论文“脉冲深度学习梯度替代算法研究综述(2024/8/29-013640)”的修改说明

感谢诸位审稿人提供的修改意见。这些意见切实地指出了上一个版本文章中存在的问题，我们已经进行了针对性的修改。在正文中，修改后的部分以红色进行标注。在本修改说明中，对于从正文摘录来的文字，原有的内容用蓝色标注，修改的内容用红色标注。下面是对每一个审稿意见的单独回复：

意见 1：第 5 页，“序列(Sequential)图像分类更受认可”不够严谨，因为尚无文献直接指出序列图像分类任务相较于神经形态数据集分类任务，更能验证网络的时域性能。
对意见 1 的修改说明：感谢您严谨的意见，我们已经修改为如下内容：
但 Laxmi 等[69]等指出 N-MNIST 等拍摄静态图片得到的神经形态数据集的时域信息较少，因而对于网络的长期依赖学习能力评估，也有一些研究采用序列(Sequential)图像分类 [70, 71].

意见 2：第 6 页，“基于蒸馏的 SNN 训练中”段首没有空两格。
对意见 2 的修改说明：感谢您严谨的意见，我们已经加上了段首空格。

意见 3：第 13 页，图 5 关于 SNN 功耗计算的内容，可以考虑作为背景知识，移动到第二章。
对意见 3 的修改说明：感谢您对写作方面的中肯建议，我们已经将这一部分内容移动到了第二章，作为背景知识的一部分，请参考原文的“SNN 的能源效率是其突出优势”段落。

意见 4：第 14 页，“普通的 BN 在 SNN 中使用时”这一段上方有多余的空行。
对意见 4 的修改说明：感谢您严谨的意见，我们已经去掉了多余的空行。

意见 5：第 14 页，“时域服用”应该“时域复用”。
对意见 5 的修改说明：感谢您严谨的意见，我们已经修改了错别字。

意见 6：第 15 页，“Li 等[122]则对神经形态数据增强进行了探索”，应该在文中补充其研究中关于神经形态数据的数据增强与普通图像数据增强之间差异的研究结论。
对意见 6 的修改说明：感谢您关于扩充内容的宝贵意见，我们已经增加了相关内容：
ANN 领域用于静态图片上的数据增强方法已经比较成熟，而 Li 等[151]则对神经形态数据集的增强进行了探索，表明剪切、旋转、平移等几何变换适用于神经形态数据；而亮度、色彩、锐度、均衡化等色彩变换会改变事件极性、破坏事件流的稀疏性，不适合使用；其研究结论可以作为神经形态数据增强的经验性准则.

意见 7：第 16 页，“接收到尺寸为”后的尺寸，最好加上括号，以与文字部分有所区分。
对意见 7 的修改说明：感谢您对写作规范的建议，我们已经加上了括号，便于读者区分，请参见原文“网络中的每层可以同时接收到尺寸为 $(T\times N\times ...)$ 的整个序列作为输入”部分。

意见 8：文中缺少对编码方式的影响分析：论文中未深入讨论不同的编码方式（如时间编码、脉冲频率编码）对梯度替代算法性能的影响。编码方式是 SNN 研究中的重要部分，直接影响网络的学习能力和效率，若能补充相关内容，并引用一些相关研究，将有助于更全面地反映梯度替代法在不同应用中的优势和局限性。
对意见 8 的修改说明：感谢您关于扩充内容的宝贵意见。我们非常赞同编码的重要性，已经在第三章单独增加了一节“3.2 编码方式”：

3.2 编码方式

神经编码(Neural Coding)泛指神经元如何将信息表示为电生理活动，是神经科学中的一个重要研究问题.在脉冲深度学习中，该问题细化为如何使用脉冲序列来表示信息，研究话题涵盖如何将非脉冲的输入编码成脉冲，以及 SNN 内部如何使用脉冲传递信息.从类型来看，编码方式可分为两类：频率编码(Rate Coding)和时间编码(Temporal Coding)，前者使用脉冲的发放频率表示数值的大小，而后者则通过脉冲的发放时刻传递信息.

常见的图像、视频、语音等数据都以整数或浮点值形式存储，与 SNN 期望的脉冲形式不符，研究者们提出了多种输入编码方式解决这一问题.泊松编码(Poisson Coding)是频率编码的代表性方法.对于输入 $x \in (0,1)$，标准的泊松编码生成数量符合强度为 x 的泊松分布的脉冲，而简化的实现则用每个时间步中脉冲发放概率均为 x 的二项分布近似.泊松编码在早期的深度 SNN 研究中[82, 83]较多使用.

时间编码方式则通过脉冲的发放时刻来表示信息，首达脉冲编码(Time-To-First-Spike Coding)[84, 85]是其中的典型代表.首达脉冲编码遵循“刺激越强，发放越早”的规则，将输入 $x \in (0,1)$ 转换成对应的发放时刻，即：

$$S[t] = \begin{cases} 1, t = t_f, \\ 0, t \neq t_f, \end{cases} \tag{16}$$

$$t_f = \text{Round}((T-1)\cdot(1-x)), \tag{17}$$

其中 Round 为四舍五入的量化函数，T 为时间步数.

泊松编码对于数值较小的输入很难触发脉冲，随机发放的特性需要较长时间步才能获取稳定结果，因而时间步数多、延迟高、性能低；而首达脉冲编码只能释放单个脉冲，且式(17)中的 Round 函数损失了一定信息，实际性能也欠佳. 目前多数高性能深度 SNN[50-52]都采用直接输入编码[86]方式. 如果输入是静态的图片，而 SNN 需要运行 T 次，则该方法将输入简单重复 T 次得到输入序列.在该输入方式下，连续浮点输入转换为离散二值脉冲的编码实际由首个突触层和脉冲神经元层完成，它们可以视作可学习的编码器[75].Rathi 等[86]的实验结果表明，尽管直接输入编码在首层引入了浮点计算，但相较于泊松编码，该方法的时间步数大幅度降低，因而最终网络的能耗也低于使用泊松编码.

绝大多数 ANN2SNN 方法使用频率编码，SNN 内部通过脉冲的发放频率表示转换前的 ANN 中 ReLU 的激活值；此外，首达脉冲编码[87-89]、相位编码(Phase Coding)[90]和进制编码(Radix Coding)[91]等效率更高的时间编码方式也被逐渐提出.而在基于直接训练的深度 SNN 中，梯度替代法提供了强大的端到端训练能力，因而无需手动设计网络内部的编码方式.从实际表现看，梯度替代法训练出的 SNN 所需的时间步数也远少于由转换法得到的 SNN，效果较好.但理解直接训练的 SNN 内部的编码方式，依然是一个值得探索的方向，现有研究较少.Li 等[92]对基于梯度替代法训练的 SNN 和同结构 ANN 进行了相似性分析，发现 SNN 的特征与 ANN 具有高度相似性，时间维度并未提供太多额外信息；Hu 等[93]则进而发现不同时间步的梯度相似性也很高；以上研究表明现有的深度 SNN 内部的编码方式可能较为接近频率编码.需要指出的是，上述研究使用直接输入编码、纯前馈 SNN，而其结论未必适用时间编码输入或者带有反馈连接的 SNN.

此外，我们也在第五章增加了相关的讨论：

(1) 生物启发的高效神经编码算法设计：生物神经系统中最早被发现的编码方式是频率编码[171]，其后更多证据表明，生物神经系统中还存在更高效的编码方式，例如人类触觉感知系统可以通过单个脉冲的精确发放时刻来编码触感[172]，苍蝇可以凭借 30 毫秒内到达的一

两个脉冲对环境做出快速反应[173].目前爆发编码(Burst Coding)[174, 175]、相位编码[90]等已经用于 ANN2SNN 方法，而在梯度替代学习算法中的应用较少.研究者们可以考虑设计生物启发的高效神经编码算法，并应用于 SNN 内部的信息表征，一方面能够充分利用时域信息以降低时间步数和能耗，另一方面也有望与脑机接口相关研究领域形成合力，加速对大脑工作原理的理解.

意见 9：文中实验和定量分析不足：虽然论文介绍了不同算法，但在同一评测基准上的性能、能耗等方面缺乏详细的定量对比，尤其是在复杂数据集（如 CIFAR-10、ImageNet）上的实验数据不够具体。增加不同算法在相同任务上的定量实验结果，并对其性能差异和能耗进行细致的讨论，将更有力地支持论文的观点，使其综述更具说服力。

对意见 9 的修改说明：感谢您关于扩充内容的宝贵意见，我们已经增加了神经形态语音数据集 SHD 数据集的分类实验、神经形态的 Gen1 数据集的目标检测实验，其中 Gen1 数据集是目前规模最大的神经形态数据集，包括 228123 辆汽车、27658 位行人共 39 小时的拍摄数据。该数据集压缩包大小为 200G，解压后达到 750G，是较为复杂的数据集。对于能耗估算，按照领域内的惯例，我们通过比较突触操作数(Synaptic Operations)来衡量不同方法的能耗。在静态/序列 CIFAR 分类、SHD 分类和 Gen1 目标检测任务上，我们都增加了各个方法的突触操作数测量。此外，我们在 CIFAR 数据集上增加了训练速度、推理速度、GPU 显存消耗、突触操作数指标，以进行全面比较。图 8 和表 4 展示了对比结果。此外，之前的 OSR 方法实现代码有一些小的错误，我们进行了修复并更新了实验结果。新增的内容如下：

图 8(b)展示了各类方法的训练速度，可以发现基于 CuPy 后端加速的 IF 和 LIF 神经元速

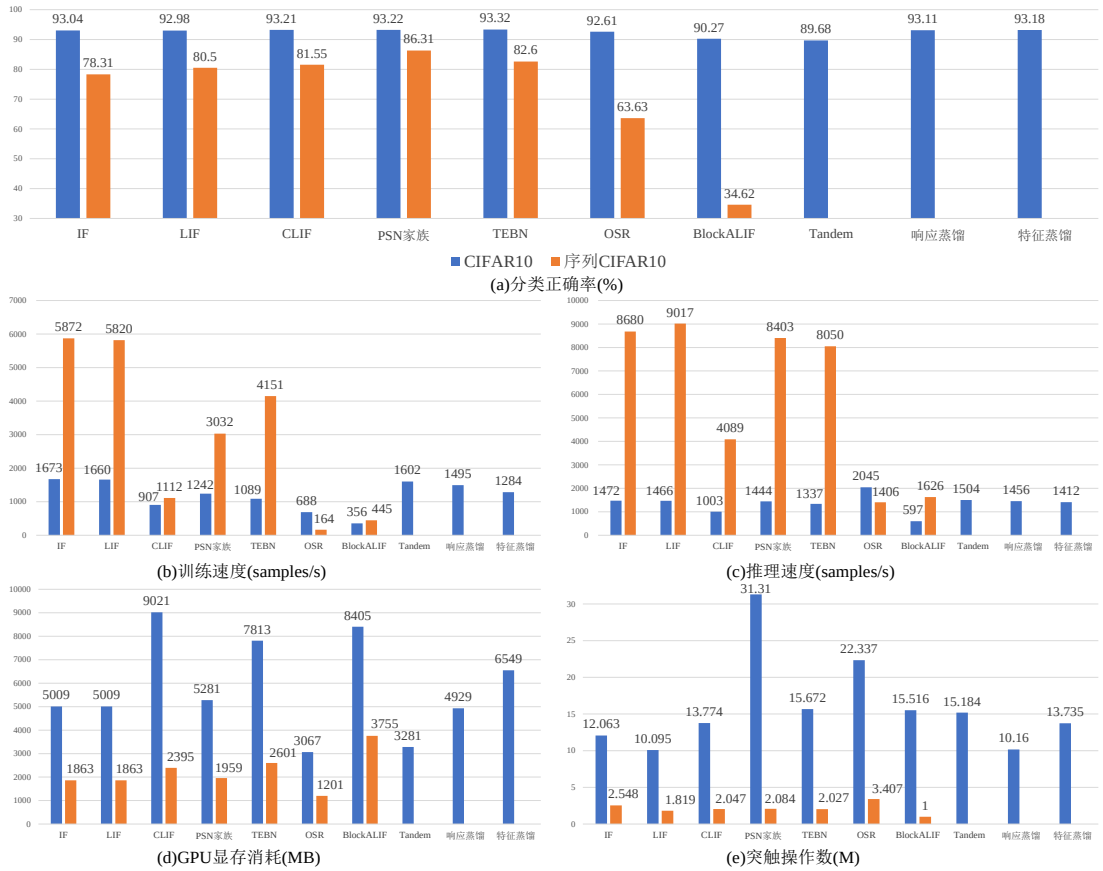

图 8 CIFAR 图片分类性能对比

度最快，体现了 SpikingJelly 框架中融合 CUDA 内核带来的巨大优势.令人意外的是，PSN 家族的速度慢于 CuPy 后端加速的 IF 和 LIF 神经元，这可能是由于该实验设置下，批量数为 128、网络通道数为 256，内存读写需求较大，PSN 家族的核心运算矩阵乘法遭遇了内存瓶颈(Memory-Bound).通过额外实验发现，设置批量大小 32、网络通道数 32 时，内存读写需求大大降低，此时 PSN 训练速度达到 2345samples/s，高于 CuPy 后端加速的 IF 神经元 1649 samples/s 的训练速度，佐证了前述猜想.TEBN 方法中使用的仍然是 CuPy 后端加速的神经元，但由于逐时刻仿射变换的额外计算而拖慢了速度.OSR 速度大幅度落后于其他方法，因其是在线学习方法，只能使用 SpikingJelly 框架中的逐步(Step-By-Step)传播，而其他方法则可以使用逐层(Layer-By-Layer)传播并通过融合时间批量维度来加速无状态层.BlockALIF 神经元的加速效果较差，主要原因在于神经动态极为复杂，且使用时间上的卷积实现并行计算，而 Fang 等[71]在设计 Sliding PSN 的经验表明卷积并行度较低，速度远慢于矩阵乘法.关于 BlockALIF 神经元的加速效果，将在后文予以详细讨论. Tandem 方法速度较慢，原因在于其需要 ANN 部分的计算，且神经元使用纯 PyTorch 实现，速度远不如 CuPy 实现.蒸馏方法中仍然使用 CuPy 后端加速的 IF 神经元，但由于 ANN 部分的计算和蒸馏损失拖累了速度，其中特征蒸馏需要更多的损失项，因而速度比响应蒸馏更慢.CLIF 神经元的作者也是使用 PyTorch 实现，且神经动态较为复杂，故速度较慢.图 8(c)展示了各类方法的推理速度，整体和训练速度一致，不再赘述.

图 8(d)展示了各类方法在训练时的 GPU 显存消耗.神经网络在训练时，需要保存权重和各种计算的中间变量以用于反向传播.权重的数量在网络结构确定后就固定了，因而中间变量的数量对内存消耗起决定性作用.使用 BPTT 训练的 SNN，需要保存所有时刻的中间变量，而在线学习方法只需要保存单个时间步的中间变量，因而 OSR 方法是各类方法中消耗内存最少的.Tandem 方法的反向传播基于 ANN，也消除了时间维度，故内存消耗量与在线学习方法接近.其他方法基于 BPTT 训练，其中 CLIF 神经元引入了额外的补充电位以及 Sigmoid 激活函数；BlockALIF 神经元具有复杂的神经动态；TEBN 逐时刻的仿射变换使 BN 层的计算变为原来的 T 倍；三者都引入了大量中间变量，因而带来了较大的内存消耗.蒸馏类方法一方面需要 ANN 计算，另一方面引入了额外损失，故内存消耗量比直接训练的方法更大，且特征蒸馏的损失项更多，故内存消耗量也显著高于响应蒸馏方法.相较于 IF 神经元和 LIF 神经元，PSN 家族并未引入额外计算或中间变量，因而内存消耗量与它们接近.需要注意的是，尽管 LIF 神经元相较于 IF 神经元神经动态更复杂、中间变量更多，但在 CuPy 后端实现时这些中间变量由 CUDA 内核内部处理，而不是通过 PyTorch 的自动微分机制，因而这些变量在 CUDA 执行完毕后就自动释放了，所以两者的内存消耗完全一致.

图 8(e)展示了各类方法的突触操作数(Synaptic Operations)，在网络结构确定后，该指标主要由神经元的发放率决定. PSN 家族的突触操作数显著高于作为基准的 IF 神经元和 LIF 神经元，表明其去除重置确实导致了发放率的显著升高.OSR 和 TEBN 的突触操作数也较高，需要注意的是前者尽管是在线学习方法，却在静态 CIFAR10 上取得了接近 BPTT 类方法的正确率，而后者则取得了静态 CIFAR10 的最高正确率，可能是它们在训练过程中通过提升发放率来获得了性能增益. 在该任务中，作为基准的 IF 神经元和 LIF 神经元每层脉冲神经元的发放率都低于 50%，而 Guo 等[169]则猜想发放率越接近 50%则信息熵越大，网络性能越好，OSR 和 TEBN 的表现与其猜想一致.其他方法的突触操作数则较为接近，但也存在一定差异.例如，LIF 神经元低于 IF 神经元，表明膜电位的泄露行为降低了整体的发放率.Tandem 方法使用脉冲频率传递信息，而特征蒸馏则是直接拟合 ANN 的 ReLU 的输出，两者都暗含使用频率编码表示信息，因而发放率和突触操作数稍高；而响应蒸馏中，由于蒸馏温度以及 ANN 输出的“暗知识(Dark Knowledge)”，导致拟合目标不是理想的独热编码形式，损失计算相较于仅使用真实标签计算交叉熵时有所松弛，可能是这一特性导致其突触操作数反而低

于作为基准的 IF 神经元.BlockALIF 神经元在序列 CIFAR10 分类的突触操作数极低，考虑到其正确率只有 34.62%，表明确实是时间维度分组限制发放导致的性能差.

4.2 神经形态语音数据SHD分类性能

本文使用神经形态的 SHD 语音数据集[68]进行实验，并使用 Ilyass 等[101]的开源代码和网络结构.需要注意的是，该实验使用固定时间间隔积分来处理输入事件[49]，因而输入的帧数为 88~126.

实验结果展示在表 4 中.在正确率方面，该数据集上各种方法的表现与 CIFAR 分类存在较大差异.作为基准的 IF 神经元和 LIF 神经元，前者性能高于后者，而在常见的神经形态数据集分类任务中则通常是 LIF 神经元性能更好，这可能暗示了语音数据和视觉数据存在较大差异.TEBN 与性能更好的 IF 神经元配合使用，但正确率低于作为基准的 IF 神经元，可能是其难以处理变长序列的输入所致.BlockALIF 神经元性能最好.通过额外的实验发现，当去除 BlockALIF 神经元的自适应的阈值后，其性能会下降到和 IF 神经元持平，表明自适应阈值使其较好地捕捉了音频数据中的时域信息.其他方法都表现较差，低于作为基准的 IF 神经元，尤其是 OSR 方法，可能是该任务的时间步较大，因而在线学习方法相较于 BPTT 难以补偿时间梯度的缺失.而在突触操作数方面，由于网络本身规模很小且层数浅，故各类方法的差别不大.

4.3 神经形态数据集Gen1目标检测性能

本文使用神经形态的 Gen1[133]数据集进行目标检测任务训练，该数据集由事件相机采集的驾驶场景组成，相较于静态图片分类任务难度更大，极具挑战性.本文使用 Fan 等[139]的开源代码和网络结构，使用 Spiking DenseNet121-16[135]作为检测的骨干网络，首先在神经形态的 NCAR[136]数据集上预训练分类任务，然后在 Gen1 数据集上进行目标检测任务训练并记录最终的 mAP.

实验结果展示在表 4 中.NCAR 预训练结果显示，LIF 神经元优于 IF 神经元，也强于 CLIF 神经元、Sliding PSN、TEBN.OSR 和 BlockALIF 神经元性能都较差.需要指出的是，根据 Fan 等[139]的设计，预训练设置了早停(Ealy Stopping)，连续 5 轮训练的最小验证集损失不减少则退出训练.该预训练仅为后续目标检测任务提供较优的初始参数，因而本身的分类正确率无需过多关注.从目标检测性能看，性能排序为 TEBN > Sliding PSN > LIF > CLIF > BlockALIF > IF > OSR，且 IF 神经元和 OSR 几乎无精度，表明训练完全不收敛.对于能够正常收敛的方法，突触操作数排序为 TEBN < BlockALIF < Sliding PSN < LIF.整体来看，TEBN 在该检测任务中表现极好，精度最高、突触操作数最低，可能是其凭借逐时刻的仿射变换增强了网络对时序数据的拟合能力；Sliding PSN 和 LIF 神经元表现稳定；IF 神经元不收敛，可能是其仅积分不作衰减的特性难以捕捉时序动态所致；CLIF 神经元和 BlockALIF 神经元性能都较差，它们都使用了自适应的阈值更新机制，可能表明该机制未必总对性能有益；OSR 方法不收敛，可能是该任务的时序信息较多，在线学习方法在缺失完整梯度的情况下难以训练参数.

表 4　对比各类代表性方法任务神经形态数据集处理任务性能

评估指标\方法	IF	LIF	CLIF	Sliding PSN	TEBN	OSR	BlockALIF
SHD 分类正确率(%)	78.33	66.98	66.1	69.23	74.54	40.93	82.36
SHD 分类突触操作数(M)	0.0251	0.0262	0.0263	0.0264	0.0262	0.0228	0.0239
NCAR 预训练分类正确率(%)	81.00	91.69	85.34	91.41	90.96	56.23	68.14
Gen1 目标检测 mAP@0.5	0.0013	0.4399	0.2388	0.4781	0.5213	0.0007	0.1347
Gen1 目标检测 mAP@0.5:0.95	0.0002	0.2106	0.090204	0.2391	0.2656	0.0003	0.0457
Gen1 目标检测突触操作数(M)	385.86	296.33	353.03	293.53	214.72	305.93	232.69

意见 10：文中未充分讨论硬件实现中的挑战：作为第三代神经网络模型，SNN 的硬件实现是关键环节。论文虽然提到了一些与神经形态计算相关的研究，但未深入讨论梯度替代算法在硬件实现中的具体挑战（如计算复杂度、能耗）以及可能的优化方向。补充在文中这方面的内容，将更好地反映梯度替代法在实际部署中的可行性和潜在改进措施。

对意见 10 的修改说明：感谢您关于扩充内容的宝贵意见。我们非常赞同您对硬件实现挑战的关注，由于 SNN 的目标设备是神经形态计算芯片，而非训练时使用的 CPU/GPU，在设计软件算法时应该充分考虑硬件特性。在我们的上一个版本论文中有一些硬件的讨论，这些内容在目前的新版文章中得到了保留，如下所示：

尽管 PLIF 神经元和 GLIF 神经元神经动态中都使用了 Sigmoid 函数，但该函数只用于包装可学习参数，其输出在训练完成后是常数，神经元推理时并不需要计算；而 CLIF 神经元的式(22)和式(23)中 Sigmoid 函数的输入是依赖于数据的，不能在推理时去除.Sigmoid 函数复杂的指数计算，可能带来 CLIF 神经元较高的硬件实现代价.

SEW ResNet 和 MS ResNet 都解决了深度 SNN 的退化问题，但同时也引入了新的问题.SEW ResNet 主要使用实验性能最好的加法来连接残差块的输入和最后一个 SN 的输出脉冲，导致残差块输出的实际上是脉冲之和，是非负整数而非二值脉冲，这可能丧失了 SNN 的二值特性以及对应的硬件实现时免乘法器的优势；MS ResNet 则是使用残差连接在网络层之间传递稠密的浮点值，破坏了 SNN 事件驱动通信的特性，难以在异步芯片实现.

此外，我们还在第五章增加了专门的小节，讨论梯度替代算法在硬件实现中的挑战：

(5) 软件仿真和硬件部署的沟壑：SNN 的目标运行设备是神经形态计算芯片，但现有学习算法更多关注于软件仿真，而对硬件部署面临的挑战关注不够.首先总结硬件相关的研究问题如下：

(a)模型量化(Model Quantization)：GPU 通常配备 GB 级别的显存，使用 float32 精度的突触，但神经形态计算芯片上资源有限，例如 Loihi[35]至多支持 9 比特表示的突触权重.已经有部分研究者在中等规模 SNN 上进行了量化并达到较好效果[184-187]，其中 Wei 等[187]将权重量化到 1 比特、膜电位量化到 2 比特，在 CIFAR100、Tiny ImageNet[188]等多个中等规模的数据集上达到了较好性能，展现出低精度 SNN 部署的巨大潜力.

(b)网络剪枝(Network Pruning)：神经形态计算芯片上的内存容量有限，无法容纳太多突触和神经元，例如 Loihi 至多支持存储空间不超过 16MB 的突触权重和 12800 个神经元，而典型的 VGG11 网络则含有 132.86M 的突触数，如果是以 float32 精度实现则需要 531.44MB 存储空间，远超 Loihi 的容纳能力.对突触和神经元进行剪枝可以大幅度降低模型大小.目前 SNN 中的剪枝技术可以分为两类，一类是通用剪枝技术[189-194]，可以同时用于 ANN 和 SNN；另一类则是 SNN 独有的，例如基于突触可塑性的剪枝方法[195-198]、基于神经元发放率的剪枝方法[199, 200]等.

(c)硬件非理想性(Hardware Non-idealities)：基于模拟电路/数模混合电路或忆阻器实现的神经形态芯片，例如 Neurogrid[201]等，受限于制作工艺和自身特性，其运行 SNN 时存在一定噪声，输出结果与在 CPU/GPU 上的仿真存在一定差异.目前有少量研究者对这一问题进行了探索.Bhattacharjee 等[202]发现 SNN 多步运行的特性在忆阻器上会累计误差，并通过利用 BN 的统计量来记录噪声，减少软件仿真和硬件部署的差异. Moro 等[203]则是通过训练时注入噪声来模拟硬件运行的环境，使硬件推理和软件模拟的结果更一致. Christensen 等[204]指出，设计“硬件感知”的软件学习算法，即在软件中模拟硬件特性，是解决硬件非理想性的关键途径.

(d)同步仿真和异步部署：深度 SNN 中通常使用元素取值仅为 0 或 1 的张量表示脉冲，多个

时刻的脉冲序列类似于视频帧的表示格式；而 DVS 相机和 Loihi、Speck[54]等基于异步电路实现的神经形态芯片则是使用 AER 协议表示脉冲事件，两者的差异导致 CPU/GPU 上的训练和神经形态芯片上推理的精度不一致.尽管增大时间步数可以降低两者的差异，但训练代价会显著增加.关于这一问题的研究还较少，Yao 等[54]通过在单步释放多个脉冲来降低误差并在 Speck 芯片上进行了验证. 设计异步仿真算法可能是解决这一问题的关键，目前已经有初步工作[205]进行了尝试.

现有的梯度替代算法，通常使用任务性能和理论功耗作为评估指标，而对硬件部署的可行性未作过多考虑.未来的研究者们可以更多地考虑硬件约束，例如在有限内存和精度的条件下设计高性能 SNN 神经元和网络架构、软件模拟硬件特性等，将模型量化、网络剪枝等技术引入，同时充分考虑硬件非理想性、同步仿真和异步部署的差异，进行软件-硬件协同设计(Software Hardware Codesign)，提升 SNN 的片上推理性能，这一设计理念也是诸多 SNN 研究者所倡导的[31, 204].

意见 11：本文只讨论了直训 SNN 在图象分类方面的算法进展，可以适当加一些目标检修以及图像分割等领域的模型探讨和实验结果。例如，以 Transformer 为基本架构的图像分类模型可以作为目标检测算法的 backbone，以及有较多基于 Resnet 结构的 SNN 目标检测算法被提出，并在 COCO 等数据集上取得不错的效果。

对意见 11 的修改说明：感谢您关于扩充内容的宝贵意见，我们已经增加了相关讨论：

受益于 SNN 网络结构的快速发展，目前梯度替代法也逐渐应用于难度较高的目标检测任务，这一任务曾经主要由 ANN2SNN 方法主导[130, 131].目标检测任务通常使用 mean Average Precision (mAP)作为性能指标，其中 mAP@0.5:0.95 表示按照 Intersection over Union (IOU)阈值从 0.50 到 0.95，以 0.05 为步长计算出的平均 mAP；mAP@0.5 表示按照 IOU 为 0.5 计算出的 mAP.常用的数据集包括传统的静态图片目标检测数据集 COCO[132]和神经形态的 Gen1[133]数据集等.Lo¨ic 等[134]最早成功将梯度替代法训练的 Spiking DenseNet[135]用于 Gen1 数据集上的目标检测任务.他们的主要创新之处在于，首先在神经形态的 NCAR[136]数据集上进行分类任务的预训练，然后再到 Gen1 数据集上进行目标检测任务的训练.Zhang 等[137]在 SEW 残差连接[50]上增加了二元门控，使得残差块可以完全变成堆叠的卷积层或恒等变换，保持纯二值输出；他们还在 SNN 的输出和检测头之间增加了注意力模块以更好的提取时空特征.Su 等[138]对 SNN 中的残差结构进行改进，使用最大池化替换原来下采样残差块中的步幅大于 1 的卷积，通过拼接(Concat)操作实现残差连接，同样保持了纯二值输出.Yao 等[122]提出的 Spike-Driven Transformer V2 在前文已有介绍，该网络结构也在目标检测任务上进行了验证.他们首先在 ImageNet 数据集上预训练，然后再到 COCO 数据集进行目标检测训练，最终仅使用 1 个时间步就超越了前述其他研究方法.Fan 等[139]基于 Lo¨ic 等[134]的方法，将网络中间层输出的不同尺度的特征，使用类似特征金字塔(Feature Pyramid)[140]的方式进行融合，大幅度提升了性能.表 2 中对这些方法以“时间步数|性能”的格式进行了统计汇总.

表 2　梯度替代法训练的 SNN 目标检测时间步数和性能

数据集	COCO		Gen1	
方法	mAP@0.5	mAP@0.5:0.95	mAP@0.5	mAP@0.5:0.95
Lo¨ic 等[134]		5 | 0.19		5 | 0.19
Zhang 等[137]	4 | 0.296			
Su 等[138]	4 | 0.501		5| 0.547	5 | 0.267
Yao 等[122]	1 | 0.512			
Fan 等[139]			5 | 0.593	5 | 0.321

此外，我们还增加了目标检测的相关对比实验，请参考我们对意见 9 的修改说明。

意见 12：本文提到设置了统一的实验环境来对不同方法进行横向对比，但具体实验细节和参数设定应该在文中更加详细地说明，以便读者能够复现实验结果。此外，除了性能对比外，文中还可以加入一些对模型复杂度、训练时间和计算资源需求等方面的比较。

对意见 12 的修改说明：感谢您关于实验细节和参数的宝贵意见，我们非常欢迎读者能够复现和进一步研究。我们在附录中提供了所有实验代码和日志的下载链接：

本文的实验代码、训练日志可以从如下网址获取，方便读者自行复现或进一步研究：
https://disk.pku.edu.cn/link/AA2810D5D934B048DE867A3925B6FBF6BA

我们还增加了详细的超参数说明，参见我们在参考文献后增加的附录：

2.1 突触操作数统计

突触操作数的统计是在整个测试集上完成的，汇报的结果为平均到每个样本的突触操作数.需要注意的是，当突触处理非脉冲的整数类型输入时，例如神经形态数据集中的事件积分得到的帧，或由 SEW 残差连接导致的脉冲之和，突触操作数按照输入整数的数值计算.例如，当输入为脉冲[0,1,0,1]则突触操作数为 2；而输入为非负整数[0,2,0,3]则突触操作数按 5 计算.

2.2 CIFAR 分类实验细节

本文使用 Fang 等[71]的网络结构，记卷积层为 Conv，批量标准化层为 BN，脉冲神经元层为 SN，平均池化层为 AP，全连接层为 FC，则网络结构为：

(Conv-BN-SN)-(Conv-BN-SN)-(Conv-BN-SN)-AP-(Conv-BN-SN)-(Conv-BN-SN)-(Conv-BN-SN)-AP-FC-SN-FC,

其中卷积层的通道数固定为 128.对于静态 CIFAR10 分类任务，使用 2 维卷积，第一个 FC 层输出特征数为 2048；对于序列 CIFAR10 分类任务，使用 1 维卷积，第一个 FC 层输出特征数为 256.两个任务的网络结构中第二个 FC 层输出特征均为 10，与总类别数一致.

数据增强方法包括 Mixup($p=1,\alpha=0.2$)、Cutmix($p=1,\alpha=1$)、随机擦除($p=0.1$)、自动数据增强(Trivial Augment)、标签平滑($p=0.1$)、数据标准化.

对于训练超参数，在不做出特殊说明的情况下，默认批量大小为 128、训练轮数 256、使用动量大小为 0.9 的 SGD 优化器、学习率 0.1、混合精度训练、周期为训练轮数的余弦退火学习率调节器[212].对于静态 CIFAR10 分类任务，时间步数为 4；对于序列 CIFAR10 分类任务，时间步数与图像宽度相同，为 32.

对于不同方法，额外调节的超参数通常是通过网格搜索确定的，具体如下：

响应蒸馏：蒸馏损失的强度为 0.001，温度为 4.作为教师的 ANN 是将 SNN 中脉冲神经元换成 ReLU 激活函数，并使用默认超参数训练出来的.

特征蒸馏：蒸馏损失的强度为 0.1，温度为 1. 作为教师的 ANN 与响应蒸馏中使用的是同一个 ANN.

Tandem：使用学习率为 0.001.

PSN 家族：静态 CIFAR10 分类任务使用 PSN，序列 CIFAR10 分类任务使用 $k=4$ 的 Sliding PSN.

Block ALIF：分块大小为 1.使用更大的分块大小，则性能会剧烈下降.

2.3 SHD 分类实验细节

本文使用 Ilyass 等[101]的开源代码和网络结构.对于神经形态的 SHD 语音数据集，使用 SpikingJelly 框架[49]中的固定时间间隔积分方式，每 20ms 积分为一帧，产生帧数为 88~126、特征数为 140 的帧.使用 3 层全连接网络，具体结构为：

(FC256-BN-SN-DP)-(FC256-BN-SN-DP)-FC20-LIF,
其中 FC256 表示输出特征数量为 256 的全连接层，FC20 表示输出特征数量为 20 的全连接层，DP 表示 Dropout 层，且丢弃率均为 0.4.SN 表示脉冲神经元层，根据不同方法使用不同的脉冲神经元，而最后输出层固定为 LIF 神经元层，且阈值设置为无穷大，网络的输出是其所有时间步的膜电位.

对于训练超参数，默认训练 150 轮，对于 LIF 神经元和其他有膜时间常数的神经元均初始化膜时间常数为 1.005，使用 SpikingJelly 框架[49]中 $\alpha=5$ 的 ArcTan 替代函数，使用 Adam 优化器，学习率 0.001，权重衰减 0.00001，使用最大学习率为 0.005、周期为训练轮数的 OneCycle 学习率调节器[213].

对于不同方法，额外调节的超参数如下：
Sliding PSN：使用 $k=3$.
TEBN：由于实验观测到 IF 神经元表现反而比 LIF 神经元好，故使用 IF 神经元配合 TEBN.
BlockALIF：分块大小为 1.

需要注意的是，TEBN 和 BlockALIF 神经元要求固定时间步数，不能处理变长输入，而本实验中使用的固定时间间隔积分方式会产生长度不固定的输入帧.因而，在实验中将它们的时间步数都设置为帧的最长长度 126，并在计算时将长度不足 126 的输入进行 0 填充，输出再重新去除填充部分.

2.4 Gen1 目标检测实验细节

本文使用 Fan 等[139]的开源代码和网络结构，使用 Spiking DenseNet121-16 作为检测的骨干网络，首先在神经形态的 NCAR[136]数据集上预训练分类任务.分类任务的默认超参数为：数据集积分到 5 帧（故时间步数为 5），批量大小为 64，使用 AdamW 优化器，学习率为 0.005，余弦退火学习率调节器（周期和训练轮数一致，最小学习率 0.00001），训练轮数 30 并设置了早停机制（Early Stop，当连续 5 轮训练的最小损失没有改善则退出训练），使用混合精度训练.网络结构为标准的 Spiking DenseNet121-16.

分类任务预训练完成后，卷积层部分权重作为目标检测任务的骨架(backbone)网络的初始权重，并在 Gen1 数据集上进行训练，默认超参数为：数据集积分到 5 帧（故时间步数为 5），训练轮数为 50，使用 AdamW 优化器，学习率 0.001，权重衰减 0.0001，余弦退火学习率调节器（周期和训练轮数一致，最小学习率 0.00001），使用混合精度训练.需要注意的是，BlockALIF 神经元的参数是逐神经元的，而该任务中，预训练分类任务和其后的目标检测任务，输入尺寸不一致.因而加载预训练权重时，卷积层可以正常加载，而 BlockALIF 神经元则由于参数尺寸不匹配无法加载，参数按照默认方式初始化.

对于不同方法，需要额外调节的超参数如下：
CLIF 神经元：学习率 0.0001.
Sliding PSN：使用 $k=3$.
TEBN：实验发现 LIF 神经元好于 IF 神经元，故使用 LIF 神经元配合 TEBN.
OSR：学习率 0.0001.
BlockALIF：学习率 0.0001.

感谢您关于增加其他对比指标的宝贵意见，我们在 CIFAR 数据集上增加了训练速度、推理速度、GPU 显存消耗、突触操作数指标的对比。我们还增加了神经形态的 SHD 语音数据分类和神经形态的 Gen1 目标检测任务性能以及突触操作数对比。新增内容请参考我们对意见 9 的修改说明。

意见 13：关于正则化方法的部分，本文提到了 Batch Normalization（BN）在 SNN 中的应用，

但对新型正则化方法如 NeuNorm 的介绍较少。建议文中扩展这部分内容，详细探讨这些方法是如何提升 SNN 训练效果的。

对意见 13 的修改说明：感谢您关于扩展内容的宝贵意见。我们增加了 NeuNorm 的相关内容：

NeuNorm[143]专用于脉冲卷积层，作用于脉冲神经元的输出，对于每层神经元，额外记录每个位置 (i, j) 在所有通道的脉冲发放次数之和，并随时间步进行移动平均来持续更新：

$$O_{norm}[t][i][j] = k_{decay} \cdot O_{norm}[t-1][i][j] + \frac{1-k_{decay}}{C^2} \cdot \sum_{c=0}^{C-1} O[t][c][i][j], \tag{39}$$

其中 k_{decay} 是衰减因子， C 是通道数， $O[t][c][i][j]$ 是 c 通道位置为 (i, j) 处的神经元在 t 时刻的输出，而 $O_{norm}[t][i][j]$ 则是 NeuNorm 正则化项，该层传递给下一层的输出会减去该正则化项.该方法具备一定的生物可解释性，与视网膜细胞中特定位置的响应会被临近细胞进行正则化的行为类似[144, 145].NeuNorm 的效果可能来源于两方面，一是对脉冲进行了时间上的指数移动平均，使得输出更为平滑，避免了单个时刻过低或过高的发放率造成的扰动；二是使脉冲神经元层的输出变成浮点值形式的发放率，相较纯二值脉冲携带了更多信息.但 NeuNorm 无法融合进突触层，使用后会引入浮点操作，可能是这一原因导致其逐渐被 BN 类方法取代.

此外，我们在第六章“6 总结与展望”中的“(5) 正则化方法”中也增加如下表述：

此外，也可考虑针对脉冲神经元的特性，设计诸如 NeuNorm[143]类型的 SNN 专用正则化方法.

意见 14：本文讨论了 NAS 在 SNN 中的应用，但并未深入探讨其实际应用中的挑战，如计算成本和训练效率。希望文中能进一步阐明如何克服这些障碍，提高方法的实用价值。

对意见 14 的修改说明：感谢您关于补充内容的宝贵意见，我们已经对相关内容进行了扩充：

除手动设计网络结构外，也有研究者将神经网络结构搜索(Neural Architecture Search, NAS)技术引入 SNN，实现自动化的模型设计.Na 等[124]首次将 NAS 用于 SNN.该方法使用超网训练和遗传算法来优化 SNN 框架，为了解决搜索方法带来的计算成本，还使用了遗传算法中常见的 One Shot Weighting Sharing 方法[125, 126]，将权重在不同候选框架中共享；同时还提出了 Spike-Aware 优化方程，通过为发放更多脉冲的候选框架赋予更低的评分作为惩罚，从而限制脉冲数量.Kim 等[124]把搜索空间延拓到前向和反馈连接，同时提出了 Sparsity-Aware Hamming Distance (SAHD)作为指标去评估 SNN 框架，通过使用该指标避免了 NAS 方法中最耗时的超网训练过程，提升了搜索速度.Che 等[81]把搜索空间延拓到层与细胞(Cell)维度，使其能够应用于深度估计等稠密预测领域，同时把替代函数也纳入搜索空间，优化替代函数的梯度.该方法首次把可微分网络结构搜索方式引入 SNN，只训练代理参数来搜索网络结构，提升了训练速度和精度.Shen 等[127]更关注于搜索的生物原理性，提出了脑启发的神经电路演化策略.此策略搜索空间包括反馈连接、兴奋性和抑制性神经元[128]，以及基于脉冲时间依赖可塑性 (Spike-Timing-Dependent Plasticity, STDP)的局部学习方法.同时该方法进一步将任务拓展到了强化学习，取得了与 ANN 相当的性能.为了更好地在准确性与计算成本之间权衡，Yan 等[129]采用了单路径 NAS 方法，即将所有候选框架编码在一个无分支的脉冲超网中.该框架不再训练不同大小的独立卷积核，而是训练一个无分支的超网卷积核，显著降低了计算成本和搜索时间.整体来看，网络结构搜索类方法本身训练开销较大，与需要多个时间步运行的 SNN 结合后问题更为明显，已有的研究也多聚焦于此问题.

意见 15：中英文摘要内容不一致。

对意见 15 的修改说明：感谢您严谨的意见，我们原来是按照模板要求（500 英文单词，内容包含中文摘要的内容），在中文摘要的基础上增加内容得到英文摘要。但我们调查发现，也有一些已经发表的文章使用和中文摘要一致的英文摘要。现在我们已经做出修改，将英文摘要设置的与中文一致：

Spiking Neural Networks (SNNs) are regarded as the third generation of neural network models with binary communication, sparse activation, event-driven computations, and power-efficient characteristics. However, the training of SNNs is challenging because of their complex temporal dynamics and non-differentiable firing mechanisms. Recently, deep learning methods, including the surrogate gradient methods and the ANN to SNN conversion methods, have been proposed and have greatly promoted the performance of SNNs, forming the emerging spiking deep learning research area. This article focuses on the surrogate gradient methods and provides a systemic review including the following topics: (i) the basic learning methods; (ii) encoding methods; (iii) neuron and synapse model modifications; (iv) network structure designs; (v) normalization methods; (vi) ANN-auxiliary training methods; (vii) event-driven learning methods; (viii) online learning methods; (ix) training acceleration methods. Additional experiments are conducted to compare and analyze representative methods from different categories. Then, the current challenging issues and potential solutions are discussed. Finally, the possible research direction to make breakthroughs is prospected.

意见 16：文中英文缩写部分未给出对应中文和英文全称而直接使用。

对意见 16 的修改说明：感谢您的仔细检查，我们已经把文章中所有的英文缩写首次出现时的英文全称进行了标注，新增的部分可以参考正文中标红的英文缩写。

意见 17：检查修改错误表述（如：第 8 页“GLIF”应为“CLIF”等）与错别字（如：第 15 页“时间驱动”应为“事件驱动”等）。

对意见 17 的修改说明：感谢您的仔细检查，我们已经审查全文并修改了错别字。

意见 18：建议第三部分调整小结顺序，将学习训练相关内容放在一起，先讲神经元模型、再讲网络结构，最后讲学习训练相关内容

对意见 18 的修改说明：感谢您关于内容排序的宝贵意见。您提出的顺序更为合理，我们已经进行了修改，目前第三章中各小节的顺序如下：

3.1 基础学习算法
3.2 编码方式
3.3 神经元和突触改进
3.4 网络结构改进
3.5 正则化方法
3.6 ANN 辅助训练算法
3.7 事件驱动学习算法
3.8 在线学习算法
3.9 训练加速方法

意见 19：3.3 关于神经元和突触改进部分罗列内容较多，但各个模型之间的关联和比较不清晰，建议按照相关性组织，可以使用图表进行比较。

对意见 19 的修改说明：感谢您关于补充模型关系的宝贵意见，我们已经增加相关内容：

图 2 对本小节介绍的部分神经元进行了梳理，清晰地展示了神经元改进工作之间的脉络关系. 表 1 总结了部分脉冲神经元改进研究在多个数据集上的时间步数和分类正确率，以“步数|正确率”的形式展示.整体来看，随着神经动态复杂度的提升，神经元的表达能力得到提高，因而网络的任务性能也进一步提升，但这通常也会导致计算代价的提升和训练速度的降低，而神经元的并行化则可能是这一问题的解决途径.需要注意的是，AMOS 神经元类方法目前任务性能还较低，并且主要使用 MNIST 之类的简单数据集评测性能，因而没有列入到表 1 中进行对比.

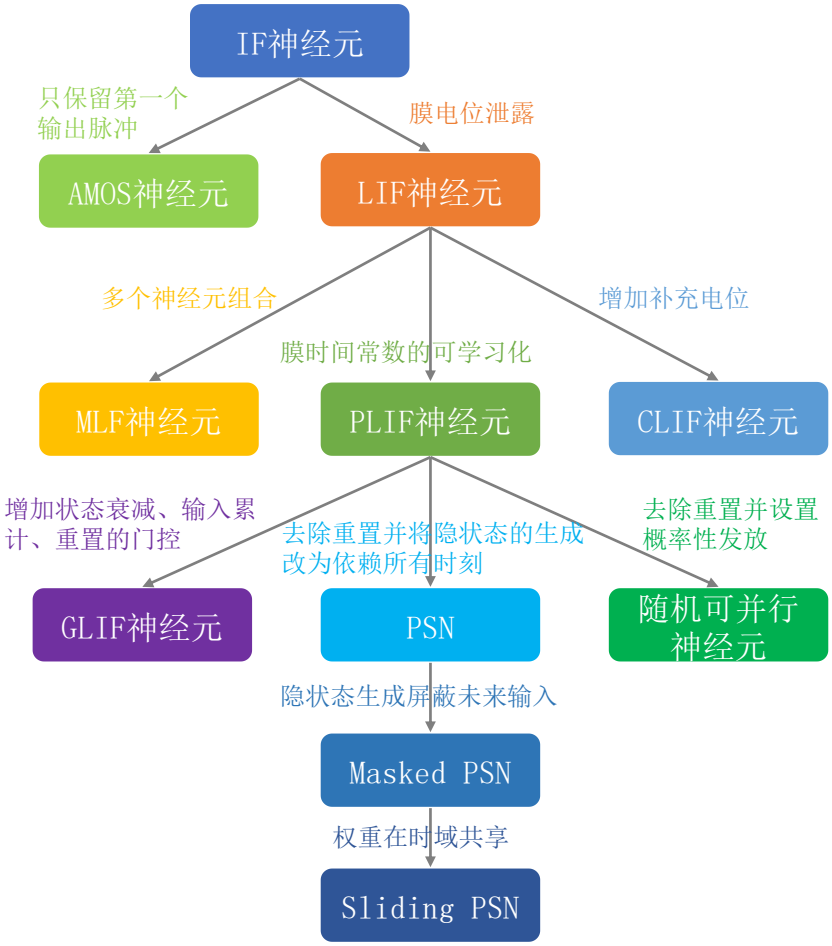

图 2 部分神经元改进工作之间的联系

意见 20：3.4 关于网络结构方面的论述建议文中也添加图表进行横向比较。

对意见 20 的修改说明：感谢您关于增加图表的宝贵建议，我们已经增加了相应内容：

图 5 展示了本小节中介绍的部分网络结构改进研究之间的递进关系.整体来看，目前新的研究集中于 Transformer 架构改进，但其中也使用了大量来自 ResNet 相关研究基础.图 6 对比了常见深度 SNN 架构在 ImageNet 数据集的分类正确率、功耗和参数量.除 MS-ResNet 外，其他网络均使用时间步数 $T=4$；默认使用 224×224 的图片分辨率进行推理，但也有部分研究者额外汇报了使用 288×288 图片分辨率推理的结果，在图中以方形点进行了标注.图 6 的结果表明，随着残差结构、自注意力机制的引入，深度 SNN 的性能得到进一步提升，在 ImageNet 数据集上已经达到 85%的正确率，同时能耗和参数量也不断优化，新的网络架构向着正确率更高且功耗和参数量更低的方向迅猛发展.

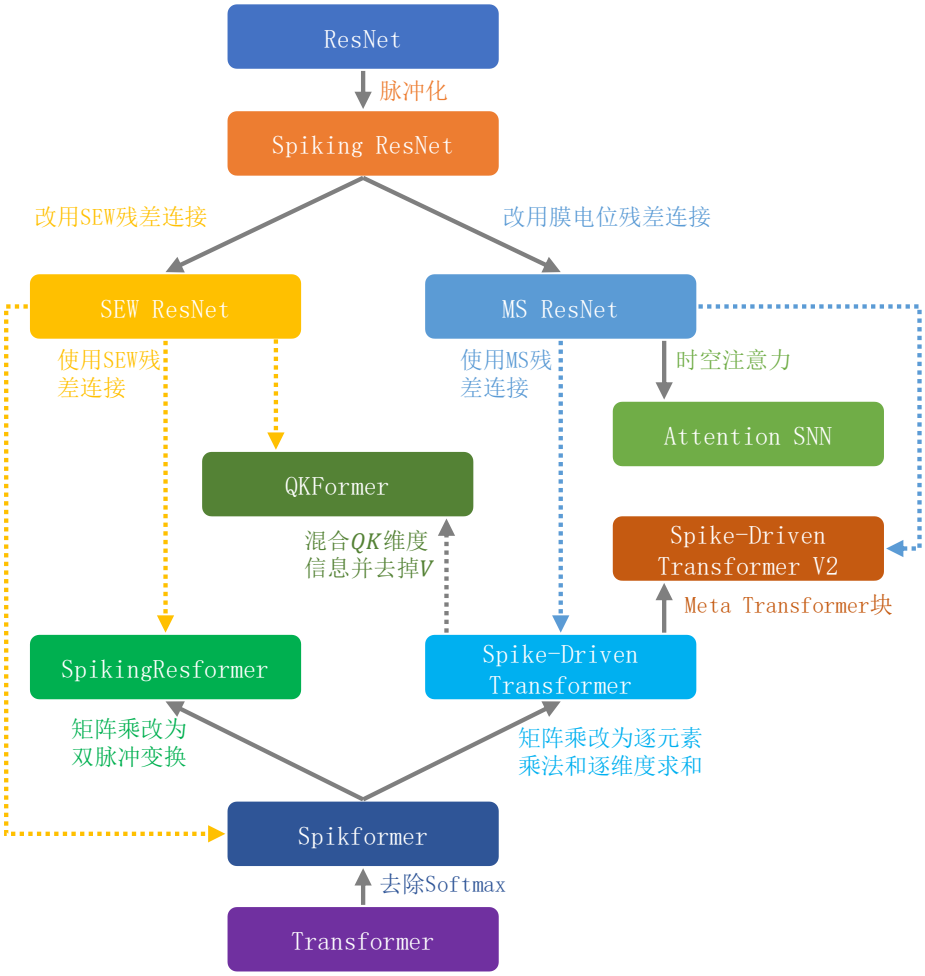

图 5 部分网络结构改进研究的联系

此外，您也可以参考我们对意见 11 的修改说明，其中补充了部分网络结构改进研究工作在目标检测任务上的性能对比表格。

意见 21：对“替代导数”的起始介绍应介绍 ANN 领域里的 Straight-Through Estimator。

对意见 21 的修改说明：感谢您关于补充背景知识的宝贵意见，我们已经增加了相关内容：

量化神经网络(Quantized Neural Network)领域的研究者，对权重或激活值进行量化时也遇到了类似的问题.量化函数是典型的输入连续、输出离散的函数，其梯度也几乎处处为 0. Bengio 等[74]提出了直通估计器(Straight-Through Estimator)以解决这一问题，其核心思路是在前向传播时使用量化函数，而在反向传播时重新定义前向传播的梯度.例如四舍五入量化的 Round 函数及其通过直通估计器定义的梯度可以表示为如下形式：

$$y = \mathrm{Round}(x), \tag{9}$$

$$\frac{\mathrm{d}y}{\mathrm{d}x} = 1. \tag{10}$$

在上述定义下，Round 函数的梯度被视作恒等函数 $y = x$ 的梯度.

在 SNN 研究领域，施路平教授团队[46]、Zenke 等[47]、Shrestha 等[48]在 2018 年分别独立地提出了梯度替代算法，其思路与直通估计器类似，成为目前直接训练深度 SNN 算法的基石. 梯度替代法在前向传播时使用 Heaviside 阶跃函数 $\Theta(x)$ 生成二值脉冲，而在反向传播时重定义 $\Theta'(x)$ 为替代函数 $\sigma(x)$ 的导数 $\sigma'(x)$.

我们还增加了从概率发放角度对梯度替代法的解释，并增加了量化神经网络中相关引用：

除了光滑近似，对于替代函数梯度的另一种解释基于概率性发放脉冲的期望的梯度[48, 76]，这一思想也被量化神经网络用于解释直通估计器[77].具体而言，将前向传播视作概率性的脉冲发放：

$$p(s=1)=\sigma(x), \tag{12}$$

$$p(s=0)=1-\sigma(x), \tag{13}$$

其中 $\sigma(x)$ 是 Sigmoid 等值域为(0,1)的函数，将输入转换为概率.相应的反向传播定义为：

$$\frac{\mathrm{d}s}{\mathrm{d}x}\approx\frac{\mathrm{d}(E(s))}{\mathrm{d}x}=\frac{\mathrm{d}(\sigma(x))}{\mathrm{d}x}=\sigma'(x), \tag{14}$$

从而得到了与将 $\sigma(x)$ 视作 $\Theta(x)$ 的光滑近似时完全相同的梯度表达式.

意见 22：本综述论文所聚焦的研究大体基于 ANN 的方法，然后融合改进 SNN 相关性能，如果说作者能够剖析或总结这一局限点，并指明 SNN 区别于（而不是“依附于”）当前 ANN 的一些发展路径或已有的一些代表性工作，将会对领域起到指导作用。

对意见 22 的修改说明：感谢您高屋建瓴的建设性意见。目前大多数研究确实是依附于 ANN 的发展路径，因而他们也占据了本文的主要篇幅。正如您所言，这种依附 ANN 的发展路径具有较大的局限性，在我们上一个版本的论文中也有相关的讨论，集中在本文第五章“5 研究挑战与未来研究方向”。在保持上述内容的基础上，我们还在文章末尾扩充了相应内容，引用了一些在该路径之外的代表性工作并进行了简要介绍：

从宏观视角来看，作为神经科学和计算科学融合产物的脉冲深度学习领域，梯度替代类学习方法目前的灵感和方法论多来自于深度学习已有的研究范式，技术路线与量化神经网络、循环神经网络、微型机器学习等领域也存在一定重合.这种研究范式是一把双刃剑，既带来了 SNN 性能的快速提升，也不可避免地引入了深度学习的固有缺陷，例如依赖大量有标注数据并进行多轮训练才能达到较好效果，而人脑则可以仅使用少量样本高效学习；在新任务上学习会引发旧任务的性能骤降，即灾难性遗忘(Catastrophic Forgetting)，而人脑则擅长利用已有经验迅速学习新任务，具有很强的迁移学习(Transfer Learning)能力；对随机扰动敏感，容易被对抗攻击(Adversarial Attack)诱导，而人类对攻击样本则展现出惊人的正确率和鲁棒性 [206].

考虑到神经科学在人工智能发展中的历史地位，以及人脑仍是已知最智能的系统这一现实，模仿大脑的结构功能和运行原理来设计 SNN 学习算法，或许能够解决传统深度学习方法面临的挑战，并推动人工智能领域取得新一次重大进展.令人欣喜的是，已经有部分研究者在这一方向进行了探索，例如通过突触可塑性[183, 207, 208]或脉冲神经元的阈值调节[209]，缓解灾难性遗忘，提升持续学习和小样本学习能力；利用循环连接[178]、超极化电流(After-hyperpolarizing Current)[179]、精细神经元模型[176]、非局部的突触可塑性[210]、兴奋型/抑制型神经元[178]、突触延迟[101]等生理结构和机制改善网络记忆能力、任务性能或参数效率；利用离散的脉冲发放过程和神经元的积分泄露机制[211]来增强对抗攻击的鲁棒性.这类研究在神经科学的指引下，充分利用了 SNN 独有的神经动态和突触可塑性机制，设计脑启发的结构和算法，并在特定任务上超越了传统深度学习方法的性能，开辟了有别于传统深度学习方法之外的研究道路，值得后来的学者参考.

意见 23：“替代导数”的核心是设计 Pseudo 导数来克服脉冲发放带来的不可导问题。领域里面不同学者对替代导数形状及参数进行了很多研究，结论有些甚至相左。也就是替代导数形状是否起决定作用。建议文中对这一重要问题进行综合实践分析，并给出指导性汇总结论，

厘清目前研究工作。

对意见 23 的修改说明：感谢您关于增加内容的宝贵意见。关于替代函数的相关问题非常重要，我们已经增加了相应的研究内容引用和论述：

替代函数是深度 SNN 训练算法的基础组件，对网络性能有着重要影响.Zenke 等[43]首先以数值模拟的方式形象展示了替代函数为何能够训练网络，他们在一个两层小网络上对比数值梯度和替代梯度，发现两者相似性较高，表明替代梯度所指示的梯度下降方向与真实梯度接近；数值梯度时而在 0 和较大值之间跳动，而替代梯度则连续且数值有限，展示出替代梯度平滑稳定的性质. Zenke 等[78]进一步研究了替代函数的形状对训练的影响.以 SuperSpike 替代函数为例，其定义为：

$$\sigma'(x)=(\beta|x|+1)^{-2}, \tag{15}$$

其中 β 为形状参数，β 越大则梯度越集中于 0 附近且数值越大，$\sigma'(x)$ 越接近 $\delta(x)$. 他们在含有一个隐藏层的 SNN 上进行学习率和替代函数形状参数的网格搜索实验，发现只要替代梯度不是常数，则训练能达到的最高正确率与形状参数无关，表明梯度替代法对替代函数的形状具有鲁棒性.但他们的这一实验也发现，不同的形状参数对应的能达到最高性能的学习率的范围也不同，如果使用不恰当的形状参数，则需要非常精细地调整学习率才能达到较好性能. Lian 等[79]的研究表明，β 直接影响替代梯度函数的宽度，太大的 β 会导致替代梯度和真实梯度之间误差较大，造成梯度不匹配；而太小的 β 则使得替代函数的宽度狭窄，造成梯度消失；两者都会导致网络难以训练. Li 等[80]则发现不同形状参数下，替代梯度和数值梯度的余弦相似度存在较大差异. 另一方面，Zenke 等[73]也测试了替代梯度的尺度对学习的影响，他们将式(15)乘上 β 得到 $\beta\sigma'(x)$ 以改变梯度的尺度，发现不同的 β 对性能影响较大，这一问题可能是较大的梯度在 SNN 使用通过时间反向传播(Back Propagation Through Time, BPTT)累乘梯度时引发梯度爆炸所致.目前研究者们通常将替代函数的梯度缩放到最大值为 1 来避免梯度爆炸问题，例如使用 SpikingJelly 框架[49]中的 $\sigma(x)=\mathrm{Sigmoid}(4x)$ 使得 $\max\sigma'(x)=\sigma'(0)=1$.

目前也有少量研究去尝试改进替代函数的选择. Li 等[80]基于数值梯度和替代梯度的余弦相似度来设置替代梯度函数的超参数，但由于数值梯度计算代价高昂，该方法只被用于网络首层脉冲神经元. Lian 等[79]根据膜电位的分布动态调整替代梯度的宽度，避免梯度不匹配或梯度消失问题.Che 等[81]对不同替代函数参数在训练时使用 Softmax 混合，而推理时使用 Argmax 选择，实现可微分参数搜索.

此外，您也可以参考我们对意见 21 的修改说明，其中还补充了一些关于量化神经网络的背景知识和基于概率角度对替代梯度的解释。

意见 24：图 1：需要提升清晰度，以更好地展示相关信息。

对意见 24 的修改说明：感谢您的宝贵意见，我们上一个版本的文章中图片清晰度不佳，深表歉意。我们已经将除图 7（该画图软件不支持导出 word 兼容的矢量图格式）外的所有图片都换成了矢量图，可以无损放大。

意见 25：第 3 章引言部分：关于 MNIST 分类任务的描述需要重新表述，以避免将其定义为“玩具级别”任务的不当表述。

对意见 25 的修改说明：感谢您的严谨意见，我们之前的表述不当，现在已经修改为：

由于高性能学习算法的缺失，SNN 一度只能解决 MNIST 分类这种简单任务。

意见 26：第 3.4 节：建议文中增加图卷积脉冲神经网络的相关讨论，以完善内容。

对意见 26 的修改说明：感谢您关于补充内容的宝贵意见。目前 SNN 领域蓬勃发展，研究范围不断扩展，我们也增加了相应的图卷积脉冲神经网络、点云脉冲网络等内容：

近年来还有一些研究将 SNN 处理的数据类型从图片和神经形态数据集扩展到图(Graph)、点云(Point Clouds)等，图卷积神经网络(Graph Convolutional Network, GCN)、PointNet 等架构的脉冲版本被相继提出.Gu 等[127]针对触觉数据构建图结构，并使用图卷积预处理，特征由 LIF 神经元转换为脉冲，继而输出到使用 LIF 神经元的 MLP，是较早的成功将 SNN 结合 GCN 的研究.Zhu 等[128]提出的脉冲卷积图神经网络，则是使用频率编码来将图卷积得到的特征转换为脉冲，且 MLP 中使用三值激活的 LIF 神经元，在 4 个引文网络公共数据集上取得了比部分经典 GCN 模型更好的性能.Li 等[141]使用了自适应阈值的 LIF 神经元，并将其用于聚合邻域信息；LIF 神经元动态的放电机制使得不同时刻的图结构也呈现动态变化，网络因此能够捕捉动态图的时序信息. Guo 等[142]将 PointNet 中的激活函数换成 LIF 神经元以构建脉冲 PointNet，并通过单步训练和多步推理以降低训练开销，同时通过训练时随机初始化膜电位的方式来降低其和多步推理时的性能差异.总体而言，将 SNN 推广至其他数据类型的网络结构，通常能带来理论上的激活值存储开销和推理能耗的降低，且脉冲神经元的特性使得网络具备一定时序信息提取能力，而主要挑战则包括梯度跨越多个时间步的衰减、BPTT 带来的较大的训练开销，以及诸如图卷积、点云采样等操作难以在现有神经形态计算芯片上实现等问题.

此外，您也可以参见我们对意见 11 的修改说明，其中还增加了在目标检测任务上的网络结构改进相关工作。

意见 27：第 3.7 节：文中应补充重要在线学习算法的介绍及分析。

对意见 27 的修改说明：感谢您关于增加内容的宝贵意见，我们上一个版本的内容中分析较为简略，现在已经重新修改，并将各个算法的关键之处进行了说明：

其后的在线学习算法则是对 BPTT 的完整梯度进行分析，对其在每个时刻的分量进行单步近似，相较于仅使用局部信息的 DECOLLE 性能更好.

由于梯度的计算通常不是逐元素的，涉及张量之间的求导，在本小节中我们使用粗体字母表示张量(Tensor)，而字母含义则与前文一致.不失一般性，对于使用 LIF 神经元的多层 SNN，基于 BPTT 的第 l 层的权重 W^l 的梯度为：

$$\begin{aligned}\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}\mathbf{W}^l}&=\sum_{t=0}^{T-1}\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}\mathbf{V}^{l+1}[t]}\frac{\partial\mathbf{V}^{l+1}[t]}{\partial\mathbf{S}^l[t]}\frac{\partial\mathbf{S}^l[t]}{\partial\mathbf{V}^l[t]}\frac{\mathrm{d}\mathbf{V}^l[t]}{\mathrm{d}\mathbf{W}^l}\\&=\sum_{t=0}^{T-1}\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}\mathbf{V}^{l+1}[t]}\frac{\partial\mathbf{V}^{l+1}[t]}{\partial\mathbf{S}^l[t]}\frac{\partial\mathbf{S}^l[t]}{\partial\mathbf{V}^l[t]}\cdot\\&\sum_{j=0}^{t}\left[\prod_{i\in[j+1,t],i\in\mathbb{N}}\left(\frac{\partial\mathbf{V}^l[i]}{\partial\mathbf{V}^l[i-1]}+\frac{\partial\mathbf{V}^l[i]}{\partial\mathbf{S}^l[i-1]}\frac{\partial\mathbf{S}^l[i-1]}{\partial\mathbf{V}^l[i-1]}\right)\right]\frac{\partial\mathbf{V}^l[j]}{\partial\mathbf{W}^l}.\end{aligned}\tag{54}$$

在线学习方法通常希望在网络运行到当前时间步 t 时，就能近似计算出式(54)的最左侧求和符号内的第 t 项. Online Training Through Time (OTTT)[160]的主要改动为：忽略重置过程的梯度，使 $\frac{\partial\mathbf{V}^l[i]}{\partial\mathbf{S}^l[i-1]}=0$；使用软重置，使得 $\frac{\partial\mathbf{V}^l[i]}{\partial\mathbf{V}^l[i-1]}=\lambda$ 为常数.在上述设置下，通过展开来自后续层的梯度，式(54)可以简化为：

$$\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}\mathbf{W}^l}=\sum_{t=0}^{T-1}\left[\left(\frac{\partial\mathcal{L}}{\partial\mathbf{S}^{l_{max}}[t]}\frac{\partial\mathbf{S}^{l_{max}}[t]}{\partial\mathbf{V}^{l_{max}}[t]}\prod_{j=l+1}^{l_{max}-1}\frac{\partial\mathbf{V}^{j+1}[t]}{\partial\mathbf{S}^{j}[t]}\frac{\partial\mathbf{S}^{j}[t]}{\partial\mathbf{V}^{j}[t]}\right)\frac{\partial\mathbf{V}^{l+1}[t]}{\partial\mathbf{S}^{l}[t]}\frac{\partial\mathbf{S}^{l}[t]}{\partial\mathbf{V}^{l}[t]}\right]^{\mathrm{T}}\cdot\left(\sum_{\tau\le t,\tau\in\mathbb{N}}\lambda^{t-\tau}\mathbf{S}^{l-1}[\tau]\right)^{\mathrm{T}}. \tag{55}$$

定义资格迹(Eligibility Traces)变量：

$$\mathbf{E}^{l-1}[t]=\lambda\mathbf{E}^{l-1}[t-1]+\mathbf{S}^{l-1}[t], \tag{56}$$

其初始化为$\mathbf{E}^{l-1}[0]=\mathbf{S}^{l-1}[0]$.式(55)中的$\sum_{\tau\le t,\tau\in\mathbb{N}}\lambda^{t-\tau}\mathbf{S}^{l-1}[\tau]$即可转换为$\mathbf{E}^{l-1}[t]^{\mathrm{T}}$，此时该式的第$t$个求和项只需要$t$时刻的信息.该工作还从理论上论证了其梯度与基于脉冲表征的Differentiation on Spike Representation方法[161]之间的正相关性.

Spatial Learning Through Time (SLTT)[162]延续了OTTT忽略重置梯度和使用软重置的设置，并完全忽略膜电位在时刻之间的梯度，即令$\frac{\partial\mathbf{V}^l[i]}{\partial\mathbf{V}^l[i-1]}=0$.在上述设置下，式(54)简化为：

$$\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}\mathbf{W}^l}=\sum_{t=0}^{T-1}\left[\left(\frac{\partial\mathcal{L}}{\partial\mathbf{S}^{l_{max}}[t]}\frac{\partial\mathbf{S}^{l_{max}}[t]}{\partial\mathbf{V}^{l_{max}}[t]}\prod_{j=l+1}^{l_{max}-1}\frac{\partial\mathbf{V}^{j+1}[t]}{\partial\mathbf{S}^{j}[t]}\frac{\partial\mathbf{S}^{j}[t]}{\partial\mathbf{V}^{j}[t]}\right)\frac{\partial\mathbf{V}^{l+1}[t]}{\partial\mathbf{S}^{l}[t]}\frac{\partial\mathbf{S}^{l}[t]}{\partial\mathbf{V}^{l}[t]}\right]^{\mathrm{T}}\mathbf{S}^{l-1}[t]^{\mathrm{T}}. \tag{57}$$

因而SLTT只需要保存当前时间步的脉冲$\mathbf{S}^{l-1}[t]$即可实现在线学习.

在OTTT的基础上，Neuronal Dynamics-based Online Training (NDOT)[163]对层内的时间依赖性进行了更细致的建模，不像式子(56)中简单的使用膜电位的衰减λ，而是将重置过程也纳入：

$$\mathbf{E}^{l-1}[t]=\mathbf{E}^{l-1}[t-1]\cdot\frac{U^l[t]-V_{th}\mathbf{S}^l[t]}{U^l[t-1]-V_{th}\mathbf{S}^l[t-1]}+\mathbf{S}^{l-1}[t]. \tag{58}$$

意见28：表4：需要明确LIF神经元的耗时单位是否为毫秒，并说明实验中所使用的输入类型。此外，建议文中对其他脉冲神经元的耗时情况进行补充，以便进行对比分析。

对意见28的修改说明：感谢您的仔细检查，我们已经在该表（现在为表5）中明确写出了LIF神经元的耗时单位为毫秒。此外，原来的BlockALIF神经元的数据录入（从原始数据到正文表格）有一些错误，我们已经进行了修复，该错误不影响实验结论，即BlockALIF神经元的加速效果仍然较差。我们在正文中增加了对输入类型的说明：

实验环境为Intel Core i9-10900X CPU，64G内存，Nvidia RTX 2080 Ti GPU；神经元数量为4096；分别测试不同神经元在时间步数$T=2,4,8,16,32,64$时进行训练（前向传播、反向传播和梯度下降）的耗时；输入是从取值范围(0,1)的均匀分布采样的随机张量.

修改后的表5如下所示：

表 5　对比加速方法性能

T	相较于 LIF 神经元的加速比						LIF 耗时(ms)
	SpikingJelly	PSN	BlockALIF 分组大小				
			2	4	8	16	
2	1.03	2.20	0.20				1.44
4	1.48	4.07	0.17	0.38	0.38		3.02
8	2.72	6.81	0.15	0.29	0.29		4.79
16	6.19	12.60	0.22	0.29	0.29	1.29	9.48
32	16.61	17.76	0.25	0.40	0.40	1.01	17.14
64	14.83	43.75	0.24	0.45	0.45	0.98	30.60

感谢您关于补充耗时情况的意见，我们在附录中增加了原始的耗时信息：

表 6　不同加速方法的耗时(ms)

T	SpikingJelly	PSN	BlockALIF 分组大小				LIF
			2	4	8	16	
2	1.40	0.65	7.08				1.44
4	2.03	0.74	17.38	7.87			3.02
8	1.76	0.70	32.18	16.34	8.04		4.79
16	1.53	0.75	42.90	32.48	17.06	7.37	9.48
32	1.03	0.97	67.67	42.99	28.90	16.90	17.14
64	2.06	0.70	125.98	68.75	42.40	31.16	30.60

此外，我们还在 CIFAR10 对比实验中增加了每种方法的训练和推理速度对比，展现在正文的图 8 中。您也可以参考我们对意见 9 的修改说明，其中增加了不同方法的比较指标。

意见 29：第 4 章综合对比实验中使用的数据集时域信息不够丰富，建议文中增加语音数据集或者其他时域信息丰富的数据；

对意见 29 的修改说明：感谢您关于补充数据类型的宝贵建议。我们已经增加了神经形态的语音 SHD 语音数据集分类任务、神经形态的 Gen1 数据集目标检测任务。关于这些内容的详细信息可以参考我们对意见 9 的修改说明。

意见 30：第 20 页正则化方法段落存在误打情况，“这一问题‘扔’有待 SNN 的研究者解决”。

对意见 30 的修改说明：感谢您的仔细检查。我们已经检查并修改了全文的错别字。