Event2Vec: Processing Neuromorphic Events Directly by Representations in Vector Space
Event2Vec：在向量空间中直接表征并处理神经形态事件

方维 [1] **Priyadarshini Panda** [2]

摘要

与传统相机相比，神经形态事件相机具有更高的时间分辨率、更好的能效以及更大的动态范围。然而，其异步且稀疏的数据格式给常规深度学习方法带来了显著挑战。现有方法大多将事件稠密化为帧，从而牺牲其稀疏异步特性；或者采用与 GPU 加速兼容性较差的不规则模型。受 word-to-vector 模型启发，我们提出 event2vec，这是一种能够使 Transformer 直接处理事件的新型表示。我们在 DVS Gesture、ASL-DVS 和 DVS-Lip 基准上验证了 event2vec 的有效性，结果表明 event2vec 具有很高的参数效率、高吞吐量和低延迟，并且即使在事件数量极少或空间分辨率很低的情况下也能取得较高精度。这些结果说明，稀疏异步事件数据可以直接融入高吞吐量 Transformer 架构，为实时神经形态视觉提供一种高效范式。代码见 https://github.com/Intelligent-Computing-Lab-Panda/event2vec。

[1]电子与计算机工程系，耶鲁大学 [2]电子与计算机工程系，南加州大学. Correspondence to: Priyadarshini Panda <priya.panda@usc.edu>.

1. 引言

神经形态计算是一个新兴研究方向，旨在通过模拟大脑工作原理来发展下一代人工智能 (Mead, 1990)。这一范式的重要成果之一是事件相机，这类传感器受到生物视网膜启发 (Gallego et al., 2022)。典型代表包括动态视觉传感器（Dynamic Vision Sensor, DVS）(Lichtsteiner et al., 2008) 和异步时间型图像传感器（Asynchronous Time-based Image Sensor, ATIS）(Posch et al., 2011)。不同于传统相机同步采集图像帧，事件相机以异步方式工作，并在每个像素亮度发生变化时生成事件。这一工作机制赋予其极高的时间分辨率（微秒量级）、低功耗以及超过 120 dB 的高动态范围（High Dynamic Range, HDR）。这种异步工作方式产生的是稀疏事件流，通常以地址事件表示（Address-Event Representation, AER）格式编码。一个事件表示为四元组 (x, y, t, p)，其中包含像素的空间坐标 (x, y)、时间戳 t 以及表示亮度变化方向的二值极性 p。

当前大多数深度学习模型都被设计为处理稠密、规则结构的多维张量。这种规则范式是主流深度学习的基础 (LeCun et al., 2015)，并广泛用于现代科学计算和机器学习框架中，例如 NumPy (Harris et al., 2020)、TensorFlow (Abadi et al., 2016) 和 PyTorch (Paszke et al., 2019)。因此，AER 格式事件流的稀疏性和异步性与这些规则方法存在根本不兼容。为弥合这一差距，大量研究致力于将事件转换为稠密表示，或设计新的数据结构和网络结构以直接处理不规则事件。

现有方法主要关注事件编码这一挑战：如何从事

Word	how	are	you
Index	5269	527	499
Position	0	1	2

Event	$E[0]$	$E[1]$	$E[2]$
Index	$x[0], y[0], p[0]$	$x[1], y[1], p[1]$	$x[2], y[2], p[2]$
Position	$t[0]$	$t[1]$	$t[2]$

Figure 1. 词与事件之间的概念类比。

件中有效提取信息，并将其表示为可供神经网络处理的形式。该挑战类似于自然语言处理中的词编码问题，而 word-to-vector（word2vec）(Mikolov et al., 2013) 已成功解决了这一问题。word2vec 模型将每个词嵌入为固定长度向量，使词与词之间的关系能够通过向量之间的数学运算来表示。这种向量表示方式与深度学习架构高度兼容，并已成为现代自然语言处理（Natural Language Processing, NLP）模型的基础组成部分 (Devlin et al., 2019; Brown et al., 2020)。如图 1 所示，我们发现词与事件之间存在许多相似之处，主要包括：

(1) **每个元素都由索引和位置共同构成。** 在 NLP 中，每个词都会由 tokenizer 转换为词表中的唯一索引；例如，图 1 中的索引由 Llama-3 tokenizer 生成 (Grattafiori et al., 2024)。词的位置是其在句子中的序列位置（例如，在 “how are you” 中，词 “how” 的位置为 0）。类似地，事件的索引是其空间地址，由三元组 (x, y, p) 表示。关键在于，事件的位置不是序列编号，而是其时间戳 t，该时间戳标记了事件流中的精确时间位置。

(2) **可能索引的集合是有限的。** 语言的词汇表构成 NLP 中使用的字典，其规模是有限的。同样，事件相机也只有有限数量的可能事件索引，该数量由传感器属性决定。例如，DVS128 相机具有 $2 \times 128 \times 128$ 个唯一索引，对应 128×128 空间分辨率下的两种极性。

(3) **序列表现出天然顺序。** 句子中的词按照特定顺序排列，而这种顺序决定语义。类似地，事件天然按照时间戳排序，反映被捕捉变化的时间演化过程。这种内在时间顺序是事件数据区别于点云等无序数据结构的关键特征。

(4) **元素的含义由上下文决定。** 一个词可能具有多重含义；例如，“transformer” 既可以指神经网络架构，也可以指动画角色，其具体含义需要由周围文本消歧。单个事件仅表示某个像素在某一时刻发生亮度变化，孤立来看信息量很少。然而，当其置于时空事件流中观察时，一系列事件可以勾勒出物体轮廓，从而赋予单个事件更高层次的含义，例如它可能是边缘的一部分。因此，事件的意义同样从根本上依赖上下文。

受 word2vec 启发，我们提出 event-to-vector（event2vec），这是一种面向异步事件的高效时空表示。本文贡献如下：

(1) 通过将事件嵌入到向量空间中，本文方法能够原生处理输入流的稀疏特性，避免事件帧等稠密中间表示，从而可利用现代网络架构实现高效的 GPU 加速处理。

(2) 我们提出参数化空间嵌入和基于卷积的时间嵌入方法，以捕捉邻域相似性。这一特性对精度至关重要，但标准嵌入层（查找表）难以学习。

(3) 我们在三个广泛使用的分类基准 DVS Gesture (Amir et al., 2017)、ASL-DVS (Bi et al., 2019) 和 DVS-Lip (Tan et al., 2022) 上验证了本文方法。该方法在取得有竞争力精度的同时，还表现出优异的参数效率、吞吐量、延迟表现，以及对低事件数量和低空间分辨率的鲁棒性。

2. 相关工作

2.1. 事件的稠密表示与处理

由原始事件流得到的稠密表示与常规深度学习方法完全兼容。这通常通过沿时间轴积分事件，将其形成稠密的 3D 或 4D 张量来实现，例如事件帧 (Liu & Delbruck, 2018)、多通道图像 (Barchid et al., 2022)、体素网格 (Bardow et al., 2016)、体积立方体 (Cordone et al., 2022) 和 patch (Sabater et al., 2023; Peng et al., 2023)。具体而言，event-to-frame 方法在离散时间间隔内累积事件，得到的帧随后可由标准神经网络直接处理。然而，这类方法的一个显著缺点是会降低

事件数据固有的高时间分辨率，因为在转换过程中单个事件的时间戳会被聚合或量化。此外，将数据转换为稠密表示也会抵消事件固有的空间稀疏性。例如，生成的帧通常包含大量零值像素；这些像素不携带信息，却仍会带来显著的内存和计算开销。许多方法隐式地使用时间戳来定义积分区间，也有一些方法显式利用时间戳生成时间权重 (Zhu et al., 2019; Gehrig et al., 2019)。最后，转换过程本身可能计算开销较大，引入的延迟通常会妨碍实时应用 (Rebecq et al., 2019; Gallego et al., 2022)。

2.2. 事件的不规则表示与处理

相反，处理不规则表示的方法旨在保留事件数据固有的稀疏性和异步性。这类方法包括脉冲神经网络（Spiking Neural Networks, SNNs）(Maass, 1997; Roy et al., 2019)、稀疏卷积网络（Sparse CNNs）(Messikommer et al., 2020; Santambrogio et al., 2024)、图神经网络（Graph Neural Networks, GNNs）(Bi et al., 2019; Schaefer et al., 2022) 以及点式方法 (Yang et al., 2019; Sekikawa et al., 2019; Lin et al., 2023; Ren et al., 2025)。

当部署在神经形态硬件上时 (Merolla et al., 2014; Davies et al., 2018)，SNNs 可以以天然异步、事件驱动的方式处理事件。然而，在标准硬件上，基于 GPU 的 SNN 仿真会产生稠密张量输出，因为硬件需要以离散时间步进行同步处理。因此，在 GPU 上训练 SNNs 通常以同步方式进行，这导致同步训练与异步推理之间不可避免地存在性能差距 (Yao et al., 2024; Du et al., 2025)。此外，对时间反向传播的依赖也使训练过程缓慢且内存开销较大。 Sparse CNNs 利用事件数据的固有稀疏性，理论上可获得较低的浮点运算次数（FLOPs）。然而，GPU 架构并未针对高效稀疏加速所需的动态计算和非结构化内存访问进行优化。因此，与 SNNs 类似，Sparse CNNs 无法充分利用 GPU 的大规模并行处理能力。

事件型 GNNs 从输入事件构建图，这种方式能够有效保留事件之间的时空关系。由于没有事件活动的空区域不会生成图节点，数据稀疏性得到了较好利用。其主要缺点在于需要仔细调整图构建中的超参数，例如事件下采样率和邻域半径。此外，GNNs 具有低通滤波器特性 (Nt & Maehara, 2019)，容易受到过平滑问题影响 (Zhou et al., 2020)，这限制了它们构建与现代 CNN 和 Transformer 相当的深层架构的能力 (Vaswani et al., 2017)。点式方法将事件相机产生的事件视为类似激光雷达（LiDAR）点云的数据。多数点云模型的一个根本限制是其置换不变性，这要求将输入视为无序集合。因此，事件时间戳通常退化为一个额外的位置坐标，从而丢弃了事件关键的因果顺序。为控制数据量，这些方法还常采用最远点采样等经典点云预处理技术，进一步增加了延迟。

3. 方法

3.1. 在向量空间中表示事件

基于词与事件之间的强类比，我们提出一种在向量空间中表示事件的方法，并将其称为 event-to-vector（event2vec）。对于空间分辨率为 $H \times W$ 的相机，一个事件表示为四元组 (x, y, t, p)。在本文嵌入方法中，我们将三元组 (x, y, p) 视为空间-极性坐标，将时间戳 t 视为时间坐标。event2vec 嵌入的一般形式定义为：

$$\mathbf{v} = \mathbf{v}_s + \mathbf{v}_t = \text{Embed}_s(x, y, p) + \text{Embed}_t(\Delta t), \quad (1)$$

其中 $\mathbf{v} \in \mathbb{R}^D$ 是最终得到的 D 维嵌入向量，$\mathbf{v}_s = \text{Embed}_s(x, y, p) \in \mathbb{R}^D$ 是空间-极性嵌入向量，$\mathbf{v}_t = \text{Embed}_t(\Delta t) \in \mathbb{R}^D$ 是时间嵌入向量。如式 1 所示，该方法通过加法融合空间信息与时间信息。这种加性融合策略直接受到 Transformer 中常用位置编码机制的启发。

3.2. 空间嵌入

空间嵌入模块的一种直接做法是借鉴 NLP 中的标准嵌入层，该层可高效实现为查找表：

$$\mathbf{v}_s = \text{Embed}_s(x, y, p) = \mathbf{W}_s[p \cdot H \cdot W + y \cdot W + x] \quad (2)$$

其中 $\mathbf{W}_s \in \mathbb{R}^{(P \cdot H \cdot W) \times D}$ 是可学习嵌入矩阵，D 是嵌入维度。该方法将每个唯一的空间-极性坐标映射到

嵌入矩阵 $\mathbf{W}_s$ 中的一个不同行索引。这种基于查找表的空间嵌入与 EventNet (Sekikawa et al., 2019) 空间映射的查找表实现类似，因为二者都作用于有限的 $2HW$ 空间-极性地址空间；我们将在 Sec. 3.5 中结合时间部分进一步讨论这一联系。

然而，这种标准嵌入层不会对索引之间的关系施加任何归纳偏置，迫使模型仅从数据中学习所有空间关系。在 tokenizer 中，词索引是非语义标识符，其分配主要由训练语料中的词频决定。因此，索引为 i 和 $i+1$ 的词并不天然具有语义相似性。这一假设并不适用于事件坐标。图像是连续二维函数 (Gonzalez, 2009)，空间相邻像素通常具有很强相关性。因此，有效的空间嵌入应当引入这种局部性偏置，使坐标接近的事件产生相似的嵌入向量：

$$\text{Embed}_s(x+\Delta x, y+\Delta y, p) - \text{Embed}_s(x,y,p) \approx \mathbf{0} \quad (3)$$

其中 $(\Delta x, \Delta y)$ 表示较小的坐标扰动。

式 2 中的标准嵌入无法刻画这一关键空间关系，从而可能阻碍学习过程。为解决该问题，我们提出一种优雅的参数化嵌入：使用神经网络 ϕ 直接将每个归一化坐标三元组 (x, y, p) 映射为其嵌入。从概念上看，该参数化嵌入也可理解为通过在所有可能空间-极性坐标上计算 ϕ 来生成嵌入矩阵 $\mathbf{W}_\phi$。为系统枚举 $P \times H \times W$ 体素中的所有空间-极性坐标（其中 $P = 2$ 表示两种极性），我们首先构造线性索引序列 $\mathbf{c} = [0, 1, \ldots, P \cdot H \cdot W - 1]$。随后将该序列分解为三个 probe 张量 $\mathbf{x}_c$、$\mathbf{y}_c$ 和 $\mathbf{p}_c$，分别对应宽度、高度和极性维度上的坐标。该变换定义如下：$\mathbf{x}_c = \mathbf{c} \pmod W, \mathbf{y}_c = \left\lfloor \frac{\mathbf{c}}{W} \right\rfloor \pmod H, \mathbf{p}_c = \left\lfloor \frac{\mathbf{c}}{WH} \right\rfloor$. 在输入 ϕ 之前，这些坐标张量会按照实现被归一化到 $[-1, 1]$：

$$\bar{\mathbf{x}}_c = \tfrac{2\mathbf{x}_c}{W-1} - 1, \quad \bar{\mathbf{y}}_c = \tfrac{2\mathbf{y}_c}{H-1} - 1, \quad \bar{\mathbf{p}}_c = \tfrac{2\mathbf{p}_c}{P-1} - 1. \quad (4)$$

最后，将这些归一化后的 probe 张量输入 ϕ，即可得到概念上的嵌入矩阵 $\mathbf{W}_\phi = \phi(\bar{\mathbf{x}}_c, \bar{\mathbf{y}}_c, \bar{\mathbf{p}}_c)$。这种参数化生成的矩阵等价于对任意原始事件坐标 (x, y, p) 先执行同样的归一化得到 $(\bar{x}, \bar{y}, \bar{p})$，再直接计算 ϕ：

$$\mathbf{W}_\phi[p \cdot H \cdot W + y \cdot W + x] = \phi(\bar{x}, \bar{y}, \bar{p}). \quad (5)$$

关键在于，参数化网络 ϕ 被设计为连续且可微的函数。该性质使我们能够通过一阶泰勒展开形式化分析相邻嵌入之间的关系：

$$\begin{aligned}&\phi(x+\Delta x, y+\Delta y, p) - \phi(x,y,p) \\ &= J_\phi^{x,y}(x,y,p) \begin{bmatrix}\Delta x \\ \Delta y\end{bmatrix} + o\left(\|(\Delta x, \Delta y)\|\right), \quad (6)\end{aligned}$$

其中 $J_\phi^{x,y}(x,y,p)$ 表示 ϕ 关于空间坐标 (x, y) 的雅可比矩阵。如式 6 所示，对于较小扰动 $(\Delta x, \Delta y)$，嵌入之间的差异可由该雅可比矩阵与扰动向量的矩阵-向量乘积近似。因此，当扰动趋近于零时，该差异向量也趋近于零。通过这种方式，连续参数化网络 ϕ 能够将所需的邻域语义，即空间归纳偏置，直接嵌入到嵌入矩阵中。该方法自然满足式 3 中给出的条件。

3.3. 时间嵌入

时间戳表示事件发生时间，其作用类似于句子中的位置索引。在现代 NLP 模型中，相对位置编码方法 (Press et al., 2021; Su et al., 2024) 越来越多地取代绝对位置编码方法，例如正弦编码 (Vaswani et al., 2017) 或可学习绝对位置嵌入 (Devlin et al., 2019)。

然而，将这些相对位置编码技术直接应用于事件时间戳并不合适。这些方法本质上面向离散且均匀间隔的索引，而事件时间戳是连续且天然非均匀的。为解决这一差异，我们提出直接从相邻时间戳之间的差分中学习时间嵌入。

具体而言，时间嵌入模块由一组卷积层堆叠而成，其输入为归一化时间戳的一阶时间差分序列。对每个事件流，令 $\tilde{\mathbf{t}} = \mathbf{t}/\max(\mathbf{t})$。我们使用 $\Delta\mathbf{t} = [0, \tilde{\mathbf{t}}[1] - \tilde{\mathbf{t}}[0], \tilde{\mathbf{t}}[2] - \tilde{\mathbf{t}}[1], \ldots, \tilde{\mathbf{t}}[L-1] - \tilde{\mathbf{t}}[L-2]]$，其中 L 为事件数量，初始的 0 用于使时间间隔序列与事件序列对齐。该设计具有以下优点：

(1) **时间平移不变性：** 通过作用于相对时间差分，嵌入天然对绝对时间平移保持不变。

(2) **上下文一致性：** 卷积操作使单个事件的时间嵌入能够受到其相邻事件时间的影响，从而在时

间域中强化邻域语义原则。另一方面，单个事件的发生可能包含一定噪声，而卷积可以起到局部平滑和降噪作用。

(3) **优化效率与归纳偏置：** 将 $\Delta\mathbf{t}$ 作为输入相当于一种时间上的“预条件化”，与残差学习思想一致 (He et al., 2016)。虽然网络理论上可以从绝对时间戳 $\mathbf{t}$ 推断时间间隔，但显式建模 $\Delta\mathbf{t}$ 通过直接表示事件速度降低了优化负担。由于卷积操作本质上执行加权求和，它在数学上与差分输入 $\Delta\mathbf{t}$ 相契合。连续时间差分的求和具有明确物理意义：它表示局部事件窗口的累积持续时间，使网络能够直接度量局部事件密度。

3.4. 事件采样与聚合

原始事件流通常包含数量极大的事件，且序列长度差异显著。此外，深度学习框架通常按 batch 处理数据，这要求同一 batch 内所有张量具有统一维度。因此，需要从每个事件流中采样或聚合事件，得到长度为 L 的固定长度序列。

本文主要使用两种方法。第一种是**均匀随机采样**。我们发现这种简单方法在大多数情况下效果良好，且计算效率极高。然而，随机采样的一个显著限制是会丢弃大部分事件，造成大量信息损失，从而在复杂任务中导致精度不理想。第二种方法通过 **K-means 聚类**将整个事件流聚合为 L 个代表性簇来缓解该问题。具体而言，聚类过程在两种事件极性上独立执行，以保留其各自的信息通道。此外，我们计算强度因子 ρ，其等于属于该簇的原始事件数量。该强度因子随后调制对应的事件 token，从而根据事件密度对表示进行有效加权。

为降低推理阶段运行 K-means 聚类算法的延迟，我们提出基于 GPU 的**批量 K-means++ 算法**。该方法通过多步批量计算近似 K-Means++ 初始化 (Arthur & Vassilvitskii, 2007) 的逐步迭代过程。同时，它仅使用新采样的一批中心增量更新最近中心距离，并基于 PyTorch 实现了高效的 GPU 版本。详细内容见附录 A.1。

3.5. Event2Vec 的形式化表达

综上，对于包含 L 个事件的序列，最终 event2vec 表示为张量 $\mathbf{V} \in \mathbb{R}^{L\times D}$。该序列中第 i 个事件的嵌入 $\mathbf{V}[i]$ 表达为：

$$\mathbf{V}[i] = (\log(\boldsymbol{\rho}[i]) + 1) \cdot \Big(\mathrm{Embed}_s(\mathbf{x}[i], \mathbf{y}[i], \mathbf{p}[i]) + \mathrm{Embed}_t(\Delta\mathbf{t})[i]\Big), \tag{7}$$

其中 $\boldsymbol{\rho}$、$\mathbf{x}$、$\mathbf{y}$、$\mathbf{p}$ 和 $\mathbf{t}$ 分别表示 L 个事件的强度因子、空间坐标、极性和时间戳序列。这里 $\Delta\mathbf{t}[0] = 0$，且对 $i \in \{1, \ldots, L-1\}$ 有 $\Delta\mathbf{t}[i] = \tilde{\mathbf{t}}[i] - \tilde{\mathbf{t}}[i-1]$，其中 $\tilde{\mathbf{t}} = \mathbf{t}/\max(\mathbf{t})$。对于原生事件，$\boldsymbol{\rho}[i]$ 为 1；对于簇事件，它表示聚合到该簇中的原始事件数量。我们对 $\boldsymbol{\rho}$ 取对数，以抑制包含过多事件的簇，防止其主导整个序列。

为更好理解该形式化表达，将 event2vec 与早期逐事件表示进行比较是有帮助的。值得注意的是，EventNet (Sekikawa et al., 2019) 也将事件分解为离散空间-极性地址和相对时间信息。具体而言，EventNet 将事件地址 $\mathbf{e}[i] = (\mathbf{x}[i], \mathbf{y}[i], \mathbf{p}[i])$ 映射为特征向量 $\mathbf{z}[i] = h(\mathbf{e}[i])$，随后使用基于 $\Delta t_{j,i} = \mathbf{t}[j] - \mathbf{t}[i]$ 的时间编码函数。由于 $\mathbf{e}[i]$ 仅有 $2HW$ 个可能值，EventNet 会预计算 $h(\mathbf{e}[i])$，并在推理时以查找表（LUT）实现。从这个意义上说，式 2 中的标准空间嵌入也在有限事件地址空间上遵循类似查找原则。

然而，两种方法对时间的概念化和使用方式存在根本差异。在 EventNet 中，$\Delta t_{j,i}$ 表示滑动窗口内某个过去事件与最新事件之间经过的时间。该时间差由手工设计的复杂时间编码函数处理，使 PointNet 风格 backbone 能够通过复数 max 聚合递归执行逐事件更新。相比之下，event2vec 使用相邻采样或聚合事件之间的一阶间隔 $\Delta\mathbf{t}[i] = \tilde{\mathbf{t}}[i] - \tilde{\mathbf{t}}[i-1]$。我们将这一相对间隔序列输入可学习的卷积时间嵌入模块。所得时间表示再与空间嵌入相加融合，形成与 Transformer 兼容的事件 token，如式 7 所示。

总结而言，尽管 EventNet 和 event2vec 都遵循将空间-极性信息与时间信息解耦的高层原则，但二者的

网络架构和计算目标显著不同。EventNet 面向使用 LUT 和对称 max 聚合的异步递归 CPU 处理；相反，event2vec 面向使用可学习嵌入的 GPU 高效 Transformer 架构。此外，event2vec 引入参数化空间嵌入以显式捕捉邻域语义，并使用强度因子 $\boldsymbol{\rho}$ 编码聚合事件密度，这些概念在 EventNet 的形式化表达中并不存在。

3.6. 网络结构

图 2 展示了网络架构，包括 (a) event2vec 的空间嵌入结构，(b) event2vec 的时间嵌入结构，以及 (c) 整体网络结构。详细信息见附录 A.2。

Event2Vec： 空间嵌入模块 ϕ 由三层线性层组成，特征数逐步从 $3 \to \frac{D}{4}$、$\frac{D}{4} \to \frac{D}{2}$ 增加到 $\frac{D}{2} \to D$。每个线性层后也插入 Layer Normalization (Ba et al., 2016) 以稳定训练。前两个 Layer Normalization 后使用 ReLU 激活。时间嵌入模块结构与空间嵌入模块类似，但将线性层替换为 kernel size 为 3、stride 为 1 的一维卷积层。第一层卷积将标量间隔序列映射到 $\frac{D}{4}$ 个通道，后两层卷积为 depthwise convolution。通道数逐步从 $1 \to \frac{D}{4}$、$\frac{D}{4} \to \frac{D}{2}$ 增加到 $\frac{D}{2} \to D$。

Backbone： Backbone 通过堆叠多个 Transformer block 构成，每个 block 包含一个自注意力层、一个由两层全连接层组成的前馈网络，以及一个用于降低序列长度的可选 pooling 层。我们在 backbone 中采用 Forgetting Transformer (Lin et al., 2025) 作为自注意力机制。Gated Linear Attention (Yang et al., 2024) 等线性注意力也可使用，但会带来轻微精度下降，相关结果见附录 A.3。需要注意，Forgetting Transformer 中的遗忘门对顺序敏感。为增强学习能力，我们将 Forgetting Transformer 扩展为参数共享的双向形式。更多细节见附录 A.4。

分类头： 我们使用平均池化层聚合序列中所有位置的特征，然后使用线性层进行分类决策。

4. 实验

我们在三个神经形态数据集 DVS Gesture、ASL-DVS 和 DVS-Lip 上开展了一系列分类任务实验。本

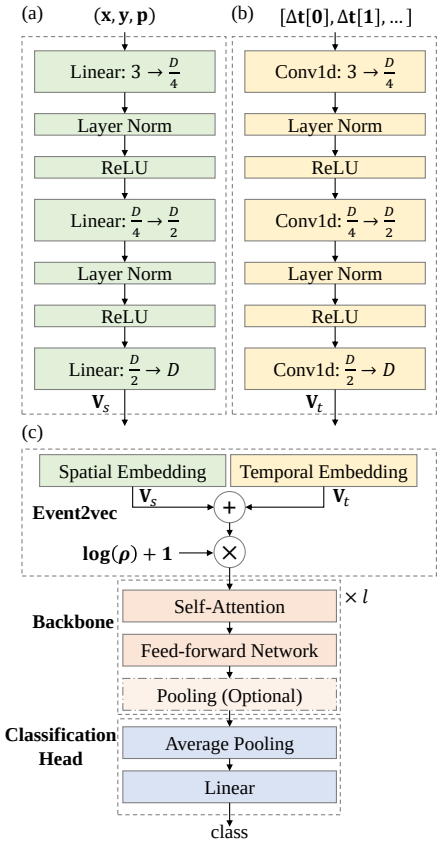

Figure 2. 使用 event2vec 表示进行事件分类的网络架构。

节结果以 $a \pm b$ 的形式报告，分别表示均值和标准差。对于涉及随机采样的实验，结果基于测试集上的 10 次独立运行计算得到。

4.1. 不同表示之间的比较

精度与参数效率 表 1 比较了 event2vec 与其他表示方法在三个数据集上的精度和模型参数量。对于 DVS Gesture 和 ASL-DVS，我们的模型直接在随机采样事件上训练。对于 DVS-Lip，我们的模型首先在簇事件上进行自监督预训练（见附录 A.5），随后报告在随机采样事件以及由本文提出的 Batched K-means++ 算法生成的簇事件上的微调精度。本文方法在 DVS Gesture 上取得了可比精度，并在 ASL-DVS 和 DVS-Lip 上达到领先表示方法中的最高精度，同时展现出出色的参数效率。例如，之前的 SOTA 模型在三个数据集上的参数量分别是本文模型的 2.79×、815.93× 和 12.22×。

吞吐量、延迟与内存 我们进一步比较了本文模型与此前 SOTA 模型的吞吐量和延迟，结果见表 2。

Table 1. 神经形态数据集上的模型性能与模型大小比较。

数据集	方法 + 表示	精度 (%)	参数量 (MB)
DVS Gesture	Sparse GRU + Frame (Subramoney et al., 2023)	97.80	4.80
	SNN + Frame (Yao et al., 2023)	98.23	6.50
	FARSE-CNN + Window Slicing (Santambrogio et al., 2024)	96.6	10.79
	Event MAE + Point Cloud (Sun et al., 2025)	97.75	N/A
	Max-Former + Frame (Fang et al., 2025)	**98.6**	1.45
	Transformer + Event2Vec (4096 个随机事件)	97.57±1.31	**0.52**
ASL-DVS	GNN, CNN + Graph (Bi et al., 2019)	90.10	19.46
	GNN & Transformer + Image & Voxel Graph (Yuan et al., 2023)	99.60	220.30
	Transformer + Event2Vec (1024 个随机事件)	**99.91±0.05**	**0.27**
DVS-Lip	ResNet-18 & BiGRU + Frame (Tan et al., 2022)	72.1	241.20
	Spiking ResNet18 & BiGRU + Frame (Dampfhoffer & Mesquida, 2024)	75.3	223.63
	Transformer + Event2Vec (1024 个随机事件)	70.62±1.55	**18.30**
	(1024 个 Batched K-Means++ 聚类事件)	**75.88**	

表 2 中用于比较的三个模型为 Max-Former (Fang et al., 2025)、GNN & Transformer (Yuan et al., 2023) 和 Spiking ResNet18 & BiGRU (Dampfhoffer & Mesquida, 2024)。这些模型均提供官方开源代码，因此我们能够基于其代码库进行实验。附录 A.6 提供了这些实验的更多细节。吞吐量是衡量模型训练效率的主要指标。对于不同模型，我们在不超过 GPU 内存限制的前提下，将 batch size 设置为尽可能大的 2 的幂，或两个相邻 2 的幂的平均值（例如 64、96、128、...），以最大化报告的吞吐量。结果表明，event2vec 充分利用了 Transformer 的计算效率，在三个数据集上的训练和推理吞吐量分别比此前工作高 4.21× 和 2.69×、11.96× 和 62.67×、以及 35.36× 和 5.70×。对于边缘设备上的推理任务（例如嵌入式神经形态系统），处理单个事件流的延迟以及模型消耗的 GPU 内存非常关键。我们进行了对比实验，并测量了三个延迟组成部分：事件数据预处理延迟、模型前向传播延迟和总延迟。结果表明，在三个数据集上，本文模型的总延迟分别仅为此前 SOTA 方法的 $68.55\%, 11.12\%$ 和 14.68%，内存消耗也仅为其 $72.18\%, 15.08\%$ 和 68.35%。

4.2. 消融实验

嵌入比较 我们在 DVS Gesture 数据集上进行了消融研究，以评估不同组件对精度的贡献，详见表 4。我们测试了空间嵌入方法（standard（式 2）与 parametric（式 5））和时间嵌入模块（基于 $\mathbf{t}$ 的正弦嵌入与基于 $\Delta\mathbf{t}$ 的卷积嵌入）的不同组合。标准嵌入与本文卷积时间嵌入的组合（Standard + Conv($\Delta\mathbf{t}$)）取得最低精度。我们认为这是因为标准嵌入层缺乏归纳偏置，无法有效学习邻域语义，并进一步限制了卷积时间编码器的性能。因此，当使用本文的参数化嵌入时，卷积编码器取得最高精度。值得注意的是，无论与哪种时间嵌入搭配，参数化嵌入都始终优于标准版本，验证了引入邻域语义的有效性。

对事件数量的鲁棒性 处理更少事件会降低资源消

Table 2. Event2Vec 与此前 SOTA 模型的吞吐量和延迟比较。

数据集	方法	批量吞吐量 (samples/s)		单流推理			
		训练	推理	延迟 (ms)			GPU 内存 (MB)
				数据预处理	前向传播	总计	
DVS Gesture	Max-Former	241.12 ± 0.55	1077.35 ± 2.20	10.29 ± 0.23	23.89 ± 11.65	34.18 ± 11.75	834
	Event2Vec	1016.19 ± 61.18	2900.08 ± 277.30	9.92 ± 5.79	13.51 ± 9.04	23.43 ± 10.63	602
ASL-DVS	GNN & Transformer	78.08 ± 6.39	200.16 ± 12.66	55.12 ± 38.35	10.90 ± 0.45	66.02 ± 38.41	5464
	Event2Vec	933.58 ± 16.14	12543.81 ± 2565.54	1.10 ± 0.23	6.24 ± 2.89	7.34 ± 2.91	824
DVS-Lip	Spiking ResNet18 & BiGRU	10.85 ± 0.03	165.29 ± 2.25	373.82 ± 2.21	15.07 ± 7.73	388.89 ± 8.14	1226
	Event2Vec	383.71 ± 0.89	942.29 ± 22.36	17.23 ± 3.33	39.87 ± 1.28	57.10 ± 3.62	838

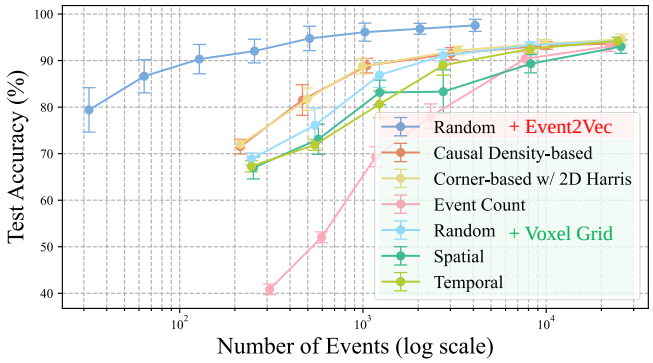

Figure 3. 在 DVS Gesture 数据集上，与 (Araghi et al., 2025) 中采样技术相比的精度-事件数量关系。

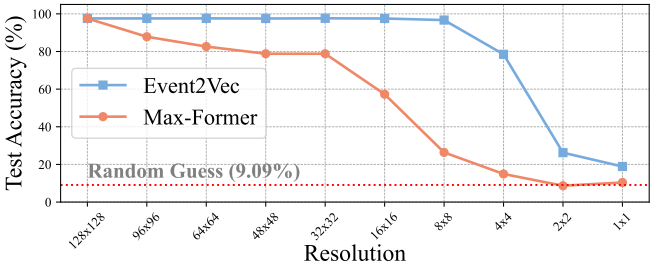

Figure 4. 精度与空间分辨率关系：在 DVS Gesture 上与 SOTA Max-Former (Fang et al., 2025) 的比较。

耗，这在事件应用中始终是期望的。图 3 将本文方法与 (Araghi et al., 2025) 中基于体素网格表示的复杂采样技术进行了比较。结果突出了 event2vec 的内在有效性：即使仅搭配简单随机采样，它也能持续优于体素网格表示，即便后者采用了更复杂、精心设计的采样策略。附录 A.7 给出了 event2vec 模型性能随事件数量变化的更多细节。

对空间分辨率的鲁棒性 我们进一步测试了 event2vec 对传感器空间分辨率变化的鲁棒性，并在 DVS Gesture 数据集上与 Max-Former (Fang et al., 2025) 进行了比较。对于 event2vec，我们将坐标视为浮点数，先缩放到目标分辨率 $h \times w$ 后量化；随后再将坐标上采样回原始分辨率 $H \times W$ 并重新量化。对于采用帧表示的 Max-Former，我们使用双线性插值将帧下采样到较低分辨率 $h \times w$，再上采样回原始分辨率 $H \times W$。图 4 的结果表明，event2vec 对分辨率变化具有强鲁棒性。此外，即使分辨率降低到 1×1（即完全丧失空间信息），它仍能保持显著高于随机猜测的分类精度，这也从侧面验证了时间嵌入的有效性。

通过线性探测评估表示学习 为验证模型是否能够学习通用特征表示，我们采用线性探测这一常用指标 (Alain & Bengio, 2017; Radford et al., 2021) 进行评估。具体而言，用于 DVS-Lip 分类的 event2vec 模型在三个分类模型中参数量最大。因此，我们使用在 DVS-Lip 上训练得到的模型，在 DVS Gesture 和 ASL-DVS 上进行线性探测。

我们首先考察 DVS Gesture 和 ASL-DVS 本身是否线性可分。表 3 第一行给出了不使用 event2vec 或 Transformer 时的线性探测精度。可以看到，两个数据集上的精度都显著较低。随后，我们将原始事件输入替换为 event2vec 编码后的嵌入；这些结果见表 3 第二行和第三行。本文提出的 Parametric Spatial + Convolutional Temporal 嵌入带来了显著影响，在两个数据集上的精度提升均超过 20 个百分点。尽管 “Standard Spatial + Sin Temporal” 这一 event2vec baseline 表现不如参数化版本，但它仍优于原始事

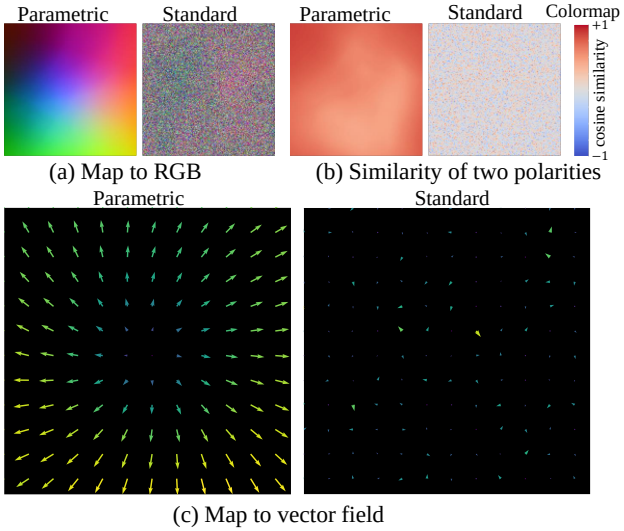

Figure 5. 学习到的空间嵌入的可视化比较。

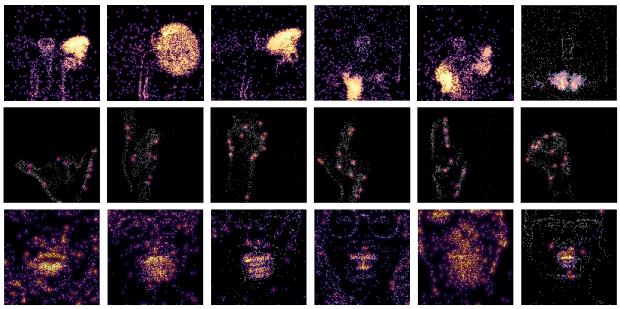

Figure 6. DVS Gesture（第 1 行）、ASL-DVS（第 2 行）和 DVS-Lip（第 3 行）样本上的事件级注意力图。

件。

进一步地，我们使用在 DVS-Lip 上训练得到的模型中的 event2vec 组件，以及 16 层 backbone 网络的前 5 层。我们选择 5 层是因为实验发现该深度性能最优；使用更少或更多层都会导致性能略有下降。我们冻结提取出的子模型参数，并在其后接入一个可训练分类头。结果见表 3 第四行和第五行。可以看到，两个数据集上的精度继续显著提升。这些结果表明，带有 event2vec 和多层 Transformer 架构的模型在从训练数据集迁移到其他数据集时，能够生成具有较高线性可分性的特征，说明模型学习到了一种通用的特征表示方法。

聚类延迟 表 5 比较了在 DVS-Lip 数据集上应用不同 K-Means 聚类方法时，测试集上的平均单样本聚类延迟，以及使用聚类数据训练的模型在测试集上的精度。我们将本文方法与两个基准进行比较：Scikit-learn 基于 CPU 的 K-Means（使用 K-Means++）以及 Meta 的 Faiss GPU K-Means (Johnson et al., 2019)。如表 5 所示，本文提出的 batched K-Means++ 在延迟接近最快设置的同时取得最高精度。值得注意的是，它显著快于 Scikit-learn，且比 GPU 加速的 Faiss 更准确。虽然 Faiss（iters=20）具有相近速度，但相较本文方法会出现 1.48% 的精度下降。

4.3. 可视化

邻域语义 为直观检查邻域语义，我们从在 DVS Gesture 数据集上训练得到的参数化嵌入层（$\mathbf{W}_\phi$）和标准嵌入层（$\mathbf{W}_s$）中提取空间嵌入权重。对于每个坐标 (x, y, p)，其 D 维嵌入向量通过主成分分析（Principal Component Analysis, PCA）投影到三维空间。这些 3D 向量随后被解释为 RGB 颜色值，并绘制在对应的 (x, y) 位置上形成图像。图 5(a) 可视化了极性 0 的结果图像（极性 1 的图像见附录 A.8）。由 $\mathbf{W}_\phi$ 得到的图像呈现平滑连续的颜色梯度，类似色盘，表明空间相邻坐标具有语义相似的嵌入。相比之下，来自 $\mathbf{W}_s$ 的图像近似随机噪声，说明其缺乏学习到的空间相关性。

极性相似性 当物体边缘经过某个像素时，通常会在短时间内触发两种极性的事件。因此，我们假设同一空间位置上相反极性的嵌入也应具有语义关联。为验证这一点，我们计算每个坐标处两种极性嵌入向量之间的余弦相似度。如图 5(b) 所示，参数化嵌入能够捕捉这种关系，呈现出明显的高相似度区域。相反，标准嵌入的相似度图大多接近零，说明其未能学习这种极性间相关性。

向量场表示 我们将学习到的空间流形可视化为向量场。D 维嵌入向量通过 PCA 投影到前两个主成分上。所得 2D 向量再通过 quiver plot 可视化，其中每个箭头表示其空间坐标处向量的方向和大小。图 5(c) 展示了结果。参数化嵌入的向量场呈现出一致、类似层流的流动，揭示了平滑结构化的语义空间。相比之下，标准嵌入的向量场显得混乱且湍流化，进一步证明其无法捕捉有意义的空间关系。

逐事件注意力 由于 event2vec 是逐事件表示，其注意力机制可以在细粒度事件级别进行可视化。图 6

Table 3. 不同 event2vec 方法以及是否使用 Transformer 层时的线性探测精度比较。

Event2Vec	Transformer	DVS Gesture	ASL-DVS
无	无	35.52±1.35%	12.65±14.61%
参数化空间 + 卷积时间	无	56.32±2.56%	38.90±10.48%
标准空间 + 正弦时间	无	37.36±2.22%	26.63±10.91%
参数化空间 + 卷积时间	前 5 层	86.94±1.21%	69.83±7.66%
标准空间 + 正弦时间	前 5 层	84.44±3.35%	59.71±8.76%

Table 4. DVS Gesture 上的嵌入消融分析。

空间嵌入	时间嵌入	精度 (%)
标准	Conv(Δ**t**)	91.18±3.70
标准	Sin(**t**)	93.16±2.19
参数化	Sin(**t**)	96.56±1.46
参数化	Conv(Δ**t**)	97.57±1.31

Table 5. DVS-Lip 上 K-Means 聚类延迟与精度。

方法	延迟 (ms)	精度 (%)
Batched K-means++ (batch size = 64, iters=20)	17.10±15.56	75.88
Scikit-learn (iters=300)	383.42 ± 149.22	75.08
Scikit-learn (iters=20)	374.22 ± 145.97	74.72
Faiss (iters=300)	162.41±126.24	74.12
Faiss (iters=20)	15.95±17.59	74.40

展示了叠加在原始事件流上的注意力热力图，分别对应 DVS Gesture（第 1 行）、ASL-DVS（第 2 行）和 DVS-Lip（第 3 行）。可视化结果表明，模型能够正确关注 DVS Gesture 中的手部、ASL-DVS 中的手指关节和轮廓，以及 DVS-Lip 中的唇部区域。然而，与 DVS-Lip 相较另外两个数据集更低的分类精度一致，我们也观察到模型有时会错误地将较多注意力分配给其他面部特征，例如眼睛和耳朵。

5. 结论

神经形态事件相机为计算机视觉带来了范式转变，同时也带来了独特机遇与重大挑战。核心挑战在于如何协调其异步、稀疏特性与深度学习中同步、稠密、规则的架构。在本文中，我们提出 event2vec，这是一种新型表示，通过使神经网络能够原生处理异步事件来直接应对这一挑战。实验结果表明，event2vec 在精度上可与成熟方法竞争，同时在参数效率、预处理开销、吞吐量以及对不同事件数量和空间分辨率的鲁棒性方面具有显著优势。 event2vec 出色的效率和鲁棒性表明，它在资源受限边缘设备的实时部署中具有重要潜力，而这类场景高度依赖低延迟感知和低功耗。除这些性能指标外，event2vec 最重要的贡献在于从概念上将事件流与自然语言处理范式对齐。这为研究和应用开辟了新的方向。通过将事件视为一种序列语言，我们可以利用为大语言模型发展出的复杂架构，进一步探索新的应用。

致谢

本工作部分受到 CoCoSys（由 DARPA 和 SRC 资助的 JUMP2.0 中心）、美国国家科学基金会（CAREER Award，Grant #2312366，Grant #2318152）、DARPA Young Faculty Award、DoE MMICC center SEA-CROGS（Award #DE-SC0023198）以及 Global Industrial Technology Cooperation Center（GITCC）项目支持。

影响声明

本文工作的目标是推动机器学习领域发展。我们的工作可能产生多方面的社会影响，但我们认为其中没有需要特别强调的具体影响。

参考文献

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., and Zheng, X. Tensorflow: a system for large-scale machine learning. In *Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation*, OSDI'16, pp. 265–283, USA, 2016. USENIX Association. ISBN 9781931971331.

Alain, G. and Bengio, Y. Understanding intermediate layers using linear classifier probes. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=HJ4-rAVtl.

Amir, A., Taba, B., Berg, D., Melano, T., McKinstry, J., Di Nolfo, C., Nayak, T., Andreopoulos, A., Garreau, G., Mendoza, M., Kusnitz, J., Debole, M., Esser, S., Delbruck, T., Flickner, M., and Modha, D. A low power, fully event-based gesture recognition system. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7388–7397, 2017. doi: 10.1109/CVPR.2017.781.

Araghi, H., van Gemert, J., and Tomen, N. Making Every Event Count: Balancing Data Efficiency and Accuracy in Event Camera Subsampling . In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 5044–5054, Los Alamitos, CA, USA, June 2025. IEEE Computer Society. doi: 10.1109/CVPRW67362.2025.00499.

Arthur, D. and Vassilvitskii, S. k-means++: the advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, pp. 1027–1035, USA, 2007. Society for Industrial and Applied Mathematics. ISBN 9780898716245.

Ba, J. L., Kiros, J. R., and Hinton, G. E. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Barchid, S., Mennesson, J., and Djéraba, C. Bina-rep event frames: A simple and effective representation for event-based cameras. In *IEEE International Conference on Image Processing*, pp. 3998–4002. IEEE, 2022.

Bardow, P., Davison, A. J., and Leutenegger, S. Simultaneous optical flow and intensity estimation from an event camera. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2016.

Bi, Y., Chadha, A., Abbas, A., Bourtsoulatze, E., and Andreopoulos, Y. Graph-based object classification for neuromorphic vision sensing. In *IEEE/CVF International Conference on Computer Vision*, pp. 491–501, 2019. doi: 10.1109/ICCV.2019.00058.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A.,

Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901, Virtual, 2020. Curran Associates, Inc.

Cordone, L., Miramond, B., and Thierion, P. Object detection with spiking neural networks on automotive event data. In *International Joint Conference on Neural Networks*, pp. 1–8. IEEE, 2022.

Dampfhoffer, M. and Mesquida, T. Neuromorphic lipreading with signed spiking gated recurrent units. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2141–2151, 2024.

Davies, M., Srinivasa, N., Lin, T.-H., Chinya, G., Cao, Y., Choday, S. H., Dimou, G., Joshi, P., Imam, N., Jain, S., Liao, Y., Lin, C.-K., Lines, A., Liu, R., Mathaikutty, D., McCoy, S., Paul, A., Tse, J., Venkataramanan, G., Weng, Y.-H., Wild, A., Yang, Y., and Wang, H. Loihi: a neuromorphic manycore processor with on-chip learning. *IEEE Micro*, 38(1):82–99, 2018. doi: 10.1109/MM.2018.112130359.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In Burstein, J., Doran, C., and Solorio, T. (eds.), *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423.

Du, K., Wu, Y., Deng, S., and Gu, S. Temporal flexibility in spiking neural networks: Towards generalization across time steps and deployment friendliness. In *International Conference on Learning Representations*, 2025.

Fang, Y., Zhou, D., Wang, Z., Ren, H., Zeng, Z., Li, L., shibo zhou, and Xu, R. Spiking neural networks need high-frequency information. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=owNPAl7LNK.

Gallego, G., Delbrück, T., Orchard, G., Bartolozzi, C., Taba, B., Censi, A., Leutenegger, S., Davison, A. J., Conradt, J., Daniilidis, K., and Scaramuzza, D. Event-based vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(1):154–180, 2022. doi: 10.1109/TPAMI.2020.3008413.

Gehrig, D., Loquercio, A., Derpanis, K. G., and Scaramuzza, D. End-to-end learning of representations for asynchronous event-based data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5633–5643, 2019.

Gonzalez, R. C. *Digital image processing*. Pearson education india, India, 2009.

Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Vaughan, A., et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., and Oliphant, T. E. Array programming with numpy. *Nature*, 585 (7825):357–362, Sep 2020. ISSN 1476-4687. doi: 10.1038/s41586-020-2649-2.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition*

(CVPR), pp. 770–778, 2016. doi: 10.1109/CVPR.2016.90.

Johnson, J., Douze, M., and Jégou, H. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2019.

Katharopoulos, A., Vyas, A., Pappas, N., and Fleuret, F. Transformers are RNNs: Fast autoregressive transformers with linear attention. In III, H. D. and Singh, A. (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5156–5165, Virtual, 13–18 Jul 2020. PMLR.

Larsson, G., Maire, M., and Shakhnarovich, G. Fractalnet: Ultra-deep neural networks without residuals. *arXiv preprint arXiv:1605.07648*, 2016.

LeCun, Y., Bengio, Y., and Hinton, G. Deep learning. *Nature*, 521(7553):436–444, May 2015. ISSN 1476-4687. doi: 10.1038/nature14539.

Lichtsteiner, P., Posch, C., and Delbruck, T. A 128× 128 120 db 15 μs latency asynchronous temporal contrast vision sensor. *IEEE Journal of Solid-State Circuits*, 43 (2):566–576, 2008. doi: 10.1109/JSSC.2007.914337.

Lin, X., Qiu, C., Shen, S., Zang, Y., Liu, W., Bian, X., Müller, M., Wang, C., et al. E2pnet: event to point cloud registration with spatio-temporal representation learning. *Advances in Neural Information Processing Systems*, 36:18076–18089, 2023.

Lin, Z., Nikishin, E., He, X., and Courville, A. Forgetting transformer: Softmax attention with a forget gate. In *International Conference on Learning Representations*, 2025.

Liu, M. and Delbruck, T. Adaptive time-slice block-matching optical flow algorithm for dynamic vision sensors. In *British Machine Vision Conference*, pp. 88. BMVA Press, 2018.

Loshchilov, I. and Hutter, F. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

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

Mead, C. Neuromorphic electronic systems. *Proceedings of the IEEE*, 78(10):1629–1636, 1990. doi: 10.1109/5.58356.

Merolla, P. A., Arthur, J. V., Alvarez-Icaza, R., Cassidy, A. S., Sawada, J., Akopyan, F., Jackson, B. L., Imam, N., Guo, C., Nakamura, Y., et al. A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science*, 345(6197): 668–673, 2014.

Messikommer, N., Gehrig, D., Loquercio, A., and Scaramuzza, D. Event-based asynchronous sparse convolutional networks. In Vedaldi, A., Bischof, H., Brox, T., and Frahm, J.-M. (eds.), *Computer Vision – ECCV 2020*, pp. 415–431, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58598-3.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. Distributed representations of words and phrases and their compositionality. In Burges, C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. (eds.), *Advances in Neural Information Processing Systems*, volume 26, Lake Tahoe, Nevada, USA, 2013. Curran Associates, Inc.

Nt, H. and Maehara, T. Revisiting graph neural networks: All we have is low-pass filters. *arXiv preprint arXiv:1905.09550*, 2019.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N.,

Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32, 2019.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Peng, Y., Zhang, Y., Xiong, Z., Sun, X., and Wu, F. Get: Group event transformer for event-based vision. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6038–6048, 2023.

Posch, C., Matolin, D., and Wohlgenannt, R. A qvga 143 db dynamic range frame-free pwm image sensor with lossless pixel-level video compression and time-domain cds. *IEEE Journal of Solid-State Circuits*, 46 (1):259–275, 2011. doi: 10.1109/JSSC.2010.2085952.

Press, O., Smith, N. A., and Lewis, M. Train short, test long: Attention with linear biases enables input length extrapolation. *arXiv preprint arXiv:2108.12409*, 2021.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. Learning transferable visual models from natural language supervision. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Rebecq, H., Ranftl, R., Koltun, V., and Scaramuzza, D. Events-to-video: Bringing modern computer vision to event cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3857–3866, 2019.

Ren, H., Zhou, Y., Zhu, J., Lin, X., Fu, H., Huang, Y., Fang, Y., Ma, F., Yu, H., and Cheng, B. Rethinking efficient and effective point-based networks for event camera classification and regression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(8):6228–6241, 2025. doi: 10.1109/TPAMI.2025.3556561.

Roy, K., Jaiswal, A., and Panda, P. Towards spike-based machine intelligence with neuromorphic computing. *Nature*, 575(7784):607–617, 2019.

Sabater, A., Montesano, L., and Murillo, A. C. Event transformer+. a multi-purpose solution for efficient event data processing. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 45(12):16013–16020, 2023.

Santambrogio, R., Cannici, M., and Matteucci, M. Farse-cnn: Fully asynchronous, recurrent and sparse event-based cnn. In *European Conference on Computer Vision*, pp. 1–18. Springer, 2024.

Schaefer, S., Gehrig, D., and Scaramuzza, D. Aegnn: Asynchronous event-based graph neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12361–12371, 2022. doi: 10.1109/CVPR52688.2022.01205.

Schuster, M. and Paliwal, K. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997. doi: 10.1109/78.650093.

Sekikawa, Y., Hara, K., and Saito, H. Eventnet: Asynchronous recursive event processing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3887–3896, 2019.

Su, J., Ahmed, M., Lu, Y., Pan, S., Bo, W., and Liu, Y. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568: 127063, 2024. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2023.127063. URL https://www.sciencedirect.com/science/article/pii/S0925231223011864.

Subramoney, A., Nazeer, K. K., Schöne, M., Mayr, C., and Kappel, D. Efficient recurrent architectures through activity sparsity and sparse back-propagation through time. In *International Conference on Learning Representations*, 2023.

Sun, J., Zhang, Q., Wang, J., Cao, J., Cheng, H., and Xu, R. Event masked autoencoder: Point-wise action recognition with event-based cameras. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 1–5, 2025. doi: 10.1109/ICASSP49660.2025.10888760.

Tan, G., Wang, Y., Han, H., Cao, Y., Wu, F., and Zha, Z.-J. Multi-grained spatio-temporal features perceived network for event-based lip-reading. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20094–20103, June 2022.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 30, Long Beach, California, USA, 2017. Curran Associates, Inc.

Wu, Y. and He, K. Group normalization. In *Proceedings of the European conference on computer vision*, pp. 3–19, 2018.

Yang, J., Zhang, Q., Ni, B., Li, L., Liu, J., Zhou, M., and Tian, Q. Modeling point clouds with self-attention and gumbel subset sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3323–3332, 2019.

Yang, S., Wang, B., Shen, Y., Panda, R., and Kim, Y. Gated linear attention transformers with hardware-efficient training. In Salakhutdinov, R., Kolter, Z., Heller, K., Weller, A., Oliver, N., Scarlett, J., and Berkenkamp, F. (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 56501–56523, Seoul, South Korea, 21–27 Jul 2024. PMLR.

Yao, M., Zhao, G., Zhang, H., Hu, Y., Deng, L., Tian, Y., Xu, B., and Li, G. Attention spiking neural networks. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 45(8):9393–9410, 2023.

Yao, M., Richter, O., Zhao, G., Qiao, N., Xing, Y., Wang, D., Hu, T., Fang, W., Demirci, T., De Marchi, M., Deng, L., Yan, T., Nielsen, C., Sheik, S., Wu, C., Tian, Y., Xu, B., and Li, G. Spike-based dynamic computing with asynchronous sensing-computing neuromorphic chip. *Nature Communications*, 15(1):4464, May 2024. ISSN 2041-1723. doi: 10.1038/s41467-024-47811-6.

Yu, X., Tang, L., Rao, Y., Huang, T., Zhou, J., and Lu, J. Point-bert: Pre-training 3d point cloud transformers with masked point modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Yuan, C., Jin, Y., Wu, Z., Wei, F., Wang, Y., Chen, L., and Wang, X. Learning bottleneck transformer for event image-voxel feature fusion based classification. In *Chinese Conference on Pattern Recognition and Computer Vision*, pp. 3–15. Springer, 2023.

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., and Sun, M. Graph neural networks: A review of methods and applications. *AI Open*, 1: 57–81, 2020. ISSN 2666-6510. doi: https://doi.org/10.1016/j.aiopen.2021.01.001.

Zhu, A. Z., Yuan, L., Chaney, K., and Daniilidis, K. Unsupervised event-based learning of optical flow, depth, and egomotion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 989–997, 2019.

A. 附录

A.1. Batched K-Means++ 聚类算法

对于 DVS-Lip 等具有挑战性的分类任务，随机采样会造成显著的信息损失，从而导致精度不理想。为确保所有事件都能参与最终表示，我们采用事件聚类方法。给定包含 N 个事件的原始事件流 $\mathcal{E} = \{(x_i, y_i, t_i, p_i)\}_{i=0}^{N-1}$，我们的目标是生成至多 L 个聚类事件 $\mathcal{R} = \{(x_{c,j}, y_{c,j}, t_{c,j}, p_{c,j}, \rho_j)\}$，其中 $(x_{c,j}, y_{c,j}, t_{c,j}, p_{c,j})$ 表示第 j 个聚类中心的坐标，ρ_j 表示该聚类中的事件数量。需要注意的是，我们对两种极性分别进行聚类，以避免极性混合造成物理意义丢失。具体而言，假设两种极性的事件数量分别为 N_0 和 N_1，我们近似按照 N_0 和 N_1 的比例分配聚类数量，并根据每种极性中可用事件的数量进行裁剪。最后，将两组聚类结果合并，并按照时间顺序排序。

K-Means 聚类通常会使用 Python 机器学习库 Scikit-learn (sklearn) (Pedregosa et al., 2011) 提供的 K-Means 函数。然而，该函数在 CPU 上实现，当事件数量较大时执行速度较慢，会显著增加模型处理实时任务的延迟。为解决这一问题，我们提出一种基于 GPU 的 Batched K-Means++ 事件聚类算法，其详细流程见算法 1。该算法的加速主要来自以下优化：

1. **降低循环开销**：循环迭代次数减少为原来的 $1/B$。
2. **并行计算**：使用 `torch.cdist` 并行计算所有点到一批 B 个新中心的距离。
3. **增量更新**：更新 $\mathbf{D}^2$ 时只比较当前已知的最小距离与到新加入中心批次的距离，避免重新计算到所有已选中心的距离。

A.2. 模型结构与超参数

除非另有说明，所有模型均使用 BFloat16 混合精度训练。所有模型的训练配置包括基础学习率 $lr_b = 0.001$、每块 GPU 的批大小 64，以及 AdamW 优化器 (Loshchilov & Hutter, 2019)，训练 64 个 epoch。有效学习率由线性缩放规则确定，该规则基于每块 GPU 的批大小 ($batch\_size$) 以及分布式数据并行训练中使用的 GPU 数量 (n_{gpus})：$lr = lr_b \cdot batch\_size \cdot n_{gpus}/256$。前 4 个 epoch 使用 warmup 阶段，在此期间学习率从 $0.01 \cdot lr$ 线性增加到 lr。随后的 epoch 使用余弦退火调度 (Loshchilov & Hutter, 2017)，逐步将学习率降低到最小值 lr_{min}。对于 DVS Gesture 和 ASL-DVS 数据集，我们禁用了权重衰减和标签平滑。相比之下，对于 DVS-Lip 分类任务，我们将权重衰减设为 0.05，并采用系数为 0.1 的标签平滑。

表 6 汇总了各模型的详细超参数。其中，D 表示嵌入维度，l 表示骨干网络中 Transformer 块的数量，D_f 表示前馈神经网络 (FFN) 的隐藏特征维度，n_{head} 表示注意力头总数。`repeats` 参数指定单个 epoch 内训练集被遍历的次数。对于 Forgetting Transformer，键 (**k**) 和值 (**v**) 投影的头数设为 $\max(\lfloor n_{head}/2 \rfloor, 1)$，查询和键投影采用 RMS 组归一化 (Wu & He, 2018)。在 DVS Gesture 和 DVS-Lip 实验中，我们使用梯度裁剪将梯度的 L_2 范数限制在 1.0。

对于 DVS Gesture 分类模型，骨干网络各阶段之间采用步幅为 2 的序列平均池化，而其他模型不使用序列池化。 DVS-Lip 分类任务的模型使用自监督学习方法在 DVS-Lip 数据集上进行预训练。该预训练阶段使用的最小学习率为 $lr_{min} = 10^{-6}$，权重衰减为 0.05，`repeats` 值为 3，掩码比例为 30%。更多细节见附录 A.5。

Algorithm 1: GPU 上的 Batched K-Means++ 事件聚类

输入: 原始事件流 $\mathcal{E} = \{(x_i, y_i, t_i, p_i)\}_{i=0}^{N-1}$，总点数 N
参数: 最大目标聚类数 L，空间尺寸 H, W，批大小 B，最大迭代次数 I_{max}，容忍阈值 tol
输出: 聚类事件集合 $\mathcal{R}$

/* 1. 数据预处理与归一化 */
将数据移至 GPU
计算时间跨度 $t_{span} = t_{N-1} - t_0$ 和归一化时间 $\hat{t}_i = (t_i - t_0)/t_{span}$
构造三维特征空间点集 $\mathbf{V} = \{(x_i/(W-1), y_i/(H-1), \hat{t}_i)\}_{i=0}^{N-1}$
根据极性 $\mathbf{p}$ 将 $\mathbf{V}$ 划分为特定极性的集合 $\mathbf{V}^0$ 和 $\mathbf{V}^1$
近似按照 N_0 和 N_1 的比例分配特定极性的目标中心数量，并按每种极性的可用事件数量进行裁剪
初始化结果集合 $\mathcal{R} = \emptyset$
/* 对每种极性分别聚类 */
for 每个点集 $\mathbf{V}_{sub} \in \{\mathbf{V}^0, \mathbf{V}^1\}$ 及其目标数量 $K_{sub} \in \{L_0, L_1\}$ **do**
    if $|\mathbf{V}_{sub}| == 0$ *or* $K_{sub} == 0$ **then**
        continue
    /* 阶段 1: Batched K-Means++ 初始化 */
    随机选择第一个中心 $\mathbf{c}_0 \in \mathbf{V}_{sub}$，初始化中心集合 $\mathbf{C} = \{\mathbf{c}_0\}$
    计算所有点到第一个中心的平方距离 $\mathbf{D}^2 = \|\mathbf{V}_{sub} - \mathbf{c}_0\|^2$
    while $|\mathbf{C}| < K_{sub}$ **do**
        计算当前批次的采样数量 $M = \min(B, K_{sub} - |\mathbf{C}|)$
        /* 并行采样：使用当前距离作为概率权重 */
        根据权重 $\mathbf{w} \propto \mathbf{D}^2$ 无放回地采样 M 个新候选中心 $\mathbf{C}_{new}$
        将 $\mathbf{C}_{new}$ 加入 $\mathbf{C}$
        /* 增量距离更新：仅针对新加入的中心 */
        $\mathbf{D}^2_{new} = \min_{c \in \mathbf{C}_{new}} \|\mathbf{V}_{sub} - c\|^2$
        更新全局最小距离 $\mathbf{D}^2 \leftarrow \min(\mathbf{D}^2, \mathbf{D}^2_{new})$
    /* 阶段 2: 标准 Lloyd 迭代 */
    for $iter = 0$ ***to*** $I_{max} - 1$ **do**
        E 步：根据 $\mathbf{C}$ 中最近的中心为点分配标签
        M 步：计算每个聚类的质心作为新的 $\mathbf{C}$
        if 中心偏移 $< tol$ **then**
            break
    /* 反归一化与强度计算 */
    统计每个聚类中的点数作为强度 $\boldsymbol{\rho}$
    将 $\mathbf{C}$ 的坐标映射回物理尺度 $(W-1, H-1, t_{span})$，并加上起始时间戳 t_0
    将结果 $(x_{c,j}, y_{c,j}, t_{c,j}, p_{sub}, \rho_j)$ 加入 $\mathcal{R}$
/* 3. 后处理 */
合并 $\mathcal{R}$ 中的所有结果
按时间 $\mathbf{t}_c$ 对结果排序，以恢复时间顺序
返回 $\mathcal{R}$

Table 6. 不同数据集分类任务中模型训练的超参数。

数据集	D	D_f	n_{head}	l	Repeats	n_{gpus}	lr_{min}
DVS Gesture	64	128	2	4	24	4	0
ASL-DVS	64	128	2	2	1	7	10^{-6}
DVS-Lip	192	384	6	16	3	4	10^{-6}

Table 7. 不同类型自注意力的精度比较。

数据集	FoX 精度 (%)	GLA 精度 (%)
DVS Gesture	97.57 ± 1.31	96.67±0.67
ASL-DVS	99.91±0.05	99.85±0.12
DVS-Lip	75.88	72.35

A.3. 不同自注意力类型下的精度

除 Forgetting Transformer (FoX) 外，我们还评估了使用 Gated Linear Attention (GLA) 时的性能，结果见表 7。结果表明，GLA 在 DVS Gesture 和 ASL-DVS 等相对简单的分类任务上取得了较高性能；然而，在更具挑战性的 DVS-Lip 分类任务上，其性能下降了 3.53 个百分点，这可能是因为线性注意力在处理长输入序列时难以防止长期记忆衰减。总体而言，这些结果表明 Event2Vec 仍然兼容不同的注意力机制，而 FoX 更适合更具挑战性的长序列任务。

A.4. 双向自注意力

GLA 是典型的线性注意力机制，可被视为循环神经网络 (RNN) (Katharopoulos et al., 2020) 的一种特殊情况，其输入序列顺序会影响输出结果。尽管 FoX 不属于线性注意力范畴，但它采用了依赖输入序列顺序的 RNN 式门控机制，因此输入顺序同样会影响其输出。由于隐藏状态的大小固定且有限，RNN 在处理超长序列时不可避免地存在长距离信息衰减。为缓解长期记忆退化，我们将 FoX 扩展为双向变体。

我们通过同时输入正向和反向的 $\mathbf{Q}, \mathbf{K}, \mathbf{V}$，将该形式改为双向形式。双向输出计算如下：

$$\mathbf{O}_f = \text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}), \tag{8}$$

$$\mathbf{O}_b = \text{Reverse}(\text{Attention}(\overleftarrow{\mathbf{Q}}, \overleftarrow{\mathbf{K}}, \overleftarrow{\mathbf{V}})), \tag{9}$$

$$\mathbf{O}_{fb}[t] = \mathbf{W}_{fb}[\mathbf{O}_f[t]; \mathbf{O}_b[t]]. \tag{10}$$

与经典双向 RNN (Schuster & Paliwal, 1997) 通常为每个方向使用独立参数不同，我们的模型在两个方向上共享参数。通过在两次计算中共享投影权重 ($\mathbf{W}_q, \mathbf{W}_k, \mathbf{W}_v, \mathbf{W}_{forget}$)，我们确保参数量与单向基线基本相当，唯一的增加来自融合输出投影 $\mathbf{W}_{fb}$。

我们还测试了不共享参数的双向自注意力带来的性能变化，结果汇总于表 8。结果表明，在不共享参数时，参数数量增加约 25%。尽管理论上不共享参数会提高拟合能力，测试集精度反而下降，说明出现了轻微过拟合。该实验结果表明，我们采用参数共享的双向自注意力不仅减少了参数数量，也缓解了过拟合。

Table 8. 使用不共享参数的双向自注意力时精度和参数量的变化。

数据集	参数量 (MB)	精度 (%)
DVS Gesture	0.65 (+25%)	96.63 (-0.94)
ASL-DVS	0.34 (+26%)	99.86 (-0.05)
DVS-Lip	22.90 (+25%)	75.36 (-0.52)

A.5. 自监督训练细节

event2vec 表示具有逐事件的性质，因此很适合进行自监督预训练，并可显著提升模型性能。具体而言，我们采用类似 BERT 的掩码建模方法。训练目标是遮蔽部分事件的空间坐标和极性 (x, y, p)，并训练模型基

于周围事件及其相关时间信息提供的上下文来预测被遮蔽的值。该任务促使模型学习对时空事件模式的有意义理解。

该自监督训练框架类似于 BERT (Devlin et al., 2019) 中的掩码语言模型 (MLM) 目标。给定一批形状为 (B, L, D) 的空间嵌入张量 $\mathbf{v}_s$，其中 B 为批大小，L 为序列长度，D 为嵌入维度，训练过程首先随机遮蔽一部分输入 token。

二值掩码 $\mathbf{m}$ 的形状为 (B, L)，通过先采样被遮蔽 span 的起始位置生成。对于 DVS-Lip 自监督配置，目标掩码比例为 30%，span 长度参数为 $l_{mask} = 10$；因此，每个 token 被选为 span 起点的概率为 $0.3/10$。为防止模型通过简单插值进行预测，每个被选中的起始位置都会遮蔽一段连续 token。每个被遮蔽 span 的长度采样为 Geometric$(p) + 1$，其中 $p = 0.1$，随后截断到最大长度 20。与掩码项 $\mathbf{m}[i][j] = 1$ 对应的每个空间 token $\mathbf{v}_s[i][j]$ 都被替换为单个可学习的 D 维掩码 token $\mathbf{v}_m$。该操作得到损坏后的空间嵌入张量，随后将其与时间嵌入和强度因子融合，形成损坏后的 event2vec 张量 $\hat{\mathbf{v}}$。同时，保留被遮蔽 token 的原始空间坐标和极性 $(\mathbf{x}_m, \mathbf{y}_m, \mathbf{p}_m)$，作为重建损失的真实值。

损坏后的张量 $\hat{\mathbf{v}}$ 随后由模型的 FoX 骨干网络处理。之后，使用掩码 $\mathbf{m}$ 从最终输出张量中提取与初始被遮蔽位置对应的输出嵌入，记为 $\hat{\mathbf{v}}_m$。

目标是让模型从这些损坏嵌入中重建原始空间和极性信息。为此，我们首先应用时空融合操作的逆过程，以分离重建嵌入中的空间分量：

$$\hat{\mathbf{v}}_s = \frac{\hat{\mathbf{v}}_m}{\log(\boldsymbol{\rho}_m) + 1} - \mathbf{v}_{t,m}. \tag{11}$$

其中 $\boldsymbol{\rho}_m$ 和 $\mathbf{v}_{t,m}$ 分别表示被遮蔽位置处的强度因子和时间嵌入。所得张量 $\hat{\mathbf{v}}_s$ 被视为重建后的空间嵌入。随后将其输入到一个与空间嵌入编码器结构相对应的解码器网络中，以预测原始极性和坐标 $(\hat{\mathbf{p}}, \hat{\mathbf{y}}, \hat{\mathbf{x}})$。具体而言，该解码器由一组线性层、层归一化和 ReLU 激活函数组成。该网络被设计为将特征维度从 D 逐步降至 3。解码器输出采用 tanh 激活函数，将预测值限制在 $(-1, 1)$ 范围内。这与输入预处理保持一致，因为真实极性和坐标也被归一化到相同范围。

最后，训练目标是最小化预测值 $(\hat{\mathbf{p}}, \hat{\mathbf{y}}, \hat{\mathbf{x}})$ 与被遮蔽 token 真实值 $(\mathbf{p}_m, \mathbf{y}_m, \mathbf{x}_m)$ 之间的均方误差 (MSE) 损失。

A.6. 性能测试实验环境

本文中与性能测试相关的所有实验（如吞吐量和延迟）均在一台 Red Hat Enterprise Linux 8.10 服务器上进行。该服务器配备 NVIDIA A100 GPU (80GB PCIe)、Intel Xeon Gold 6326 CPU（使用 8 个核心）以及 256GB RAM。为减轻数据 I/O 的影响，在实验期间所有数据集均被完整加载到 RAM 中。

A.7. 事件数量对 DVS Gesture 的影响

如图 7 所示，我们评估了随机采样事件数量 (L) 对若干关键指标的影响，包括训练/推理吞吐量、单个事件流推理延迟，以及 DVS Gesture 数据集上的精度。ASL-DVS 数据集上的实验结果呈现相似趋势，因此此处不再展示。随着 L 增加，训练和推理吞吐量均急剧下降，而单流延迟几乎保持不变。这表明对于单个样本，主要延迟来自 CUDA kernel 启动开销，而非计算本身。同时，精度随 L 增长而快速提升，说明更多事件有助于模型进行决策。

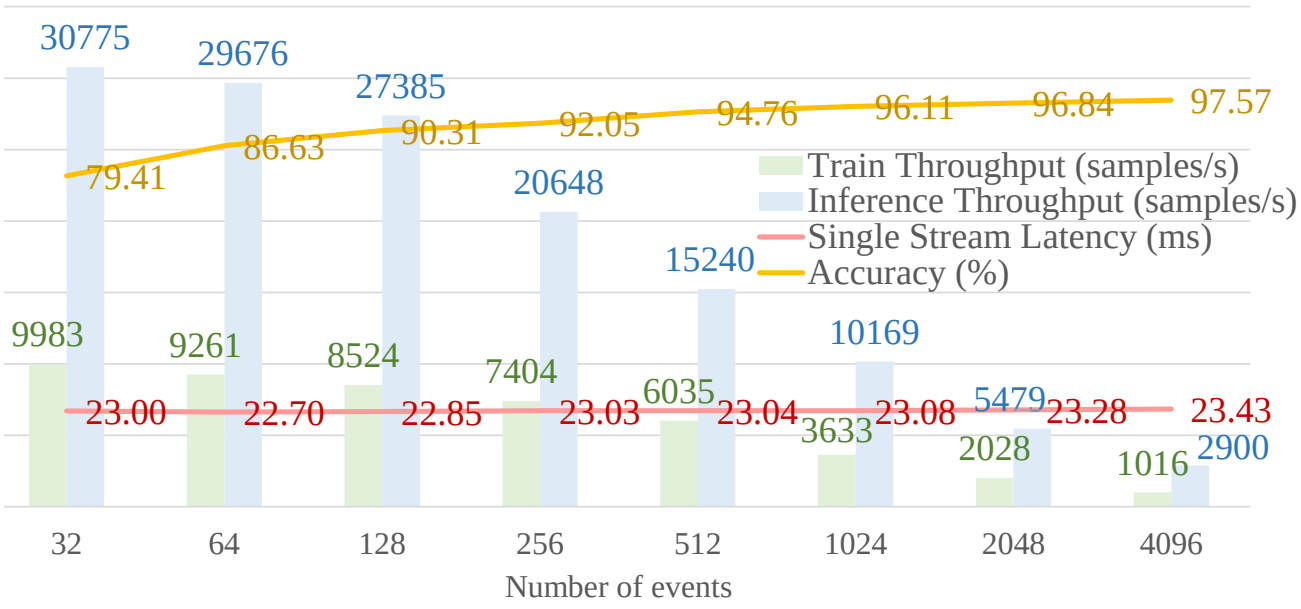

Figure 7. 事件数量对 DVS Gesture 的影响。

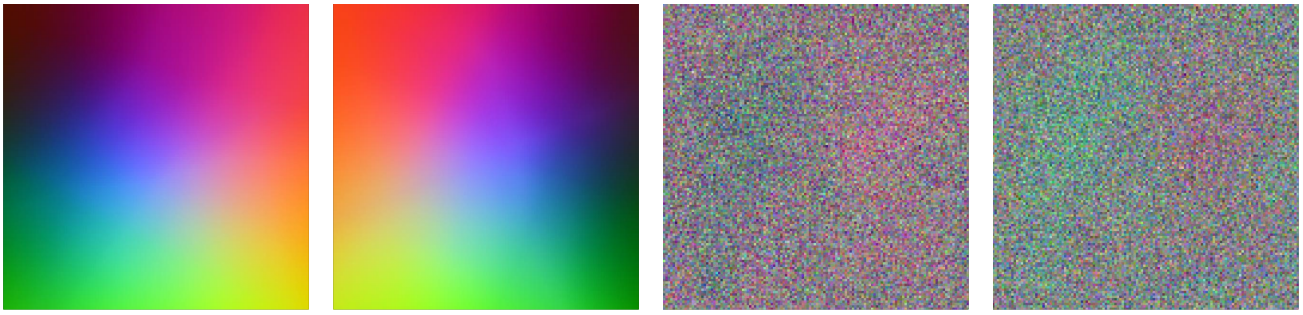

Figure 8. RGB 域中参数化嵌入权重 $\mathbf{W}_\phi$ 与标准嵌入权重 $\mathbf{W}_s$ 的可视化。

A.8. 邻域语义可视化

由于正文篇幅有限，图 5(a) 和图 5(c) 仅展示了单一事件极性的可视化结果。为保持完整性，本节提供包含两种极性的补充可视化结果。图 8 展示了映射到 RGB 颜色空间的嵌入权重，图 9 则将其表示为向量场。

A.9. 数据增强

Transformer（包括线性注意力）缺乏归纳偏置，因此需要更多数据进行学习。我们使用数据增强方法在一定程度上扩充数据量，从而提升性能。具体而言，我们没有在 ASL-DVS 数据集上使用数据增强，因为我们发现即使不使用数据增强也能达到当前最佳 (SOTA) 性能；这可能是由于该数据集规模充足：其训练集样本数约为 80,640，而 DVS Gesture 为 1,176，DVS-Lip 为 14,896。

记 $\mathcal{U}(a, b)$ 为 a 和 b 之间的均匀分布，$\mathrm{RandInt}(m, n)$ 为从集合 $\{m, m+1, \ldots, n\}$ 中采样得到的随机整数，其中每个整数被选中的概率相同。

对于事件流，数据增强直接作用于事件。为简洁起见，本小节省略事件索引。当启用随机变换时，除非另

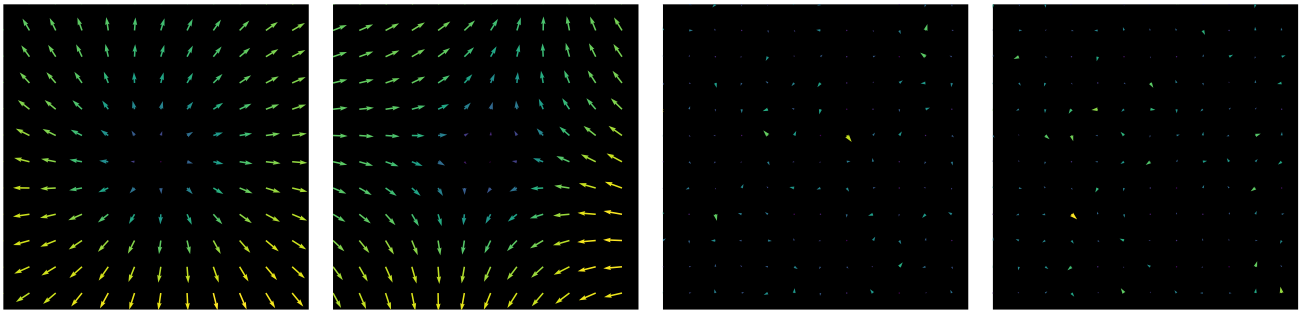

Figure 9. 以向量场形式可视化参数化嵌入权重 $\mathbf{W}_{\phi}$ 与标准嵌入权重 $\mathbf{W}_{s}$。

有说明，其随机参数会针对批中的每个事件流分别采样。需要注意的是，在应用任何增强之前，坐标会被转换为浮点精度。在完成所有增强后，仅保留坐标有效的事件，即满足 $x \in [0, W-1], y \in [0, H-1]$ 的事件。

对于 DVS Gesture 上的分类任务，随机应用策略以 0.6 的概率独立启用以下每种变换：

- **随机缩放**：坐标 (x, y) 被缩放为 $(s_x \cdot x, s_y \cdot y)$，缩放因子满足 $s_x, s_y \sim \mathcal{U}(0.8, 1.2)$。
- **随机旋转**：坐标旋转角度 $r \sim \mathcal{U}(-10, 10)$ 度。
- **随机错切**：应用错切变换，错切因子满足 $\lambda_x, \lambda_y \sim \mathcal{U}(-0.02, 0.02)$。
- **随机平移**：坐标按偏移量 $d_x, d_y \sim \mathcal{U}(-16, 16)$ 平移。
- **随机擦除**：以 0.1 的概率擦除一个 $h \times w$ 区域，其中 $h, w \sim \mathcal{U}(0, 16)$。该区域中心 (c_x, c_y) 满足 $c_x \sim \mathcal{U}(0, W-1), c_y \sim \mathcal{U}(0, H-1)$。
- **时间块丢弃**：从事件流中移除八个候选时间块。令 L_{valid} 为有效事件数量。每个被移除块的长度采样为 $l_{chunk} = \text{RandInt}(1, 256) \cdot L_{valid}/L$，其起始位置从 token 索引中均匀采样。

在 DVS-Lip 分类模型的自监督阶段，我们采用一系列几何变换。随机应用策略以 0.5 的概率独立启用每个列出的变换：

- **随机缩放**：坐标 (x, y) 被缩放为 $(s_x \cdot x, s_y \cdot y)$，缩放因子满足 $s_x, s_y \sim \mathcal{U}(0.8, 1.2)$。
- **随机旋转**：坐标旋转角度 $r \sim \mathcal{U}(-15, 15)$ 度。
- **随机错切**：应用错切变换，错切因子满足 $\lambda_x, \lambda_y \sim \mathcal{U}(-0.05, 0.05)$。
- **水平翻转**：事件流以内部概率 0.5 进行水平翻转。
- **随机平移**：坐标按偏移量 $d_x, d_y \sim \mathcal{U}(-16, 16)$ 平移。

训练 DVS-Lip 分类模型时，我们使用以下数据增强：

- **随机缩放**：坐标 (x, y) 被缩放为 $(s_x \cdot x, s_y \cdot y)$，缩放因子满足 $s_x, s_y \sim \mathcal{U}(0.8, 1.2)$。

Table 9. 表 1 中各方法的数据增强策略。

数据集	方法 + 表示	数据增强
DVS Gesture	Sparse GRU + Frame (Subramoney et al., 2023)	随机裁剪、平移和旋转
	SNN + Frame (Yao et al., 2023)	随机切片并积分
	FARSE-CNN + Window Slicing (Santambrogio et al., 2024)	随机坐标平移
	Event MAE + Point Cloud (Sun et al., 2025)	来自 Point-BERT (Yu et al., 2022) 的点重采样
	Max-Former + Frame (Fang et al., 2025)	Mixup 和 Cutmix
	Transformer + Event2Vec	随机缩放、旋转、错切、平移、擦除和时间块丢弃
ASL-DVS	GNN, CNN + Graph (Bi et al., 2019)	节点位置的随机缩放、翻转和旋转
	GNN & Transformer + Image & Voxel Graph (Yuan et al., 2023)	随机缩放和平移
	Transformer + Event2Vec	无
DVS-Lip	ResNet-18 & BiGRU + Frame (Tan et al., 2022)	随机裁剪和水平翻转
	Spiking ResNet18 & BiGRU + Frame (Dampfhoffer & Mesquida, 2024)	随机裁剪、水平翻转、空间掩码、缩放和时间掩码
	Transformer + Event2Vec	随机缩放、旋转、错切、翻转、平移、擦除和时间块丢弃

- **随机旋转**：坐标旋转角度 $r \sim \mathcal{U}(-15, 15)$ 度。
- **随机错切**：对 x 和 y 应用错切变换，错切因子满足 $\lambda_x, \lambda_y \sim \mathcal{U}(-0.05, 0.05)$。
- **随机翻转**：事件流以概率 1 进行水平翻转。
- **随机平移**：坐标 x 和 y 按偏移量 $d_x, d_y \sim \mathcal{U}(-16, 16)$ 平移。
- **随机擦除**：以 0.1 的概率擦除一个 $h \times w$ 区域，其中 $h, w \sim \mathcal{U}(0, 16)$。该区域中心 (c_x, c_y) 满足 $c_x \sim \mathcal{U}(0, W-1), c_y \sim \mathcal{U}(0, H-1)$。
- **随机时间块丢弃**：从事件流中移除四个候选时间块。令 L_{valid} 为有效事件数量。每个被移除块的长度采样为 $l_{chunk} = \text{RandInt}(1, 128) \cdot L_{valid}/L$，其起始位置从 token 索引中均匀采样。

随机应用策略以 0.5 的概率独立启用上述每种增强。TokenMix 以 0.5 的概率应用于嵌入张量。具体而言，在聚类事件上训练时，强度 ρ 以 0.1 的概率被随机设为 1。我们在 FoX 骨干网络中使用 drop path (Larsson et al., 2016)，其概率随深度从 0 线性增加到 0.4。

此外，表 1 中的其他方法也使用了数据增强，其汇总见表 9。