\partial \hat{g}_i(z^l)}{\partial \lambda^l} = \begin{cases} \frac{1}{2L}, & -\frac{\lambda^l}{2L} < z_i^l < \lambda^l - \frac{\lambda^l}{2L} \\ -\frac{z_i^l}{(\lambda^l)^2}, & \text{else} \end{cases} \tag{25}$$

3 实验

3.1 实验数据集与设计

本文使用了 3 个公开且不同规模的目标检测数据集 UAV, PASCAL VOC, MS COCO. 其中, UAV 是一个无人机数据集, 总共有 1 000 多张无人机图片, 包含多个场景下的无人机. PASCAL VOC 是 PASCAL VOC 挑战赛的数据集, 是一个中等规模的数据集, 共有 20 个类别. MS COCO 共有 330 万张图片, 有 80 个目标类别. 我们将数据集的图像的分辨率统一缩放成 416×416, 并对数据集使用了输入图像使用随机剪切框 (cutout) 和图片拼接 (mosaic) 来进行数据增强.

在训练前, 对 3 个数据集用 K-means 聚类算法得到 6 个先验框中心, 3 个数据集的先验框中心如表 2 所示.

表 2 3 个数据集的先验框中心

数据集	框中心					
	1	2	3	4	5	6
UAV	8, 9	10, 12	13, 12	14, 15	17, 20	23, 26
VOC	24, 44	54, 89	82, 199	144, 123	179, 268	346, 334
COCO	10, 14	23, 27	37, 58	81, 82	135, 169	344, 319

本文所有的实验均在 Python 3.7 环境下, 基于 PyTorch 的深度学习框架上进行编码, 使用 NVIDIA GPU 进行训练和推理. 本文将所提出的 Spiking YOLOv3 网络和进一步优化的 Spiking YOLOv3-CF 网络与 Kim 提出的 Spiking YOLO 网络模型分别在 PASCAL VOC, MS COCO 上作了对比, Spiking YOLOv3 网络模型和 Spiking YOLOv3-CF 网络模型在 UAV 数据集上作了对比. 训练时 UAV, PASCAL VOC, MS COCO 数据集的量化步长 L 分别设置为 128, 256, 256, 优化器选择为 AdamW, 基础学习率设置为 0.001, 权重衰减设置

为 0.000 5, 动量设置为 0.9, 批次大小设置为 16, 共训练 300 轮.

3.2 实验结果与分析

本文提出的 Spiking YOLOv3 模型和 Spiking YOLOv3-CF 模型与 Spiking YOLO 分别在 PASCAL VOC 和 MS COCO 数据集上进行了对比实验. 表 3 为 PASCAL VOC 数据集上的实验结果, 表 4 为 MS COCO 数据集上的结果. 表 3 和表 4 中 Spiking YOLO^[10], Spiking YOLOv3, Spiking YOLOv3-CF 网络模型分别简称为 SY, SY3, SY3-CF.

表 3 3 类 SNN 模型在 PASCAL VOC 数据集上基于不同步长 (T) 的全类平均精度 (mAP) 对比 (%)

SNN	$T=64$	$T=128$	$T=256$	$T=512$	$T=1024$	$T=2048$
SY	6.89	18.58	36.22	46.66	47.92	49.98
SY3	36.79	43.16	47.49	50.13	53.96	55.48
SY3-CF	39.60	48.25	52.64	54.59	56.71	56.73

表 4 3 类 SNN 模型在 MS COCO 数据集上基于不同步长 (T) 的全类平均精度 (mAP) 对比 (%)

SNN	$T=64$	$T=128$	$T=256$	$T=512$	$T=1024$	$T=2048$
SY	0.45	6.59	15.08	21.05	23.15	24.10
SY3	13.22	16.03	21.13	24.76	29.93	33.50
SY3-CF	16.48	20.59	26.68	30.91	35.36	37.85

Spiking YOLO, Spiking YOLOv3, Spiking YOLOv3-CF 模型的原 ANN 模型 (转换之前的深度学习模型) 在 PASCAL VOC 数据上精度分别为 51.83%, 56.50%, 56.70%, 在 MS COCO 数据集的精度分别为 26.24%, 38.70%, 38.78%. 当时间步长为 2 048 时, Spiking YOLOv3-CF 模型在 VOC 数据集上实现原 ANN 模型的精度, 在 COCO 数据集上实现的精度有略微的下降.

在精度方面, 在时间步长为 2 048 时, Spiking YOLOv3 和 Spiking YOLOv3-CF 网络模型在 PASCAL VOC 数据集上可以实现 55.48% 和 56.73% 的精度, 两个模型实现精度比 Spiking YOLO 模型分别提升了 5.5%, 6.75%. 在 MS COCO 数据集上可以实现 33.50% 和 37.85% 的精度, 两个模型实现精度比 Spiking YOLO 模型分别提升了 9.4%, 13.75%.

在推理时延方面, 本文所提出的 SNN 模在短时间步长上具有显著的优势. 尤其是 Spiking YOLOv3-FC 模型, 在短时间步长内就能达到较高精度. 在 PASCAL VOC 数据集上, 当 Spiking YOLOv3-FC 模型实现精度为 48.25% 时, 所需时间步长为 128. 相比之下, Spi-

king YOLOv3 模型实现该精度所需时间步长约为 300, Spiking YOLO 模型需要约 1500 个时间步长. 在 COCO 数据集上, 当 Spiking YOLOv3-FC 模型实现精度为 20.59% 时, 所需时间步长为 128. 相比之下, Spiking YOLOv3 模型实现该精度所需时间步长约为 200, Spiking YOLO^[10]模型需要约 500 个时间步长.

在精度转换误差方面, 在 PASCAL VOC 数据集上, 当时间步长为 1024 时, Spiking YOLOv3-FC 模型实现了无损转换. 在 COCO 数据集上, 当时间步长 T 为 2048, Spiking YOLO, Spiking YOLOv3 两个模型的精度转换误差为 2.14%, 7.2%, 而 Spiking YOLOv3-FC 模型的精度转换误差减小至 0.93%.

Spiking YOLOv3, Spiking YOLOv3-CF 模型在原 ANN 上该数据集的精度分别为 99.10%, 99.12%, 当时间步长为 2048 时, Spiking YOLOv3-FC 模型实现精度比原 ANN 实现的精度要高.

表 5 为本文所提出的两个 SNN 模型在 UAV 小目

标检测数据集上的结果, Spiking YOLOv3-FC 模型相比于 Spiking YOLOv3 模型精度提升了 1.05%. 优化后模型在达到相同精度情况下, 约只需优化前 1/2 的时间步长, 加速了推理过程.

表 5 两类 SNN 模型在 UAV 数据集上基于不同步长 (T) 的全类平均精度 (mAP) 对比 (%)

SNN	$T=64$	$T=128$	$T=256$	$T=512$	$T=1024$	$T=2048$
SY3	79.01	87.04	90.59	93.93	97.39	98.20
SY3-CF	83.30	95.55	96.69	97.95	99.14	99.25

图 7 是不同 SNN 模型在 PASCAL VOC 数据集不同时间步长上可视化的对比, 可以看出 Spiking YOLOv3-FC 模型有在短时间内准确检测多个目标的优势, 而 Spiking YOLO 在时间步长较小时, 有漏检的情况, 需要比较长的时间步长才能检测多个目标, 这并不利于目标检测的实际应用. 我们所提出的 Spiking YOLOv3-FC 模型能在低时延情况下实现高效的目标检测, 同时也对部署在神经形态的硬件上友好.

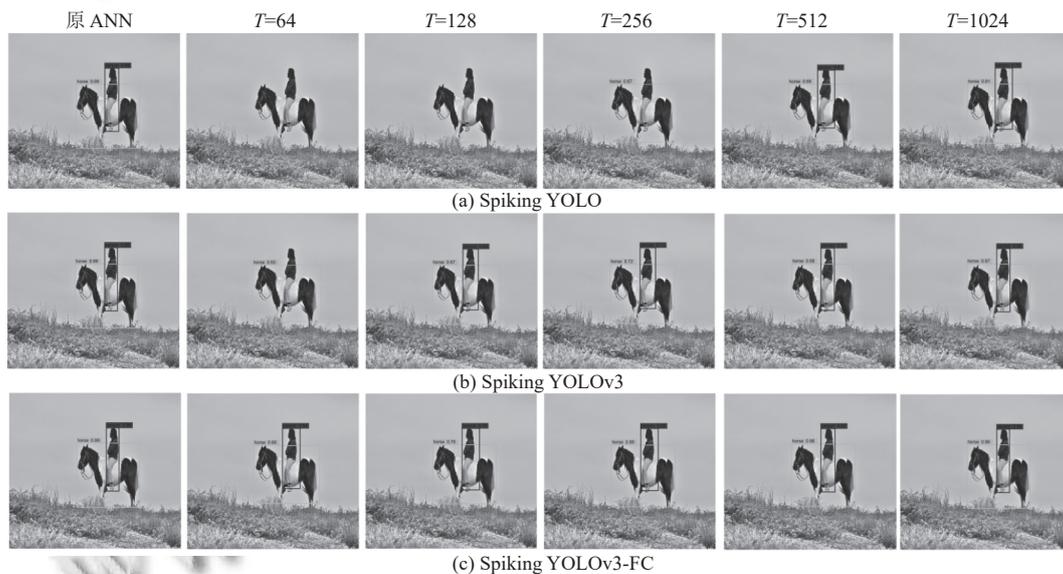


图 7 3 类 SNN 模型在 PASCAL VOC 数据集上基于不同步长的可视化对比

我们记录了时间步长为 256 时, Spiking YOLOv3 和 Spiking YOLOv3-FC 模型在 VOC 数据集上, 网络的脉冲发射频率的结果, 如图 8 所示. 可以看出, Spiking YOLOv3 和 Spiking YOLOv3-FC 模型的脉冲发射频率在区间 $[0.009\ 18, 0.046\ 02]$ 和区间 $[0.005\ 89, 0.033\ 41]$ 之间变化, 且 Spiking YOLOv3-FC 模型的脉冲发射频率整体要小于 Spiking YOLOv3 模型. 这表明本文所提出的两个网络模型的稀疏性, 并且 Spiking YOLOv3-FC

在提高准确率的基础上, 比 Spiking YOLOv3 模型具有更少的脉冲数量, 其功耗更低, 所需的时延也更低.

此外, 本文还计算了 Spiking YOLOv3-FC 模型的能量效率, 在 ANN 中, 每个运算计算一个点积, 包括一个乘法和一个加法. 然而, 在 SNN 中, 因其计算是事件驱动的, 在脉冲发射时计算, 在没有脉冲发射的情况下不会发生计算, 每个运算仅包含一次加法, 大大降低了计算开销. 对于 32 位的浮点运算, 其中乘加运算的消

耗为 4.6 pJ, 而积分累加运算消耗仅为 0.9 pJ. 对于 32 位的整数运算, 其中乘法运算的消耗为 3.2 pJ, 而积分累加运算的消耗仅为 0.1 pJ^[20]. 根据 Rathi 等人^[21]提出的计算方法, 本文所提出的 Spiking YOLOv3-FC 检测模型的所需的能耗仅为原 ANN 的 33.50%. 相比于深度学习模型, SNN 在实现相同精度的情况下所需的能耗大大减少, 能更好地部署在硬件, 实现嵌入式系统.

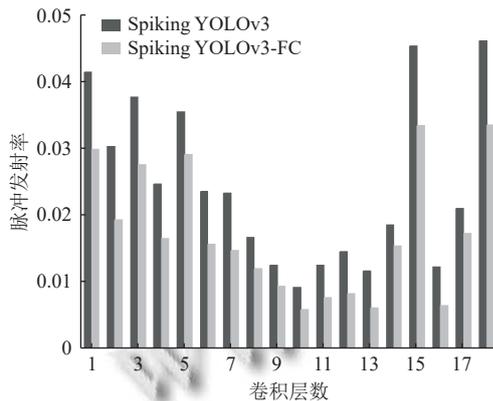


图 8 两类 SNN 模型卷积层脉冲发射率对比

4 结论与展望

本文首先提出了一种对 ANN-SNN 转换友好且完整的转换架构—Spiking YOLOv3 网络模型, 并分析了 ANN-SNN 转换过程中产生的误差, 用量化步长的方法优化 Spiking YOLOv3 网络模型, 进一步提出了能减少转换误差, 且在低时延情况下实现高精度的 Spiking YOLOv3-FC 网络模型. 本文实验从精度, 时间步长, 转换误差, 能耗 4 个方面对比了不同的 SNN 模型的性能. 实验表明, 本文提出的两个网络模型在精度与推理时延上相较于之前的工作 Spiking YOLO 模型, 得到提升, 在相同精度下所需的时间步长更少. 特别是 Spiking YOLOv3-FC 网络模型, 实现了低时延高精度的目标检测任务; 在精度转换误差方面, Spiking YOLOv3-FC 检测模型在 VOC 与 UAV 两个数据集上实现了无损转换, 在 COCO 数据集上将精度转换误差降低到 0.93%; 在能耗方面, Spiking YOLOv3-FC 检测模型相较于深度学习模型, 所需的能耗仅为原来 ANN 的 33.50%, 这对将复杂任务部署在神经形态硬件上具有友好性. 在大型目标检测数据集, 当前模型仍不可能做到完全无损的 ANN-SNN 转换, 后续工作将着力解决在转化过程中产生的不均匀误差问题, 进一步增强 Spiking

YOLOv3-FC 模型的部署效能.

参考文献

- Xu K, Ba JL, Kiros R, *et al.* Show, attend and tell: Neural image caption generation with visual attention. Proceedings of the 32nd International Conference on International Conference on Machine Learning. Lille: JMLR.org, 2015. 2048–2057.
- Hariharan B, Arbeláez P, Girshick R, *et al.* Simultaneous detection and segmentation. Proceedings of the 13th European Conference on Computer Vision. Zurich: Springer, 2014. 297–312.
- Maass W. Networks of spiking neurons: The third generation of neural network models. *Neural Networks*, 1997, 10(9): 1659–1671. [doi: 10.1016/S0893-6080(97)00011-7]
- Wu YJ, Deng L, Li GQ, *et al.* Spatio-temporal backpropagation for training high-performance spiking neural networks. *Frontiers in Neuroscience*, 2018, 12: 331. [doi: 10.3389/fnins.2018.00331]
- Lee JH, Delbruck T, Pfeiffer M. Training deep spiking neural networks using backpropagation. *Frontiers in Neuroscience*, 2016, 10: 508.
- Jin YYZ, Zhang WR, Li P. Hybrid macro/micro level backpropagation for training deep spiking neural networks. Proceedings of the 32nd International Conference on Neural Information Processing Systems. Montréal: Curran Associates Inc., 2018. 7005–7015.
- Sengupta A, Ye YT, Wang R, *et al.* Going deeper in spiking neural networks: VGG and residual architectures. *Frontiers in Neuroscience*, 2019, 13: 95. [doi: 10.3389/fnins.2019.00095]
- Han B, Srinivasan G, Roy K. RMP-SNN: Residual membrane potential neuron for enabling deeper high-accuracy and low-latency spiking neural network. Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Seattle: IEEE, 2020. 13555–13564.
- Deng SK, Gu S. Optimal conversion of conventional artificial neural networks to spiking neural networks. Proceedings of the 9th International Conference on Learning Representations. ICLR, 2021.
- Kim S, Park S, Na B, *et al.* Spiking-YOLO: Spiking neural network for energy-efficient object detection. Proceedings of the 34th AAAI Conference on Artificial Intelligence. New York: AAAI, 2020. 11270–11277.
- Cao YQ, Chen Y, Khosla D. Spiking deep convolutional

- neural networks for energy-efficient object recognition. *International Journal of Computer Vision*, 2015, 113(1): 54–66. [doi: [10.1007/s11263-014-0788-3](https://doi.org/10.1007/s11263-014-0788-3)]
- 12 Diehl PU, Neil D, Binas J, *et al.* Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing. *Proceedings of the 2015 International Joint Conference on Neural Networks*. Killarney: IEEE, 2015. 1–8.
- 13 Rueckauer B, Lungu IA, Hu YH, *et al.* Theory and tools for the conversion of analog to spiking convolutional neural networks. *arXiv:1612.04052*, 2016.
- 14 Ho ND, Chang IJ. TCL: An ANN-to-SNN conversion with trainable clipping layers. *Proceedings of the 58th ACM/IEEE Design Automation Conference*. San Francisco: IEEE, 2021. 793–798.
- 15 Li YH, Deng SK, Dong X, *et al.* A free lunch from ANN: Towards efficient, accurate spiking neural networks calibration. *Proceedings of the 38th International Conference on Machine Learning*. ICML, 2021. 6316–6325.
- 16 Kim S, Park S, Na B, *et al.* Towards fast and accurate object detection in bio-inspired spiking neural networks through Bayesian optimization. *IEEE Access*, 2021, 9: 2633–2643. [doi: [10.1109/ACCESS.2020.3047071](https://doi.org/10.1109/ACCESS.2020.3047071)]
- 17 Na B, Mok J, Park S, *et al.* AutoSNN: Towards energy-efficient spiking neural networks. *Proceedings of the 39th International Conference on Machine Learning*. Baltimore: ICML, 2022. 16253–16269.
- 18 Bu T, Fang W, Ding JH, *et al.* Optimal ANN-SNN conversion for high-accuracy and ultra-low-latency spiking neural networks. *Proceedings of the 10th International Conference on Learning Representations*. ICLR, 2022.
- 19 Bengio Y, Léonard N, Courville A. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv:1308.3432*, 2013.
- 20 Horowitz M. 1.1 Computing’s energy problem (and what we can do about it). *Proceedings of the 2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers*. San Francisco: IEEE, 2014. 10–14.
- 21 Rathi N, Roy K. DIET-SNN: Direct input encoding with leakage and threshold optimization in deep spiking neural networks. *arXiv:2008.03658*, 2020.

(校对责编: 张重毅)