Efficient 3D Recognition with Event-driven Spike Sparse Convolution

Xuerui Qiu[1,2], Man Yao[1*], Jieyuan Zhang[4], Yuhong Chou[1,5], Ning Qiao[6], Shibo Zhou[7], Bo Xu[1], Guoqi Li[1,3,8*]

[1]Institute of Automation, Chinese Academy of Sciences
[2] School of Future Technology, University of Chinese Academy of Sciences
[3]Peng Cheng Laboratory
[4]University of Electronic Science and Technology of China
[5]The Hong Kong Polytechnic University
[6]SynSense AG Corporation
[7]Huinao Zhixin
[8] Institute of Automation, Key Laboratory of Brain Cognition and Brain-inspired Intelligence Technology, Chinese Academy of Sciences
{qiuxuerui2024, guoqi.li}@ia.ac.cn

Abstract

Spiking Neural Networks (SNNs) provide an energy-efficient way to extract 3D spatio-temporal features. Point clouds are sparse 3D spatial data, which suggests that SNNs should be well-suited for processing them. However, when applying SNNs to point clouds, they often exhibit limited performance and fewer application scenarios. We attribute this to inappropriate preprocessing and feature extraction methods. To address this issue, we first introduce the Spike Voxel Coding (SVC) scheme, which encodes the 3D point clouds into a sparse spike train space, reducing the storage requirements and saving time on point cloud preprocessing. Then, we propose a Spike Sparse Convolution (SSC) model for efficiently extracting 3D sparse point cloud features. Combining SVC and SSC, we design an efficient 3D SNN backbone (E-3DSNN), which is friendly with neuromorphic hardware. For instance, SSC can be implemented on neuromorphic chips with only minor modifications to the addressing function of vanilla spike convolution. Experiments on ModelNet40, KITTI, and Semantic KITTI datasets demonstrate that E-3DSNN achieves state-of-the-art (SOTA) results with remarkable efficiency. Notably, our E-3DSNN (1.87M) obtained 91.7% top-1 accuracy on ModelNet40, surpassing the current best SNN baselines (14.3M) by 3.0%. To our best knowledge, it is the first direct training 3D SNN backbone that can simultaneously handle various 3D computer vision tasks (e.g., classification, detection, and segmentation) with an event-driven nature.

Code — https://github.com/bollossom/E-3DSNN/

Introduction

3D recognition has been a highly researched area due to its wide applications in autonomous driving (Cui et al. 2021), virtual reality (Zhu et al. 2024), and robotics (Pomerleau et al. 2015). However, these methods involve numerous operations, leading to high computational costs and energy consumption, which limits their deployment on resource-constrained devices. Bio-inspired Spiking Neural Networks (SNNs) provide an energy-efficient way to extract features from 3D event streams due to their unique event-driven nature and spatio-temporal dynamics (Maass 1997; Roy et al. 2019; Li et al. 2023). For instance, the Speck (Yao et al. 2024c) chip uses event-by-event sparse processing to handle event streams, with operational power consumption as low as 0.7 mW. Point clouds and event streams are both sparse 3D data, which theoretically suggests that SNNs should be well-suited for processing 3D sparse point clouds. However, when applying SNNs to point clouds, they often exhibit limited performance and fewer application scenarios in most cases.

For instance, most applications of SNN algorithms (Lan et al. 2023; Wu et al. 2024a; Ren et al. 2024) in 3D recognition are limited to simple 3D classification tasks with toy model datasets (Wu et al. 2015) and the performance gap between these works and ANNs is significant. To better apply the SNNs in the efficient 3D recognition field, we identify the key issues in the SNN processing of point clouds. The first is the appropriate preprocessing method. Vanilla methods (Ren et al. 2024; Wu et al. 2024b) use point-based methods to process input point clouds, the inherent sparsity of SNNs can obscure local geometric information, and the high computational load makes training on large datasets time-consuming. The second is selecting efficient feature extraction tools. 3D data itself has high redundancy. While SNNs use 2D spike convolution effectively for event streams, applying it to 3D sparse point clouds results in cubic growth in computational complexity as it calculates each point. This makes feature extraction inefficient and challenging.

In this work, we aim to address the above issues in a unified manner. Our goal is to highlight the low power consumption and distinct sparse event-driven advantages of SNNs. We accomplish this through two main approaches. First, we propose a Spike Voxel Coding (SVC) scheme for processing point clouds. As shown in Fig. 2, SVC can encode the 3D point clouds into sparse and spatio-temporal spike trains, reducing the storage requirements and saving time on point cloud preprocessing. Second, we propose a Spike Sparse Convolution (SSC) block for extracting 3D point cloud features, which leverages sparsity to reduce redundant computations

*Corresponding author.
Copyright © 2025, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

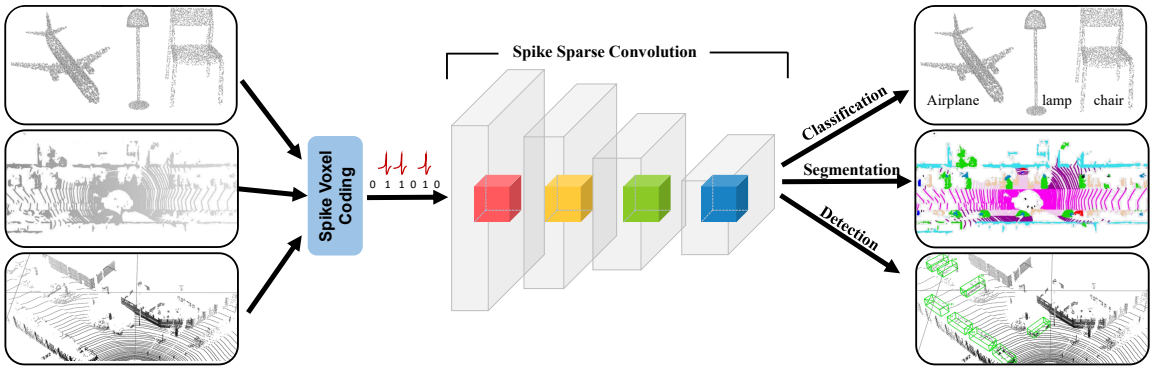

Figure 1: The workflow of our efficient 3D SNN backbone (E-3DSNN), which uses residual connections between membrane potentials and handles various 3D computer vision tasks with only sparse ACcumulate. It consists of two main components: the Spike Voxel Coding (SSC) and Spike Sparse Convolution (SSC). The SVC scheme first voxelizes the input 3D points. Then, the voxelized data is transformed into spatio-temporal spike trains using sparse convolution and spiking neurons. The SSC block only calculates the overlapping activation features between the center of the point cloud and the convolution kernel.

on background points and avoids the densification issues associated with vanilla spike convolution. As shown in Fig. 3, SSC adds just one extra condition compared to the Vanilla Spike Convolution (VSC). Because of their similarities, SSC can be implemented on neuromorphic chips with only minor modifications to the VSC addressing function. Finally, leveraging SVC and SSC, we redesigned an efficient 3D SNN backbone (E-3DSNN) using residual connections between membrane potentials, as shown in Fig. 1. To demonstrate the effectiveness of E-3DSNN, we evaluate our models on the simple ModelNet40 (Wu et al. 2015) and two large-scale benchmarks (e.g., KITTI (Geiger et al. 2012a) and Semantic KITTI (Behley et al. 2019) datasets). E-3DSNN achieves leading performance with high efficiency with only sparse ACcumulate (AC) in SNNs. Our main contribution can be summarized as:

- We introduce the SVC scheme and SSC block, enhancing SNN efficiency and performance in processing 3D point clouds. SVC converts point clouds to sparse spike trains, while SSC extracts effective representations from them.
- We explore suitable architectures in SNNs and redesigned LiDAR-based 3D SNN backbone by residual connections between membrane potentials, handling various 3D computer vision tasks with sparse AC operation.
- Experiments demonstrate that our E-3DSNN achieves outstanding accuracy with remarkable efficiency up to 11$\times$ on various datasets (e.g., ModelNet40, KITTI, and semantic KITTI), revealing the potential of SNNs in efficient 3D recognition.

Related Works

SNN Training and Architecture Design

The development of SNNs has long been hindered by the challenge of training non-differentiable binary spikes. To address this, researchers have focused on improving training methods and architectural designs. Recently, two primary methods for training high-performance SNNs have emerged. One approach is to convert ANNs into spike form through neuron equivalence (Li et al. 2021), known as ANN-to-SNN conversion. However, this method requires long simulation time steps and increases energy consumption. We employ the direct training method (Wu et al. 2018).

Regarding architectural design, Spiking ResNet (Fang et al. 2021; Shan et al. 2023) has long dominated the SNN field because residual learning (He et al. 2016a) can address the performance degradation of SNNs as they deepen. The main differences among these are the locations of shortcuts and their ability to achieve identity mapping (He et al. 2016b). Notably, MS-ResNet (Hu et al. 2024b; Qiu et al. 2024) maintains high performance while preserving the spike-driven sparse addition nature of SNNs by establishing residual connections between membrane potentials. Our E-3DSNN design draws on this idea and extends it to the 3D scene.

Feature Extractors on LiDAR-based 3D Recognition

The key challenge in LiDAR-based 3D recognition is learning effective representations from sparse and irregular 3D geometric data. Currently, there exist two main approaches. Point-based methods (Qi et al. 2017; Zhao et al. 2021) utilize the PointNet series to directly extract geometric features from raw point clouds and make predictions. However, these methods require computationally intensive point sampling and neighbor search procedures. Additionally, in 3D scenes, numerous background points unrelated to the task contribute to redundant computations at each stage. Voxel-based methods (Wu et al. 2015; Choy et al. 2019; Zhou et al. 2018) first convert the point cloud into regular voxels and then use 3D sparse convolutions for feature extraction. Due to its efficiency advantages, this approach has been widely applied to various 3D tasks. Nevertheless, the improved accuracy is often accompanied by increased computational costs, limiting its applicability in practical systems. However, as voxel resolution increases, both computational costs and memory requirements grow cubically.

Numerous studies (Lan et al. 2023; Ren et al. 2024; Wu et al. 2024b) in the SNN field combine spiking neurons

with Point-based methods like PointNet. These methods have been successfully applied to simple datasets with shallow networks, but achieving high performance becomes more challenging as datasets and networks become larger and more complex, which restricts SNNs' application in 3D recognition. This is due to their oversight of SNNs' inherent sparsity, which can obscure local geometric information, and the high computational load of point-based methods, resulting in lengthy training times on large datasets. We adopt a voxel-based approach for point cloud processing and leverage the sparse nature of spiking neurons to reduce unnecessary computation costs caused by 3D spatial redundancy.

Efficient 3D Recognition SNN Backbone

In this section, we begin by briefly introducing the spike neuron layer, followed by the Spike Voxel Coding (SVC) and the Spike Sparse Convolution (SSC). Finally, we introduce our general efficient 3D recognition SNN backbone (E-3DSNN).

Leaky Integrate-and-Fire Spiking Neuron

The Leaky Integrate-and-Fire (LIF) spiking neuron is the most popular neuron to balance bio-plausibility and computing complexity (Maass 1997). We first translate the LIF spiking neuron to an iterative expression with the Euler method (Wu et al. 2018), which can be described as:

$$U^{t,n} = H^{t-1,n} + f(W^n, X^{t,n-1}), \tag{1}$$

$$S^{t,n} = \Theta(U^{t,n} - V_{th}), \tag{2}$$

$$H^{t,n} = \beta(U^{t,n} - S^{t,n}), \tag{3}$$

where β is the time constant, t and n respectively represent the indices of the time step and the n-th layer, W denotes synaptic weight matrix between two adjacent layers, $f(\cdot)$ is the function operation stands for convolution or fully connected layer, X is the input, and $\Theta(\cdot)$ is the Heaviside function. When the membrane potential U exceeds the firing threshold V_{th}, the LIF neuron will trigger a spike S.

However, converting the membrane potential of spiking neurons into binary spikes introduces inherent quantization errors, which significantly limit the model's expressiveness. To reduce the quantization error, we incorporate the Integer LIF (I-LIF) neuron (Luo et al. 2024) into our E-3DSNN, allowing us to rewrite Eq. (2) as:

$$S^{t,n} = \lfloor \mathrm{clip}\{U^{t,n}, 0, D\} \rceil, \tag{4}$$

where $\lfloor \cdot \rceil$ denotes the rounding operator, $\mathrm{clip}\{x, a, b\}$ confines x within range $[a, b]$, and D is a hyperparameter indicating the maximum emitted integer value by I-LIF. In the backpropagation stage, the Eq. (2) is non-differentiable. Previous studies have introduced various surrogate gradient functions (Wu et al. 2018; Neftci, Mostafa, and Zenke 2019), primarily designed to address binary spike outputs. In our approach, we consistently utilize rectangular windows as the surrogate function. For simplicity, we retain gradients only for neurons activated within the $[0, D]$ range, setting all others to zero. Moreover, I-LIF will emit integer values while training, and convert them into 0/1 spikes by expanding the virtual timestep to ensure that the inference is spike-driven with only sparse addition.

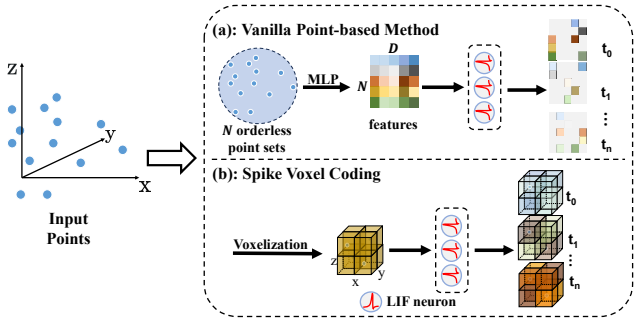

Figure 2: Comparison of different point cloud pre-processing ways in SNN. (a) The vanilla point-based method (Lan et al. 2023; Ren et al. 2024; Wu et al. 2024a) directly processes raw points, but the inherent sparsity of SNNs can obscure local geometric details. (b) We proposed a spike voxel coding (SVC) scheme, which leverages the sparsity of SNNs and, after additional voxelization pretreatment, can handle structural data with higher efficiency and lower power consumption.

Spike Voxel Coding

In this section, we proposed a Spike Voxel Coding (SVC) scheme for efficiently transforming point clouds into spatio-temporal voxelized spike trains (Qiu et al. 2023). The overall process of SVC processing a 3D point cloud is as follows.

First, consider the input is a 3D point set with sparse voxelized 3D scene representation $\mathcal{V} = \{\mathcal{P}, \mathcal{I}\}$. It contains voxels sets $V_k^t = \{P_k^t, I_k^t\}$, where $P_k^t \in \mathbb{R}^3$ represents the 3D coordinates and $I_k^t \in \mathbb{R}^d$ is the corresponding feature with d channels at timestep t. Next, we divide the global voxel set $\mathcal{V}$ into N non-overlapping voxel grids $[\mathcal{V}_1^t, \mathcal{V}_2^t, \ldots, \mathcal{V}_N^t]$, $\mathcal{V}_i^t = \{V_j^t \mid P_j^t \in \Phi^t(i)\}$, where $\mathcal{V}_i^t$ represents the i-th voxel grid and $\Phi^t(i)$ means the index range of the i-th voxel grid at timestep t. Then we encode these voxel grids into spike trains, which can be expressed as:

$$\mathcal{S} = \mathcal{SN}^m(\mathcal{F}^m(\mathcal{V})), \tag{5}$$

where $\mathcal{SN}^m$ and $\mathcal{F}(\cdot)^m$ is m consecutive I-LIF spiking neuron and sparse convolution, respectively. And $\mathcal{S} = [\mathcal{S}_1^t, \mathcal{S}_2^t, \ldots, \mathcal{S}_N^t]$ is a set of output spike trains. After our SVC, we obtain a sparse spiking representation $\mathcal{S}$ of the input 3D point cloud, which can reduce the storage requirements.

Spike Sparse Convolution

Vanilla Spike Convolution (VSC) is performed on neuromorphic chips in a spike-driven manner. Then we will introduce how the VSC extracts 3D features. Consider weight W^t contains $c_{\text{in}} \times c_{\text{out}}$ spatial kernels K and S_p^t as an input feature with t timestep at position p , VSC can be expressed by:

$$U^t = \sum_{k \in K^3} W_k \cdot S_{\vec{p}_k}^t, \tag{6}$$

Here U^t is the output membrane potential and $\vec{p}_k$ is the position offset around center p, which can be expressed by:

$$\vec{p}_k = p + k = (p_x + k_x, p_y + k_y, p_z + k_z), \tag{7}$$

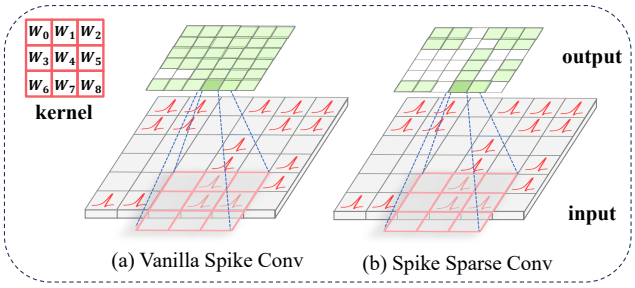

Figure 3: Comparison of Spike Sparse Conv (SSC) and Vanilla Spike Conv (VSC). Inputs and outputs are shown as 2D features for simplicity: green for activation, red for spikes, and white for no activation. On a neuromorphic chip, when a spike occurs, the address mapping function finds the synapses and neurons that need to be added and then takes out the corresponding weights to perform the addition operations. The only difference between VSC and SSC is the addressing mapping function. In SSC, it is specified that the convolution is performed only if there is a spike input at the position corresponding to W_4 (the center position of the convolution kernel). VSC does not have this restriction.

where k represents the kernel offset that enumerates all the discrete positions within the 3D kernel space K^3.

Spike Sparse Convolution (SSC) VSC performs well in 2D scenes. However, in the 3D sparse point cloud, it needs to calculate each point and the computational complexity grows cubically when processing the point cloud, making it difficult to extract features efficiently. To address this issue, we propose Spike Sparse Convolution (SSC), which performs only on the key spiking locations, significantly reducing the computational requirements. It can be expressed by:

$$U_p^t = \sum_{k \in K^d} \alpha(W_k \cdot S_{\vec{p}_k}^t), \tag{8}$$

where $\alpha \in \{0, 1\}$ is a selector. When the center $p \in S^t$ is the active binary spike, α equals 1, indicating that the position p participates in the computation. $\alpha = 0$ is the opposite. As depicted in Fig. 3, SSC only has one more judgment condition than VSC when performing spike convolution. Given the commonality of VSC and SSC, we only need to slightly modify the addressing mapping function corresponding to VSC to execute SSC on the neuromorphic chip.

The specific process is as follows. Upon the reception of a spike, the SNN core first builds a rulebook, which records the activation spikes and the corresponding kernels space $K^3(p, P_{\text{in}})$. The kernel space is a subset of K^3, leaving out the empty position. It is conditioned on the position p and input feature space P_{in} as:

$$K^3(p, P_{\text{in}}) = \left\{k | p + k \in P_{\text{in}}, k \in K^3\right\}. \tag{9}$$

Then the rulebook searches for and identifies the corresponding synaptic weights and the positions of the target neurons, and adds them together.

Overall Architecture

Fig. 1 shows the overview of our hieratical 3D Computer Vision SNN Backbone (E-3DSNN). Drawing inspiration of MS-ResNet (Hu et al. 2024b), we establish residual connections between the membrane potentials of spiking neurons. This avoids the common spike degradation issue (Yao et al. 2023) in SNNs and ensures that the network remains spike-driven during inference (Yao et al. 2024a,b). Considering the input is a 3D point set with a sparse voxelized 3D scene representation $\mathcal{V}$, our E-3DSNN can be formulated as follows:

$$\mathcal{S}^{t,0} = \mathcal{C}(\mathcal{V}), \tag{10}$$

$$U^{t,l} = \mathcal{B}^l(\text{Down}^l\{\mathcal{S}^{t,0}\}), \tag{11}$$

$$U^{t,l+1} = \mathcal{B}^{l+1}(\text{Down}^{l+1}\{U^{t,l}\}), \tag{12}$$

where $\mathcal{C}(\cdot)$ denotes spike voxel coding, and $l = 1, \cdots, L$ represents the layer number, with L equal to 4 in our study. Down$(\cdot)$ is the downsample layer, which consists of a spiking neuron and a spike sparse convolution. Both the kernel size and stride are set to 2, reducing the spatial size to $\frac{1}{8}$ with each operation. $\mathcal{B}(\cdot)$ is the basic spike sparse block. Considering the input of the basic spiking sparse block is U, this block can be expressed as:

$$U' = \text{SSC}\{\mathcal{SN}(U)\} + U, \tag{13}$$

$$U'' = \text{SSC}^m\{\mathcal{SN}^m(U')\}, \tag{14}$$

where SSC^m and $\mathcal{SN}^m$ indicate m consecutive spike sparse convolution and spiking neurons, which is set to 2 in our study. The kernel size of SSC and stride are set to 3 and 1, respectively.

Theoretical Energy Consumption

The 3DSNN architecture can transform matrix multiplication into sparse addition, which can be implemented as an addressable addition on neuromorphic chips. In the spike voxel coding layer, convolution operations serve as MAC operations that convert analog inputs into spikes, similar to direct coding-based SNNs (Wu et al. 2019). Conversely, in SNN's architecture, the Conv or FC layer transmits spikes and performs AC operations to accumulate weights for post-synaptic neurons. Additionally, the inference energy cost of E-3DSNN can be expressed as follows:

$$E_{total} = E_{MAC} \cdot FL_{conv}^1 + E_{AC} \cdot T \sum_{n=2}^{N} FL_{conv}^n \cdot fr^n, \tag{15}$$

where N and M are the total number of spike sparse conv, E_{MAC} and E_{AC} are the energy costs of MAC and AC operations, and fr^m, fr^n, FL_{conv}^n and FL_{fc}^m are the firing rate and FLOPs of the n-th spike sparse conv. Previous SNN works (Horowitz 2014; Rathi and Roy 2021) assume 32-bit floating-point implementation in 45nm technology, where E_{MAC} = 4.6pJ and E_{AC} = 0.9pJ for various operations.

Experiments

In this section, we first give the hyper-parameters setting. Then we validate the E-3DSNN on diverse vision tasks, including

Architecture	Method	Input	$T \times D$	Param (M)	Power (mJ)	Accuracy (%)
ANN	PointNet (Qi et al. 2017)*CVPR*	Point	N/A	3.47	0.14	89.2
	KPConv (Thomas et al. 2019)*CVPR*	Point	N/A	14.3	-	92.9
	Pointformer (Zhao et al. 2021)*ICCV*	Point	N/A	4.91	0.11	93.7
	3DShapeNets (Wu et al. 2015)*CVPR*	Voxel	N/A	6.92	0.15	77.3
	3DVGG-B (Graham et al. 2017)	Voxel	N/A	5.23	0.12	88.2
	E-3DSNN*	Voxel	N/A	3.27	0.17	90.9
SNN	SpikePointNet (Ren et al. 2024)*NeurIPS*	Point	4×1	3.47	0.03	88.2
	SpikingPointNet (Lan et al. 2023)*ICCV*	Point	16×1	3.47	0.13	88.6
	P2SResLNet (Wu et al. 2024b)*AAAI*	Point	4×1	14.3	-	88.7
	E-3DSNN (Ours)	Voxel	1×4	**1.87**	**0.01**	91.5
		Voxel	1×4	3.27	0.02	**91.7**

Table 1: Shape classification results on the ModelNet40 dataset (Wu et al. 2015). Power is the estimation of energy consumption same as (Hu et al. 2024a; Shan et al. 2024). * We convert 3.27M of E-3DSNN into ANN with the same architecture.

3D classification, object detection, and semantic segmentation. Next, we ablate the different blocks of E-3DSNN to prove the effectiveness of our method. For further detailed information on architecture, more experiments on the NuScenes (Caesar et al. 2020) datasets, and visualizations, refer to the **Appendix**.

Hyper-parameters Setting

In this section, we give the specific hyperparameters of our training settings in all experiments, as depicted in Tab. 2. In this work, we train our E-3DSNN with 4 A100 GPUs.

Parameter	ModelNet40	KITTI	SemanticKITTI
Learning Rate	$1e-1$	$1e-2$	$2e-3$
Weight Decay	$1e-4$	$1e-2$	$5e-3$
Batch Size	16	64	96
Training Epochs	200	80	100
Optimizer	SGD	Adam	AdamW

Table 2: Hyper-parameter training settings of 3DSNN.

3D Classification

The ModelNet40 (Wu et al. 2015) dataset contains 12,311 CAD models with 40 object categories. They are split into 9,843 models for training and 2,468 for testing. For the input data, we clip the point clouds to ranges of $[-0.2\text{m}, 0.2\text{m}]$ for the X-axis, $[-0.2\text{m}, 0.2\text{m}]$ for the Y-axis, and $[-0.2\text{m}, 0.2\text{m}]$ for the Z-axis. The input voxel size is set to 0.01m. In terms of the evaluation metrics, we use the point cloud classification overall accuracy.

As shown in Tab. 1, we compare our method with the previous state-of-the-art ANN and SNN domain. Notably, with only 3.27M parameters, the E-3DSNN achieves the best accuracy of 91.7%, regardless of voxel or point input in the SNN domain, showcasing significant advantages in both accuracy and efficiency. Specifically, E-3DSNN (This work) vs. SpikePointNet vs. SpikingPointNet: Power 0.01mJ vs. 0.03mJ vs. 0.13mJ; Param 1.87M vs. 3.47M vs. 3.47M; Accuracy 88.2% vs. 88.6% vs. 91.5%. That is, our model has +2.8% higher accuracy than SpikingPointNet (Lan et al. 2023) with only the previous 11.4% energy consumption. Moreover, the performance gap between SNNs and ANNs is significantly narrowed. For instance, under lower parameters, the performance of Pointformer (Zhao et al. 2021) and E-3DSNN are comparable, and the energy efficiency is $11\times$.

3D Object Detection

The large KITTI dataset (Geiger et al. 2012b) consists of 7481 training samples, which are divided into trainsets with 3717 samples and validation sets with 3769 samples. In detection, E-3DSNN are evaluated as backbones equipped with VoxelRCNN Head (Deng et al. 2021). We transform OpenPCDet (Team 2020) codebase into a spiking version and use it to execute our model. Raw point clouds are divided into regular voxels before being input to our 3DSNN on KITTI (Geiger et al. 2012a). For the input data, we clip the point clouds to the following ranges: $[0, 70.4]\text{m}$ for the X-axis, $[-40, 40]\text{m}$ for the Y-axis, and $[-3, 1]\text{m}$ for the Z-axis. The input voxel size is set to $(0.05\text{m}, 0.05\text{m}, 0.1\text{m})$. In terms of the evaluation metrics, we use the Average Precision (AP) calculated by 11 recall positions for the Car class.

As shown in Tab. 3, we compare our method in 3D object detection with the previous state-of-the-art (SOTA) ANN domain. Since no SNN has yet reported results on the KITTI dataset, we employ the I-LIF spiking neuron (Luo et al. 2024) to directly convert the PointRCNN (Shi et al. 2019) architecture into a spike-based version, referred to as Spike-PointRCNN. We obtained 89.6%, 84.0%, 78.7% AP, which is higher than the prior state-of-the-art SNN by a large margin, i.e., 5.8%, 11.9%, 6.8% absolute improvements on easy, moderate, and hard levels of class Car. E-3DSNN also has significant advantages over existing SNNs and ANNs regarding parameters and power. For instance, E-3DSNN (This work) vs. SpikePointNetRCNN vs. VoxelRCNN: AP 89.6% vs. 83.8% vs. 89.4%; Power 3.4mJ vs. 4.4mJ vs. 28.9mJ. In summary, E-3DSNN achieved state-of-the-art performance in the SNN domain in terms of both accuracy and efficiency on

Architecture	Method	Input	$T \times D$	Param (M)	Power (mJ)	Car 3D AP (R11) Easy	Mod.	Hard
ANN	PointRCNN (Shi et al. 2019)^{CVPR}	Point	N/A	4.0	22.5	88.8	78.6	77.3
	PVRCNN (Shi et al. 2020)^{CVPR}	Point	N/A	13.1	34.9	89.3	83.6	78.7
	Second (Yan et al. 2018)^{Sensor}	Voxel	N/A	5.3	23.9	88.6	78.6	77.2
	VoxelRCNN (Deng et al. 2021)^{AAAI}	Voxel	N/A	7.5	28.9	89.4	84.5	78.9
	GLENet (Zhang et al. 2023)^{IJCV}	Voxel	N/A	8.3	-	89.8	84.5	78.7
	E-3DSNN*	Voxel	N/A	8.5	31.2	89.4	83.7	78.2
SNN	SpikePointRCNN*	Point	1×4	4.0	4.4	83.8	72.1	71.9
	E-3DSNN (Ours)	Voxel	1×4	8.5	**3.4**	**89.6**	**84.0**	**78.7**

Table 3: 3D object detection results on the KITTI val benchmarks (Geiger et al. 2012a). * We convert 8.5M of E-3DSNN into ANN with the same architecture.

Architecture	Method	Input	$T \times D$	Param (M)	Power (mJ)	mIoU (%)
ANN	PointNet (Qi et al. 2017)^{CVPR}	Point	N/A	3.5	-	14.6
	Pointformer V3 (Wu et al. 2024b)^{CVPR}	Point	N/A	46.2	47.1	72.3
	SparseUNet (Graham et al. 2017)	Voxel	N/A	39.1	69.1	63.8
	SphereFormer (Lai et al. 2023)^{CVPR}	Voxel	N/A	32.3	49.2	67.8
	E-3DSNN*	Voxel	N/A	20.1	54.1	69.4
SNN	SpikePointNet*	Point	1×4	3.5	-	12.1
	SpikePointformer*	Point	1×4	46.2	13.1	67.2
	E-3DSNN (Ours)	Voxel	1×4	**17.9**	**4.5**	68.5
		Voxel	1×4	20.1	6.1	**69.2**

Table 4: 3D semantic segmentation results on Semantic KITTI val benchmarks (Behley et al. 2019). * We convert 20.1M of E-3DSNN into ANN with the same architecture.

the KITTI dataset, while also achieving results comparable to ANNs.

3D Semantic Segmentation

The large SemanticKITTI dataset (Behley et al. 2019) consists of sequences from the raw KITTI dataset, which contains 22 sequences in total. Each sequence includes around 1,000 lidar scans, corresponding to approximately 20,000 individual frames. We first transform Pointcept (Contributors 2023) codebase into a spiking version and use it to execute our model. Then we design an asymmetric encoder-decoder structure similar to UNet (Choy et al. 2019; Wu et al. 2023), with our E-3DSNN as encoder responsible for extracting multi-scale features and the decoder sequentially fusing the extracted multi-scale features with the help of skip connections. For voxelize implementation, the window size is set to $[120\text{m}, 2^\circ, 2^\circ]$ for (r, θ, ϕ). During data preprocessing, we restrict the input scene to the range $[-51.2\text{m}, -51.2\text{m}, -4\text{m}]$ to $[51.2\text{m}, 51.2\text{m}, 2.4\text{m}]$. The voxel size is set to 0.1m.

As shown in Tab. 4, we compare our method in 3D Semantic Segmentation with the previous state-of-the-art ANN domain. Since no SNN has yet reported results on the SemanticKITTI dataset, we employ the I-LIF spiking neuron (Luo et al. 2024) to convert the PointNet and Pointformer architectures into spike-based versions directly. We found that our E-3DSNN achieves the best mIoU of 69.2%, which is 2.0% and 1.6% higher than the previous SOTA SNN. Our E-3DSNN also demonstrates significant advantages over existing SNNs and ANNs in terms of parameter efficiency and power consumption for 3D semantic segmentation. For instance, E-3DSNN (This work) vs. SpikePoinformer vs. SphereFormer: mIoU 69.2% vs. 67.2% vs. 67.8%; Power 8.2mJ vs. 19.0mJ vs. 49.2mJ; Param: 20.1M vs. 46.2M vs. 32.3M.

Ablation Study

We first compared the results of ANN and SNN with the same architecture on the Semantic KITTI validation benchmarks. As shown in Tab. 5, while E-3DSNN's mIoU accuracy is 0.2% lower than the corresponding ANN, it shows an 8.8× improvement in energy efficiency. This indicates that SNNs have significant potential in efficiently processing sparse 3D point clouds.

Next, we ablate two components of our E-3DSNN, namely the SVC and SSC, to verify the effectiveness of the proposed method. As shown in Tab. 5, using SVC alone yields a slight decrease in mIoU by 0.3% but achieves a 1.8× improvement in energy efficiency. When both SVC and SSC are employed, there is a 0.2% reduction in mIoU, but energy efficiency improves by 8.8×. Therefore, the proposed SSC and SVC can significantly reduce power consumption and improve effi-

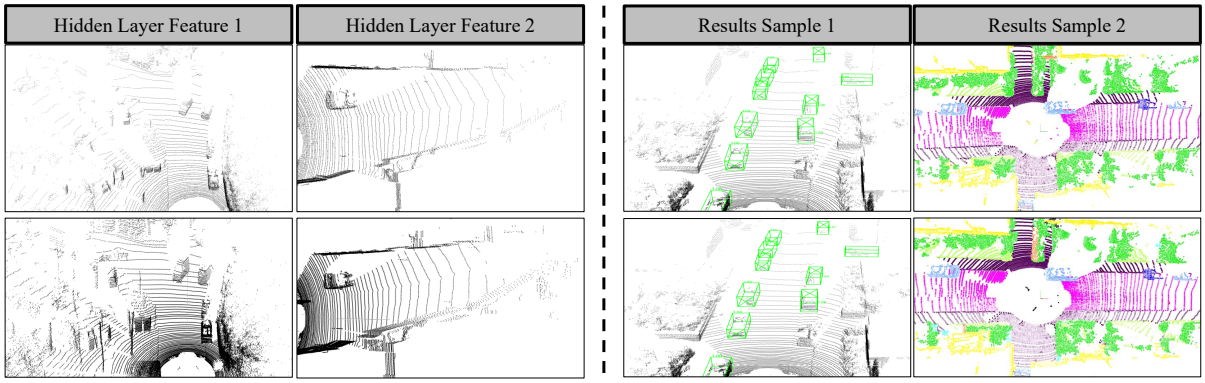

Figure 4: Visualization of E-3DSNN in hidden layer features and results. (a) We compared the hidden layer features generated with (top) and without SVC and SSC (bottom). (b) We compared the results of our E-3DSNN (top) in detection and segmentation with the ground truth (bottom).

ciency while maintaining high performance. Their combined use yields even more substantial improvements, highlighting their effectiveness in enhancing energy efficiency in 3D recognition tasks.

Then we evaluate the effects of varying T and D. We found that increasing the number of timesteps can enhance performance but affect inference time and increase power consumption. For instance, for the set with D = 1, extending T from 2 to 4 increases the mIoU by 1.4% but doubles the power consumption. Additionally, we found that expanding D while keeping T fixed improves performance and reduces power consumption. For instance, 2 × 1 vs. 2 × 2: mIoU, 67.1% vs. 68.9%; Power, 8.4mJ vs. 8.1mJ. 1 × 2 vs. 1 × 4: mIoU, 67.9% vs. 69.2%; Power, 6.3mJ vs. 6.1mJ.

Method	SVC	SSC	$T \times D$	Power (mJ)	mIoU (%)
ANN*	-	-	N/A	54.1	69.4
E-3DSNN	✓	-	1×4	29.1	69.3
	✓	✓	1×4	6.1	69.2
	✓	✓	1×2	6.3	67.9
	✓	✓	2×1	8.4	67.1
	✓	✓	2×2	8.1	68.9
	✓	✓	4×1	16.1	68.5

Table 5: Ablation study of the E-3DSNN on Semantic KITTI val benchmarks (Behley et al. 2019). * We convert 20.1M of E-3DSNN into ANN with the same architecture.

Visualization

We evaluated the effectiveness of E-3DSNN in reducing irrelevant redundant points in the background. By training E-3DSNN with and without SVC and SSC on Semantic KITTI (Behley et al. 2019) and KITTI (Geiger et al. 2012a) datasets, we generated the hidden layer feature maps and final detection and segmentation results shown in Fig. 4. It can be observed that our SSC and SVC help 3D SNNs significantly reduce irrelevant redundant points in the background in 3D detection and segmentation tasks. For instance, in the intermediate feature maps Fig. 4 (a) and (b), we notice that most foreground points are preserved, while road points, being easily identifiable as redundant, are largely removed. In the result Fig. 4 (c) and (d), we observe that our E-3DSNN achieves visual effects in detection and segmentation that are comparable to those of ANNs with the same architecture. For instance, in detection, our E-3DSNN has detected all car categories with high confidence. In segmentation, for fine-grained categories such as fence and sidewalk, our E-3DSNN demonstrates excellent segmentation performance.

Conclusion

This paper significantly narrows the performance gap between ANN and SNN on 3D recognition tasks. We accomplish this through two key issues with SNN in processing 3D point cloud data. First, to tackle the disordered and uneven nature of point cloud data, we propose the Spike Voxel Coding (SVC) scheme, which significantly improves storage and preprocessing efficiency. Second, to overcome the rapid increase in computational complexity when applying SNNs to 3D point clouds, we introduce Spike Sparse Convolution (SSC), which reduces redundant computations on background points. The E-3DSNN backbone utilizes these innovations along with residual connections between membrane potentials to handle various 3D computer vision tasks efficiently. Experiments conducted on ModelNet, KITTI, and Semantic KITTI datasets demonstrate that E-3DSNN achieves state-of-the-art performance in terms of accuracy and efficiency across different tasks including 3D classification, object detection, and semantic segmentation. We hope our investigations pave the way for efficient 3D recognition and inspire the design of sparse event-driven SNNs.

Acknowledgements

This work was partially supported by National Key Research and Development Program of China (2023YFF1204200), CAS Project for Young Scientists in Basic Research (YSBR-116), National Distinguished Young Scholars (62325603), National Natural Science Foundation of China (62236009, U22A20103, 62441606), Beijing Science and Technology Plan (Z241100004224011).

References

Behley, J.; Garbade, M.; Milioto, A.; Quenzel, J.; Behnke, S.; Stachniss, C.; and Gall, J. 2019. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 9297–9307.

Caesar, H.; Bankiti, V.; Lang, A. H.; Vora, S.; Liong, V. E.; Xu, Q.; Krishnan, A.; Pan, Y.; Baldan, G.; and Beijbom, O. 2020. nuScenes: A Multimodal Dataset for Autonomous Driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Choy, C.; Gwak, J.; Savarese, S.; and ruijie, z. 2019. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Rattern Recognition (CVPR)*, 3075–3084.

Contributors, P. 2023. Pointcept: A Codebase for Point Cloud Perception Research. https://github.com/Pointcept/Pointcept.

Cui, Y.; Chen, R.; Chu, W.; Chen, L.; Tian, D.; Li, Y.; and Cao, D. 2021. Deep learning for image and point cloud fusion in autonomous driving: A review. *IEEE Transactions on Intelligent Transportation Systems*, 23(2): 722–739.

Deng, J.; Shi, S.; Li, P.; Zhou, W.; Zhang, Y.; and Li, H. 2021. Voxel r-cnn: Towards high performance voxel-based 3d object detection. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, 1201–1209.

Fang, W.; Yu, Z.; Chen, Y.; Huang, T.; Masquelier, T.; and Tian, Y. 2021. Deep residual learning in spiking neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, 21056–21069.

Geiger, A.; Lenz, P.; Urtasun, R.; and ruijie zhu. 2012a. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3354–3361.

Geiger, A.; Lenz, P.; Urtasun, R.; and ruijie zhu. 2012b. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3354–3361.

Graham, B.; Van der Maaten, L.; Ruijie, Z.; and Guoqi, L. 2017. Submanifold sparse convolutional networks. *arXiv preprint arXiv:1706.01307*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016a. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016b. Identity mappings in deep residual networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 630–645.

Horowitz, M. 2014. 1.1 computing's energy problem (and what we can do about it). In *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, 10–14. IEEE.

Hu, J.; Yao, M.; Qiu, X.; Chou, Y.; Cai, Y.; Qiao, N.; Tian, Y.; Xu, B.; and Li, G. 2024a. High-Performance Temporal Reversible Spiking Neural Networks with $O(L)$ Training Memory and $O(1)$ Inference Cost. *arXiv preprint arXiv:2405.16466*.

Hu, Y.; Deng, L.; Wu, Y.; Yao, M.; and Li, G. 2024b. Advancing Spiking Neural Networks Toward Deep Residual Learning. *IEEE Transactions on Neural Networks and Learning Systems*, 1–15.

Lai, X.; Chen, Y.; Lu, F.; Liu, J.; and Jia, J. 2023. Spherical transformer for lidar-based 3d recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 17545–17555.

Lan, Y.; Zhang, Y.; Ma, X.; Qu, Y.; and Fu, Y. 2023. Efficient converted spiking neural network for 3d and 2d classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 9211–9220.

Li, G.; Deng, L.; Tang, H.; Pan, G.; Tian, Y.; Roy, K.; and Maass, W. 2023. Brain inspired computing: A systematic survey and future trends. *Authorea Preprints*.

Li, Y.; Deng, S.; Dong, X.; Gong, R.; and Gu, S. 2021. A free lunch from ANN: Towards efficient, accurate spiking neural networks calibration. In *International conference on machine learning (ICML)*, 6316–6325. PMLR.

Luo, X.; Yao, M.; Chou, Y.; Xu, B.; and Li, G. 2024. Integer-Valued Training and Spike-Driven Inference Spiking Neural Network for High-performance and Energy-efficient Object Detection. *arXiv preprint arXiv:2407.20708*.

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

Neftci, E. O.; Mostafa, H.; and Zenke, F. 2019. Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Processing Magazine*, 36(6): 51–63.

Pomerleau, F.; Colas, F.; Siegwart, R.; et al. 2015. A review of point cloud registration algorithms for mobile robotics. *Foundations and Trends in Robotics*, 4(1): 1–104.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 652–660.

Qiu, X.; Zhu, R.-J.; Chou, Y.; Wang, Z.; Deng, L.-j.; and Li, G. 2024. Gated attention coding for training high-performance and efficient spiking neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 38, 601–610.

Qiu, X.-R.; Wang, Z.-R.; Luan, Z.; Zhu, R.-J.; Wu, X.; Zhang, M.-L.; and Deng, L.-J. 2023. VTSNN: a virtual temporal spiking neural network. *Frontiers in Neuroscience*, 17: 1091097.

Rathi, N.; and Roy, K. 2021. Diet-snn: A low-latency spiking neural network with direct input encoding and leakage and threshold optimization. *IEEE Transactions on Neural Networks and Learning Systems*, 34(6): 3174–3182.

Ren, D.; Ma, Z.; Chen, Y.; Peng, W.; Liu, X.; Zhang, Y.; and Guo, Y. 2024. Spiking PointNet: Spiking Neural Networks for Point Clouds. *Advances in Neural Information Processing Systems (NeurIPS)*, 36: 41797–41808.

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

Shan, Y.; Qiu, X.; Zhu, R.-j.; Li, R.; Wang, M.; and Qu, H. 2023. OR Residual Connection Achieving Comparable Accuracy to ADD Residual Connection in Deep Residual Spiking Neural Networks. *arXiv preprint arXiv:2311.06570*.

Shan, Y.; Zhang, M.; Zhu, R.-j.; Qiu, X.; Eshraghian, J. K.; and Qu, H. 2024. Advancing Spiking Neural Networks towards Multiscale Spatiotemporal Interaction Learning. *arXiv preprint arXiv:2405.13672*.

Shi, S.; Guo, C.; Jiang, L.; Wang, Z.; Shi, J.; Wang, X.; and Li, H. 2020. PV-RCNN: Point-voxel feature set abstraction for 3D object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 10526 – 10535.

Shi, S.; Wang, X.; Li, H.; and Zhu, R. 2019. PointRCNN: 3D Object Proposal Generation and Detection From Point Cloud. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–779.

Team, O. D. 2020. OpenPCDet: An Open-source Toolbox for 3D Object Detection from Point Clouds. https://github.com/open-mmlab/OpenPCDet.

Thomas, H.; Qi, C. R.; Deschaud, J.-E.; Marcotegui, B.; Goulette, F.; and Guibas, L. J. 2019. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 6411–6420.

Wu, Q.; Zhang, Q.; Tan, C.; Zhou, Y.; and Sun, C. 2024a. Point-to-Spike Residual Learning for Energy-Efficient 3D Point Cloud Classification. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 38, 6092–6099.

Wu, X.; Jiang, L.; Wang, P.-S.; Liu, Z.; Liu, X.; Qiao, Y.; Ouyang, W.; He, T.; and Zhao, H. 2024b. Point Transformer V3: Simpler Faster Stronger. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4840–4851.

Wu, X.; Wen, X.; Liu, X.; and Zhao, H. 2023. Masked scene contrast: A scalable framework for unsupervised 3d representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Rattern Recognition (CVPR)*, 9415–9424.

Wu, Y.; Deng, L.; Li, G.; Zhu, J.; and Shi, L. 2018. Spatio-temporal backpropagation for training high-performance spiking neural networks. *Frontiers in Neuroscience*, 12: 331.

Wu, Y.; Deng, L.; Li, G.; Zhu, J.; Xie, Y.; and Shi, L. 2019. Direct training for spiking neural networks: Faster, larger, better. In *Proceedings of the AAAI conference on artificial intelligence (AAAI)*, volume 33, 1311–1318.

Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; and Xiao, J. 2015. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1912–1920.

Yan, Y.; Mao, Y.; Li, B.; and Zhu, R. 2018. SECOND: Sparsely Embedded Convolutional Detection. *Sensors*, 18.

Yao, M.; Hu, J.; Hu, T.; Xu, Y.; Zhou, Z.; Tian, Y.; Bo, X.; and Li, G. 2024a. Spike-driven Transformer V2: Meta Spiking Neural Network Architecture Inspiring the Design of Next-generation Neuromorphic Chips. In *The Twelfth International Conference on Learning Representations (ICLR)*.

Yao, M.; Hu, J.; Zhou, Z.; Yuan, L.; Tian, Y.; Xu, B.; and Li, G. 2023. Spike-driven transformer. In *Proceedings of the 37th International Conference on Neural Information Processing Systems (NeurIPS)*, 64043–64058.

Yao, M.; Qiu, X.; Hu, T.; Hu, J.; Chou, Y.; Tian, K.; Liao, J.; Leng, L.; Xu, B.; and Li, G. 2024b. Scaling Spike-driven Transformer with Efficient Spike Firing Approximation Training. *arXiv preprint arXiv:2411.16061*.

Yao, M.; Richter, O.; Zhao, G.; Qiao, N.; Xing, Y.; Wang, D.; Hu, T.; Fang, W.; Demirci, T.; De Marchi, M.; et al. 2024c. Spike-based dynamic computing with asynchronous sensing-computing neuromorphic chip. *Nature Communications*, 15(1): 4464.

Zhang, Y.; Zhang, Q.; Zhu, Z.; Hou, J.; and Yuan, Y. 2023. Glenet: Boosting 3d object detectors with generative label uncertainty estimation. *International Journal of Computer Vision*, 131(12): 3332–3352.

Zhao, H.; Jiang, L.; Jia, J.; Torr, P. H.; and Koltun, V. 2021. Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 16259–16268.

Zhou, Y.; Tuzel, O.; ruijie, z.; and guoqi, L. 2018. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4490–4499.

Zhu, Q.; Fan, L.; Weng, N.; and Chen, R. 2024. Advancements in point cloud data augmentation for deep learning: A survey. *Pattern Recognition*, 110532.