3DSMT++: TOWARDS HIGH-PERFORMANCE SPIKE-DRIVEN POINT CLOUD LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Spiking Neural Networks (SNNs) offer a promising energy-efficient perception paradigm via sparse, event-driven computation. Recently, deep SNNs have made strong progress in 2D visual representation learning. However, extending these advances to sparse and irregular 3D point clouds remains challenging. Existing spiking point cloud models still lag behind strong Artificial Neural Network (ANN) methods. This is largely because sparse spike representations fail to fully capture rich geometric structures in point clouds. To address this issue, we propose 3DSMT++, an enhanced hybrid Spiking Mamba-Transformer for high-performance and energy-efficient point cloud analysis. First, we introduce Multi-Scale Spiking Patch Embedding (MSPE) to extract complementary geometric features at different spatial scales, together with masked pre-training to improve multi-scale representation learning. We further propose Spiking Distance-aware Offset Attention (SDOA), which leverages spatial relations and feature variations of neighboring points for precise local geometric modeling. In addition, we design a Spiking Point Mamba (SPM) architecture to improve efficient global feature interaction for unordered point clouds. Overall, 3DSMT++ improves multi-scale, local, and global representations while retaining event-driven computational efficiency. Extensive experiments on point cloud benchmarks show that our method achieves SOTA performance among SNN point cloud methods, outperforms many ANN approaches, and maintains competitive computational and energy efficiency.

1 INTRODUCTION

Spiking Neural Networks (SNNs) Maass (1997), inspired by biological neural systems, provide an energy-efficient computing paradigm through sparse binary communication and event-driven neuronal dynamics. Unlike conventional Artificial Neural Networks (ANNs) Priddy (2005) that rely heavily on dense floating-point operations, SNNs activate computation only when spikes are generated, allowing many multiply-and-accumulate operations to be replaced by sparse accumulation. Together with their temporal dynamics and compatibility with neuromorphic computing, these properties make SNNs particularly attractive for building high-performance perception systems under strict computation and energy budgets.

Driven by advances in surrogate-gradient learning and deep network design, SNNs have achieved remarkable progress in deep learning. Early convolutional SNNs have gradually evolved toward more expressive architectures, including the Spikformer Hu et al. (2024) and Spike-driven Transformer Yao et al. (2023), which introduce powerful attention-based representation learning into spike-driven computation. Recently, state-space modeling has further expanded the architectural design space of SNNs toward efficient long-range interaction. Despite these advances, much of the development of deep SNN architectures has been concentrated on 2D visual perception Wang et al. (2025), while their potential for irregular and sparse 3D data remains comparatively under-explored.

Point cloud analysis plays an important role in many real-world 3D perception tasks, including autonomous driving, robotics, and augmented and virtual reality. However, achieving strong point cloud understanding often requires considerable computation and energy, which limits the deployment of advanced models on resource-constrained devices. This challenge makes energy-efficient SNNs particularly attractive for point cloud analysis. Nevertheless, directly extending SNNs to point clouds is non-trivial because sparse binary activations must simultaneously model irregular

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

local geometry, unordered point interactions, and long-range dependencies. Existing spiking point cloud methods have made important progress, and 3DSMT Zhou et al. (2026c) further establishes a hybrid framework that combines spiking local offset attention with Mamba global modeling. However, its representation capability remains insufficient for capturing rich geometric structures in point clouds, limiting its performance compared with strong ANN methods.

To further push the performance boundary of spike-driven point cloud analysis, we build upon 3DSMT and propose 3DSMT++, an enhanced hybrid Spiking Mamba-Transformer with stronger geometric representation capability. Specifically, we introduce Multi-Scale Spiking Patch Embedding (MSPE) to learn complementary geometric representations across different spatial granularities and combine it with masked pre-training to strengthen representation learning. We further develop Spiking Distance-aware Offset Attention (SDOA) to improve local geometric interaction by exploiting spatial relations and feature variations among neighboring points. Finally, a Spiking Point Mamba (SPM) architecture is introduced to enhance efficient global feature modeling. As shown in Figure 1, 3DSMT++ achieves superior point cloud processing performance while retaining the energy-efficiency advantages of spike-driven computation. Our main contributions are summarized as follows:

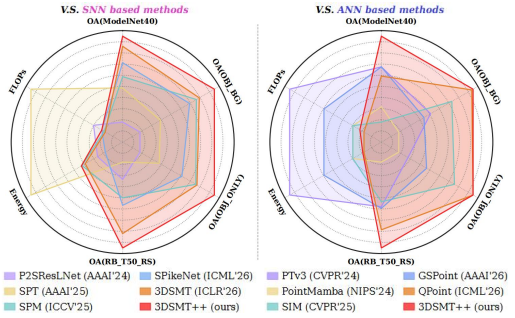


Figure 1: Comparison of our proposed method with SNN and ANN based methods in terms of energy consumption, FLOPs, overall accuracy (OA) on ModelNet40 dataset and three variants of ScanObjectNN dataset.

- We introduce MSPE with masked pre-training, which learns complementary multi-scale geometric representations and improves the representation learning capability of the spiking point cloud backbone by better preserving geometric structures across different spatial scales.
- We enhance both local and global feature modeling by proposing SDOA for distance- and difference-aware local geometric interaction and a SPM architecture for effective global feature integration, enabling richer geometric dependency modeling under sparse representations.
- We present 3DSMT++, a stronger hybrid Spiking Mamba-Transformer that advances spike-driven point cloud analysis toward higher representation capability and task performance.
- Extensive experiments demonstrate that 3DSMT++ achieves state-of-the-art performance among SNN-based point cloud methods on multiple benchmarks, outperforms many ANN-based counterparts, and maintains competitive computational and energy efficiency.

2 Related Work

Deep Spiking Architectures. Spiking Neural Networks (SNNs) have attracted increasing attention as an energy-efficient alternative to conventional Artificial Neural Networks (ANNs), owing to sparse spike communication and event-driven computation. Early studies on deep SNNs mainly focused on network depth and trainability. Representative residual architectures such as SEW-ResNet Fang et al. (2021) improve information and gradient propagation in directly trained SNNs. With the success of Transformers, the Spikformer series Zhou et al. (2022; 2023) explores self-attention and Transformer-style feature interaction under spiking dynamics, while the Spike-driven Transformer series Yao et al. (2023; 2024; 2025) advances fully spike-driven computation through redesigned attention, residual connections, network structures, and training strategies for higher efficiency and stronger performance. More recently, state-space models provide another promising direction for efficient long-range modeling in SNNs. Representative works such as SpikeMba Li et al. (2024) and SpikySpace Tang et al. (2026) integrate spiking dynamics with state-space modeling and selective scanning, extending SNNs beyond attention-based sequence modeling. These developments improve the representation capability and computational efficiency of deep SNNs, providing important foundations for high-performance spike-driven learning.

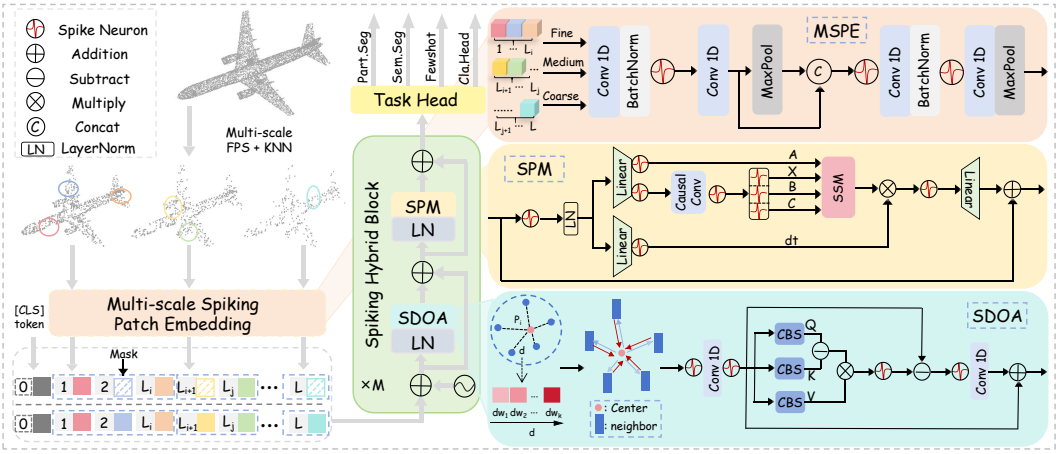


Figure 2: Overall architecture of 3DSMT++. MSPE extracts complementary multi-scale geometric representations, while stacked Spiking Hybrid Blocks combine SDOA for local geometric interaction and SPM for efficient global feature modeling. The resulting representations are used for various point cloud classification and segmentation tasks. Detailed structures of MSPE, SDOA, and SPM are shown on the right.

Point Cloud Learning. Deep point cloud learning has evolved from point-wise feature extraction toward hierarchical, attention-based, and state-space geometric modeling. PointNet Qi et al. (2017a) pioneers direct learning on unordered point sets, laying the foundation for modern point-based 3D representation learning, while PointNet++ Qi et al. (2017b) introduces hierarchical local aggregation to capture geometry at multiple resolutions. The Point Transformer series Zhao et al. (2021); Wu et al. (2022; 2024) progressively improves attention-based geometric interaction and scalable point feature modeling, advancing Transformer architectures for 3D perception. More recently, PointMamba Liang et al. (2024) introduces state-space modeling into point cloud analysis, enabling efficient long-range feature interaction with linear sequence complexity. In parallel, SNN point cloud analysis has emerged as a promising direction for energy-efficient 3D perception. SPT Wu et al. (2025a) introduces Transformer-style representation learning and point-to-spike encoding, while SPM Wu et al. (2025b) combines Mamba global modeling with spike-based pre-training. 3DSMT Zhou et al. (2026c) integrates spiking local offset attention and Mamba global modeling into a unified local-global framework, achieving a favorable balance between performance and energy efficiency. Despite these advances, SNN point cloud models still face challenges in geometric representation, representation learning, and the accuracy-efficiency trade-off compared with strong ANN methods, leaving room for more expressive and efficient spike-driven architectures.

3 METHOD

3.1 OVERVIEW

As illustrated in Fig. 2, 3DSMT++ mainly consists of Multi-Scale Spiking Patch Embedding (MSPE), stacked Spiking Hybrid Blocks (SHBs), and task-specific heads. Given an input point cloud $\mathbf{P} \in \mathbb{R}^{N\times 3}$, MSPE first constructs local patches at different spatial scales and maps them into high-dimensional spike representations. The features from different scales are then concatenated into a unified token sequence. A learnable [CLS] token and spiking positional encoding are further incorporated before feature learning. The resulting tokens are processed by a sequence of SHBs. Each SHB contains a Spiking Distance-aware Offset Attention (SDOA) module for local geometric modeling and a Spiking Point Mamba (SPM) module for global feature interaction. the i-th SHB can be formulated as:

$$\mathbf{O}_0 = [\mathbf{X}_{cls}; \mathbf{F}_{ms}], \tag{1}$$

$$\mathbf{O}'_i = \mathrm{SDOA}\left(\mathrm{LN}(\mathbf{O}_{i-1} + \mathbf{S}_{pos})\right) + \mathbf{O}_{i-1}, \tag{2}$$

$$\mathbf{O}_i = \mathrm{SPM}\left(\mathrm{LN}(\mathbf{O}'_i)\right) + \mathbf{O}'_i, \quad i = 1, \ldots, M, \tag{3}$$

where $\mathbf{F}_{ms}$ denotes the multi-scale features produced by MSPE, $\mathbf{S}_{pos}$ is the spiking positional encoding, and LN denotes LayerNorm. The learned features are finally delivered to different task heads for classification, few-shot learning, part segmentation, and semantic segmentation.

3.2 MULTI-SCALE SPIKING PATCH EMBEDDING

Different spatial scales capture different aspects of point cloud geometry. A single-scale patch embedding is limited in capturing both fine local details and broader structural context, while simply increasing the number of tokens at the same scale may introduce redundant representations. To capture richer geometric information, we propose Multi-Scale Spiking Patch Embedding (MSPE), which learns complementary point representations from fine, medium, and coarse spatial scales. Given the input point cloud $\mathbf{P}$, we progressively downsample it through farthest point sampling (FPS) to obtain three point sets,

$$\mathbf{P}^f = \text{FPS}(\mathbf{P}, N_f), \quad \mathbf{P}^m = \text{FPS}(\mathbf{P}^f, N_m), \quad \mathbf{P}^c = \text{FPS}(\mathbf{P}^m, N_c), \tag{4}$$

where f, m, and c denote the fine, medium, and coarse scales, respectively. At each scale, FPS and KNN are used to construct local patches with a scale-specific neighborhood size. The fine scale preserves denser local structures, while the medium and coarse scales gradually enlarge the spatial context. For each scale $s \in \{f, m, c\}$, the local patches are processed by the same spiking embedding network with shared parameters. Let $\mathbf{E}^s$ denote the local patch features at scale s. The embedding process is formulated as:

$$\mathbf{U}^s = \text{Conv}(\text{SN}(\text{BN}(\text{Conv}(\mathbf{E}^s)))), \qquad \mathbf{G}^s = \text{MaxPool}(\mathbf{U}^s), \tag{5,6}$$

$$\mathbf{F}^s = \text{MaxPool}(\text{MLP}(\text{SN}(\text{Concat}(\mathbf{U}^s, \mathbf{G}^s)))), \quad \mathbf{F}_{ms} = \text{Concat}\big(\mathbf{F}^f, \mathbf{F}^m, \mathbf{F}^c\big). \tag{7,8}$$

where SN denotes the spiking neuron and MLP is implemented with Conv1D and Batch Normalization layers (BN). We allocate different numbers of tokens to different spatial scales, with the token budget gradually reduced from fine to coarse scales. The resulting multi-scale features are directly concatenated along the token dimension, where L_f, L_m, and L_c denote the token numbers at the fine, medium, and coarse scales, respectively, and $L = L_f + L_m + L_c$.

To further improve the learning of multi-scale representations, we apply random token masking after multi-scale feature fusion. Following the spike-based pre-training strategy in SPM Wu et al. (2025b), visible tokens are processed by the spiking encoder, while a lightweight ANN decoder reconstructs the masked point regions. This asymmetric pre-training scheme encourages the backbone to recover missing geometry from the visible multi-scale context. The decoder is removed after pre-training and only the spiking encoder is used for downstream tasks.

3.3 SPIKING DISTANCE-AWARE OFFSET ATTENTION

Local geometric relations are crucial for distinguishing fine-grained structures in irregular point clouds. Although neighborhood grouping establishes local connectivity, processing all neighboring-point features identically ignores their different spatial relevance to the center point. We therefore propose Spiking Distance-aware Offset Attention (SDOA), which uses spatial distance to regulate local feature propagation and constructs efficient vector-subtraction offset attention from feature differences, thereby modeling local feature variations with linear complexity. Given the coordinate $\mathbf{p}_i$ and feature $\mathbf{x}_i$ of a center point, KNN retrieves its neighboring coordinates $\mathbf{p}_{ij}$ and features $\mathbf{x}_{ij}$. We compute the relative feature and Euclidean distance between each neighbor and the center point, and normalize the relative feature within the corresponding neighborhood:

$$\mathbf{r}_{ij} = \mathbf{x}_{ij} - \mathbf{x}_i, \quad d_{ij} = \|\mathbf{p}_{ij} - \mathbf{p}_i\|_2, \quad \hat{\mathbf{r}}_{ij} = \frac{\mathbf{r}_{ij}}{\text{Std}_j(\mathbf{r}_{ij}) + \epsilon} \tag{9,10,11}$$

To distinguish the spatial relevance of different neighbors, we normalize their distances by the mean distance within each neighborhood. The reciprocal distance is then normalized to obtain the propagation weight:

$$\hat{d}_{ij} = \frac{d_{ij}}{\frac{1}{K}\sum_{j=1}^{K} d_{ij} + \epsilon}, \quad g_{ij} = \frac{1}{\hat{d}_{ij} + \epsilon}, \quad w_{ij} = \frac{g_{ij}}{\frac{1}{K}\sum_{j=1}^{K} g_{ij} + \epsilon} \tag{12,13,14}$$

This formulation assigns larger weights to neighbors closer to the center point while maintaining a stable feature scale. The weighted relative feature is concatenated with the center feature and transformed using learnable affine parameters. The propagated neighbor features are then aggregated through exponential weighting:

$$\mathbf{H}_{ij} = \boldsymbol{\alpha} \odot \text{Concat}(w_{ij}\hat{\mathbf{r}}_{ij}, \mathbf{x}_i) + \boldsymbol{\beta}, \quad \mathbf{x}_i^{\text{local}} = \frac{\sum_{j=1}^{K} \mathbf{H}_{ij} \odot \exp(\mathbf{H}_{ij})}{\sum_{j=1}^{K} \exp(\mathbf{H}_{ij})} \tag{15,16}$$

Through this process, spatial distance regulates the contribution of each neighbor during feature propagation, while relative features preserve local structural variations with respect to the center point. After propagation and aggregation, Conv1D and spiking neurons convert the local features into spike representations. Three independent LBS branches subsequently generate the query, key, and value features:

$$\mathbf{S}_2 = \mathrm{SN}\big(\mathrm{Conv1D}\big(\mathrm{SN}(\mathbf{X}^{\mathrm{local}})\big)\big)\,, \tag{17}$$

$$\mathbf{Q} = \mathrm{LBS}_q(\mathbf{S}_2), \mathbf{K} = \mathrm{LBS}_k(\mathbf{S}_2), \mathbf{V} = \mathrm{LBS}_v(\mathbf{S}_2) \tag{18,19,20}$$

Here, LBS denotes a sequence of Linear, normalization, and spiking neuron layers. Instead of constructing pairwise query-key correlations, SDOA directly generates the attention response through vector subtraction between $\mathbf{Q}$ and $\mathbf{K}$, and uses $\mathbf{V}$ to modulate the resulting feature difference. Motivated by the graph Laplacian transformation, which emphasizes the difference between an input feature and its contextual response, we subtract this attention response from the input spike feature to construct the feature offset:

$$\mathbf{A} = (\mathbf{Q} - \mathbf{K}) \odot \mathbf{V}, \quad \mathbf{S}_{\text{out}} = \mathrm{Conv}(\mathrm{SN}(\mathbf{S}_2 - \mathrm{SN}(\mathbf{A}))) + \mathbf{S}_2 \tag{21,22}$$

Since the subtraction and modulation are performed element-wise, SDOA avoids constructing an $L \times L$ attention matrix and reduces the attention complexity from $\mathcal{O}(L^2C)$ to $\mathcal{O}(LC)$. By combining distance-aware feature propagation with vector-subtraction offset attention, SDOA models local geometric relations while retaining efficient spike-driven computation.

3.4 Spiking Point Mamba

While SDOA focuses on local geometric structures, integrating multi-scale point representations also requires effective global interaction. We therefore design Spiking Point Mamba (SPM), a spike-driven global modeling module based on the state-space formulation of Mamba-2 Dao & Gu (2024). Unlike the bidirectional Mamba-1-style structure used in 3DSMT, SPM adopts a single state-space path and converts its feature projection, causal mixing, and output modulation into spike-driven operations. Given the input spike feature $\mathbf{X}$, SPM first applies an SN layer followed by layer normalization. The normalized feature is then linearly projected and converted into spikes, producing the state-transition branch $\mathbf{A}$ and three state-space branches $\mathbf{X}_0$, $\mathbf{B}_0$, and $\mathbf{C}_0$:

$$\widetilde{\mathbf{X}} = \mathrm{LN}(\mathrm{SN}(\mathbf{X}))\,, \quad [\mathbf{A}, \mathbf{X}_0, \mathbf{B}_0, \mathbf{C}_0] = \mathrm{Split}\Big(\mathrm{SN}\Big(\mathrm{Linear}_s(\widetilde{\mathbf{X}})\Big)\Big)\,. \tag{23,24}$$

The state-transition branch $\mathbf{A}$ directly enters the subsequent state-space model. In contrast, $\mathbf{X}_0$, $\mathbf{B}_0$, and $\mathbf{C}_0$ are separately processed by causal convolution followed by an SN layer:

$$\mathbf{X}_s = \mathrm{SN}(\mathrm{CConv}(\mathbf{X}_0))\,, \quad \mathbf{B}_s = \mathrm{SN}(\mathrm{CConv}(\mathbf{B}_0))\,, \quad \mathbf{C}_s = \mathrm{SN}(\mathrm{CConv}(\mathbf{C}_0))\,. \tag{25,26,27}$$

Here, $\mathrm{CConv}(\cdot)$ denotes the causal convolution operation. In parallel, an independent modulation branch applies a linear projection followed by an SN layer to generate the spike-based gating feature $\mathbf{dt} = \mathrm{SN}(\mathrm{Linear}_{dt}(\widetilde{\mathbf{X}}))$. The four branches $\mathbf{A}$, $\mathbf{X}_s$, $\mathbf{B}_s$, and $\mathbf{C}_s$ are jointly processed by the Mamba-2 state-space model, where $\mathbf{X}_s$ provides the state input, $\mathbf{A}$ controls state evolution, and $\mathbf{B}_s$ and $\mathbf{C}_s$ regulate input-state and state-output interactions. Denoting the Mamba-2 state-space operator as $\mathcal{M}_2(\cdot)$, the state-space propagation and modulated output are jointly written as

$$\mathbf{Y} = \mathcal{M}_2(\mathbf{X}_s; \mathbf{A}, \mathbf{B}_s, \mathbf{C}_s)\,, \quad \mathbf{O} = \mathbf{X} + \mathrm{Linear}_o(\mathrm{SN}(\mathbf{Y}) \odot \mathbf{dt})\,. \tag{28,29}$$

Here, $\odot$ denotes element-wise multiplication. By retaining $\mathbf{A}$ as a direct state-transition branch while applying causal spike mixing to $\mathbf{X}$, $\mathbf{B}$, and $\mathbf{C}$, SPM separates state evolution from local context encoding. The Mamba-2 state-space model then propagates the context-enhanced features globally, while the spike-based $\mathbf{dt}$ branch selects the effective state response, enabling local-to-global feature interaction through a single sparse state-space path.

4 Experiments

We evaluate 3DSMT++ on ModelNet40 (Wu et al., 2015) for object and few-shot classification, on the PB_T50_RS, OBJ_BG, and OBJ_ONLY variants of ScanObjectNN (Uy et al., 2019) for object classification, on ShapeNetPart (Yi et al., 2016) for part segmentation, and on S3DIS (Armeni et al., 2016) for semantic segmentation. All experiments are implemented in PyTorch with SpikingJelly (Fang et al., 2023) and conducted on NVIDIA RTX 4090 GPUs with 24 GB memory, using one GPU for training from scratch and three GPUs for pre-training on ShapeNet (Chang et al., 2015). Dataset descriptions, training configurations, proofs and experiments are provided in the *Appendix*.

Table 1: Classification results on ScanObjectNN and ModelNet40. We report overall accuracy (OA, %) on three ScanObjectNN variants (BG: OBJ_BG, ONLY: OBJ_ONLY, RS: PB_T50_RS), and OA on ModelNet40. FLOPs are in Giga (G) and Energy in millijoule (mJ). Bold and underline denote the best and second-best results, respectively; '-' indicates results not reported in the original paper; '*' marks our reproduced results.

Methods	Type	Flops ↓	ModelNet40		ScanObjectNN			
			OA ↑	Energy ↓	BG (OA) ↑	ONLY (OA) ↑	RS (OA) ↑	Energy ↓
Supervised Learning Only								
PTv3 Wu et al. (2024)[CVPR]	ANN	18.9*	**94.5**	86.9*	90.7	88.4	87.9	86.9*
PointMamba Liang et al. (2024)[NIPS]	ANN	3.6	93.2	16.6	88.3	87.8	82.5	16.6
PCM Zhang et al. (2025)[AAAI]	ANN	45.0	93.4	207.0	-	-	88.1	207.0
SIM Bahri et al. (2025)[CVPR]	ANN	3.6	92.7	16.6	92.3	91.4	87.3	16.6
GSPoint Yuan et al. (2026)[AAAI]	ANN	10.6	**94.5**	48.8	90.2*	89.6*	88.1	48.8
QPoint Zhou et al. (2026a)[ICML]	ANN	0.9	94.2	**4.2**	**93.8**	**92.6**	**90.7**	**4.2**
SPTWu et al. (2025a)[AAAI]	SNN	14.0	91.4	13.3	82.8	83.4	78.0	14.6
E-3DSNNQiu et al. (2025)[AAAI]	SNN	2.1	91.7	1.8	-	-	83.4	2.6
SPMWu et al. (2025b)[ICCV]	SNN	6.1	92.3	5.4	90.2	89.5	84.2	7.6
SDT Lu et al. (2026)[ICLR]	SNN	1.5	92.5	**1.3**	-	-	86.2	2.1
SpikeNet Zhou et al. (2026b)[ICML]	SNN	1.9	93.4	1.7	88.8*	87.2*	85.5	**1.8**
SVL Qiu et al. (2026)[ICML]	SNN	5.7	92.6	5.1	-	-	82.1	-
3DSMT Zhou et al. (2026c)[ICLR]	SNN	**1.3**	94.7	4.3	90.8	89.7	90.4	4.9
3DSMT++ (ours)	SNN	1.9	**95.5**(↑ 0.8)	5.5	**93.9**(↑ 3.1)	**92.6**(↑ 2.9)	**92.9**(↑ 2.5)	6.2
with Self-supervised Pre-training								
Point-BERT Yu et al. (2022)[CVPR]	ANN	4.8	93.4	22.1	88.1	88.3	83.7	22.1
MaskPoint Liu et al. (2022)[ECCV]	ANN	4.8	93.8	22.1	89.3	88.1	84.3	22.1
Point-MAE Pang et al. (2022)[ECCV]	ANN	4.8	93.8	22.1	90.0	88.3	85.2	22.1
Point-M2AE Zhang et al. (2022)[NIPS]	ANN	3.6	94.0	16.6	91.2	88.8	86.4	16.6
PointGPT Chen et al. (2024)[NIPS]	ANN	-	94.0	-	91.6	90.0	86.9	-
Mamba3D Han et al. (2024)[ACM MM]	ANN	3.9	**94.7**	17.9	**93.1**	**92.1**	**92.1**	17.9
PointMamba Liang et al. (2024)[NIPS]	ANN	3.6	93.6	16.6	90.7	88.4	84.9	16.6
SPTWu et al. (2025a)[AAAI]	SNN	14.0	92.5	13.3	87.3*	86.8*	82.6	14.6
SPMWu et al. (2025b)[ICCV]	SNN	6.1	93.1	5.4	91.3*	90.4*	86.5	7.6
SVL Qiu et al. (2026)[ICML]	SNN	10.9	93.9	9.8	-	-	83.4	9.8
3DSMT Zhou et al. (2026c)[ICLR]	SNN	**1.3**	94.9*	**4.3**	92.1*	90.7*	91.0*	**4.9**
3DSMT++ (ours)	SNN	1.9	**95.6**(↑ 0.7)	5.5	**95.0**(↑ 2.9)	**93.4**(↑ 2.7)	**92.9**(↑ 1.9)	6.2

4.1 Comparison with State-of-the-art Methods

We compare 3DSMT++ with representative ANN and SNN methods on classification and segmentation tasks. For classification, we report overall accuracy (OA), while few-shot classification uses the average accuracy and standard deviation under 5-way 10-shot, 5-way 20-shot, 10-way 10-shot, and 10-way 20-shot settings. For segmentation, we report category mIoU (Cat.mIoU) and instance mIoU (Ins.mIoU) on ShapeNetPart, OA and mIoU on S3DIS. We further report FLOPs and estimated energy consumption to evaluate the trade-off between task performance and efficiency.

Shape Classification on ModelNet40. ModelNet40 evaluates object-level shape classification on clean 3D point clouds and serves as a standard benchmark for measuring point cloud representation quality. As shown in Table 1, when trained from scratch, 3DSMT++ achieves 95.5% OA, outperforming 3DSMT by 0.8%. Compared with the ANN method GSPoint, our model improves accuracy by 1.0%, while reducing FLOPs from 10.6G to 1.9G and energy consumption from 48.8 mJ to 5.5 mJ. With pre-training, 3DSMT++ further reaches 95.6% OA, exceeding 3DSMT and Mamba3D by 0.7% and 0.9%, respectively. Compared with Mamba3D, our method also reduces FLOPs from 3.9G to 1.9G and energy consumption from 17.9 mJ to 5.5 mJ. These improvements on clean point cloud object shapes further demonstrate the stronger point cloud representation capability of 3DSMT++, while maintaining competitive computational and energy efficiency.

Shape Classification on ScanObjectNN. ScanObjectNN provides a challenging real-world benchmark with complex backgrounds and object variations, making it suitable for evaluating the robustness of point cloud models. As shown in Table 1, when trained from scratch, 3DSMT++ achieves 93.9%, 92.6%, and 92.9% OA on BG, ONLY, and RS, outperforming 3DSMT by 3.1%, 2.9%, and 2.5%, respectively. Compared with the ANN method GSPoint, 3DSMT++ improves the three results by 3.7%, 3.0%, and 4.8%, while reducing FLOPs from 10.6 G to 1.9 G and energy consumption from 48.8 mJ to 6.2 mJ. With pre-training, 3DSMT++ further reaches 95.0%, 93.4%, and 92.9%, exceeding 3DSMT by 2.9%, 2.7%, and 1.9%. Compared with Mamba3D, our method improves the three results by 1.9%, 1.3%, and 0.8%, while reducing FLOPs from 3.9 G to 1.9 G and energy consumption from 17.9 mJ to 6.2 mJ. Overall, 3DSMT++ shows strong robustness on complex real-world point clouds while maintaining low computational and energy costs.

Table 2: Few-shot Classification results on ModelNet40. We report the average accuracy (%) and standard deviation (%) with only Supervised Learning and Self-supervised Pre-training.

Methods	Type	5-way		10-way	
		10-shot↑	20-shot↑	10-shot↑	20-shot↑
Supervised Learning Only					
PointNet Qi et al. (2017a)[CVPR]	ANN	52.0 ± 3.8	57.8± 4.9	46.6 ± 4.3	35.2± 4.8
PointNet++ Qi et al. (2017b)[NIPS]	ANN	38.5± 3.6	42.4 ± 2.9	23.1± 4.8	18.8± 5.2
PointMLP Ma et al. (2022)[ICLR]	ANN	90.5± 4.2	94.2± 4.1	84.1± 5.6	91.5± 5.1
Mamba3D Han et al. (2024)[ACM MM]	ANN	92.6± 3.7	96.9± 2.4	88.1± 5.3	93.1± 3.6
QPoint Zhou et al. (2026a)[ICML]	ANN	**95.0± 3.4**	**98.1± 1.7**	**90.7± 5.0**	**94.1± 2.3**
SpikePointNet Ren et al. (2024)[NIPS]	SNN	52.3± 3.2	57.1± 3.8	46.2± 2.6	34.1 ± 4.2
SpikeNet Zhou et al. (2026b)[ICML]	SNN	87.6*± 4.9	90.4*± 4.8	81.8*± 5.1	87.1*± 5.2
3DSMT Zhou et al. (2026c)[ICLR]	SNN	92.8± 4.3	96.2± 3.6	87.2± 5.7	92.1± 6.1
3DSMT++ (ours)	SNN	**93.3**(↑ 0.5)± **4.1**	**96.4**(↑ 0.2)± **3.0**	**88.0**(↑ 0.8)± **4.9**	**92.6**(↑ 0.5)± **4.0**
with Self-supervised Pre-training					
Point-BERT Yu et al. (2022)[CVPR]	ANN	94.6 ± 3.1	96.3± 2.7	91.0 ± 5.4	92.7± 5.1
Point-M2AE Zhang et al. (2022)[NIPS]	ANN	96.8± 1.8	98.3± 1.4	92.3± 4.5	95.0± 3.0
PointMamba Liang et al. (2024)[NIPS]	ANN	95.0± 2.3	97.3± 1.8	91.4± 4.4	92.8± 4.0
Mamba3D Han et al. (2024)[ACM MM]	ANN	96.4± 2.2	98.2± 1.2	92.4± 4.1	95.2± 2.9
PointGPT Chen et al. (2024)[NIPS]	ANN	**96.8± 2.0**	**98.6± 1.1**	**92.6± 4.6**	**95.2± 3.4**
3DSMT Zhou et al. (2026c)[ICLR]	SNN	93.0·*± 3.9	96.3*± 3.2	87.6*± 6.0	92.3*± 5.8
3DSMT++ (ours)	SNN	**93.6**(↑ 0.6)± **3.9**	**96.8**(↑ 0.5)± **3.2**	**88.4**(↑ 0.8)± **5.6**	**92.9**(↑ 0.6)± **5.1**

Table 3: Segmentation results on ShapeNetPart part segmentation and S3DIS semantic segmentation datasets.

Methods	Type	ShapeNetPart				S3DIS(Area 5)			
		Cat.mIoU ↑	Ins.mIoU ↑	Flops ↓	Energy ↓	OA ↑	mIoU ↑	Flops ↓	Energy ↓
PTv1 Zhao et al. (2021)[ICCV]	ANN	83.7	86.5	27.6	127.0	90.8	70.4	16.7	76.8
PTv2 Wu et al. (2022)[NIPS]	ANN	84.0	86.7	-	-	91.1	71.6	87.0	400.1
PTv3 Wu et al. (2024)[CVPR]	ANN	84.8	87.1	-	-	**91.7**	**73.6**	149.5	687.7
PCM Zhang et al. (2025)[AAAI]	ANN	85.3	87.0	45.0	207.0	88.2	63.4	64.6	297.2
GSPoint Yuan et al. (2026)[AAAI]	ANN	**85.6**	**87.2**	-	-	91.2	71.5	-	-
QPoint Zhou et al. (2026a)[ICML]	ANN	85.1	87.0	**4.3**	**19.8**	91.0	71.2	**9.4**	**43.2**
E-3DSNNQiu et al. (2025)[AAAI]	SNN	81.7	83.8	9.8	8.8	89.8	67.4	16.1	14.4
SPMWu et al. (2025b)[ICCV]	SNN	82.3	84.8	18.3	16.5	90.1	67.8	21.2	19.1
SpikeNet Zhou et al. (2026b)[ICML]	SNN	83.9	85.4	6.1	5.4	90.4	68.9	11.3	10.2
SDT Lu et al. (2026)[ICLR]	SNN	83.7	85.1	**5.2**	**4.7**	90.1	69.6	**8.1**	**7.3**
3DSMT Zhou et al. (2026c)[ICLR]	SNN	82.7	85.1	11.3	10.2	90.6	70.2	13.7	11.4
3DSMT++ (ours)	SNN	**84.1**(↑ 0.2)	**85.9**(↑ 0.5)	13.1	11.6	**90.9**(↑ 0.3)	**71.0**(↑ 0.8)	15.9	14.0

Few-shot Classification on ModelNet40. Few-shot classification evaluates the generalization ability of point cloud models with only limited labeled samples. As shown in Table 2, when trained from scratch, 3DSMT++ achieves the best performance among SNN methods in all four settings and improves 3DSMT by 0.5%, 0.2%, 0.8%, and 0.5%, respectively. With pre-training, our method further improves 3DSMT by 0.6%, 0.5%, 0.8%, and 0.6%, respectively. Although several ANN methods achieve higher accuracy in some settings, 3DSMT++ retains energy-efficient spike-driven computation. The consistent gains across all four settings indicate that the enhanced multi-scale representations generalize well when only limited labeled samples are available.

Part Segmentation on ShapeNetPart. Part segmentation requires a model to distinguish fine-grained geometric structures within each object, placing high demands on local feature modeling. As shown in Table 3, 3DSMT++ achieves 84.1% Cat.mIoU and 85.9% Ins.mIoU, setting SOTA performance among SNN works. Compared with SpikeNet, our method improves the metrics by 0.2% and 0.5%, respectively. Compared with the ANN method QPoint, 3DSMT++ shows gaps of only 1.0% and 1.1%, while reducing energy consumption from 19.8 mJ to 11.6 mJ. Such performance suggests that the proposed multi-scale representation and local geometric modeling better capture fine-grained point structures without sacrificing the energy advantage of spiking computation.

Semantic Segmentation on S3DIS. Semantic segmentation on S3DIS requires dense point-wise prediction over large indoor scenes and demands both strong geometric modeling and efficient computation. In Table 3, 3DSMT++ achieves 90.9% OA and 71.0% mIoU, outperforming all compared SNN methods and improving 3DSMT by 0.3% and 0.8%, respectively. Compared with the ANN method QPoint, our model shows only a 0.1% gap in OA and a 0.2% gap in mIoU, while reducing energy cost from 43.2 mJ to 14.0 mJ. Taken together, the competitive accuracy and much lower energy cost highlight a favorable accuracy-energy trade-off for large-scale 3D scene understanding.

Table 4: Classification Results of Different Architecture Variants on ScanObjectNN and ModelNet40.

Method	Type	ScanObjectNN				ModelNet40		
		BG (OA) ↑	ONLY (OA) ↑	RS (OA) ↑	Energy ↓	OA↑	mAcc↑	Energy↓
Full	ANN	94.8	93.2	93.4	42.1	95.6	92.4	42.1
Full	SNN	**93.9**	**92.6**	**92.9**	6.2	**95.5**	**92.2**	5.5
w/o MSPE	SNN	92.7(↓ 1.2)	91.0(↓ 1.6)	91.2(↓ 1.7)	4.6	93.9(↓ 1.6)	90.6(↓ 1.6)	4.0
w/o SDOA	SNN	93.1(↓ 0.8)	92.1(↓ 0.5)	92.2(↓ 0.7)	5.2	94.7(↓ 0.8)	91.5(↓ 0.7)	4.7
w/o SPM	SNN	92.9(↓ 1.0)	91.5(↓ 1.1)	92.0(↓ 0.9)	4.9	94.5(↓ 1.0)	91.2(↓ 1.0)	4.5
w/o SDOA+SPM	SNN	91.7(↓ 2.2)	90.2(↓ 2.4)	90.2(↓ 2.7)	4.2	93.6(↓ 1.9)	89.8(↓ 2.4)	3.6
w/o MSPE+SDOA	SNN	90.9(↓ 3.0)	89.4(↓ 3.2)	89.6(↓ 3.3)	4.0	93.3(↓ 2.2)	88.9(↓ 3.3)	3.5
w/o MSPE+SPM	SNN	90.4(↓ 3.5)	88.6(↓ 4.0)	89.0(↓ 3.9)	4.0	93.0(↓ 2.5)	88.6(↓ 3.6)	3.5
w/o MSPE+SDOA+SPM	SNN	87.3(↓ 6.6)	85.7(↓ 6.9)	87.0(↓ 5.9)	3.9	92.6(↓ 2.9)	88.2 (↓ 4.0)	3.3

Table 5: Effect of TimeStep on ModelNet40

TimeStep	OA	mAcc
1	94.2	90.8
2	94.7	91.5
4	**95.5**	**92.2**
6	95.3	91.9

Table 6: Effect of different point scales on ModelNet40

L token	OA	mAcc
64	93.3	90.6
128	**95.5**	**92.2**
192	95.1	91.9
256	94.5	91.3

Table 7: Effect of Mamba Strategy on ModelNet40

Mamba Strategy	OA	mAcc
NOrder	**95.5**	**92.2**
Shuffle	94.8	91.5
Bi-NOrder	95.4	92.0
Bi-Shuffle	94.9	91.7

4.2 Ablation Studies

Analysis of Architecture Components. 3DSMT++ strengthens point cloud representation from multi-scale, local, and global perspectives through MSPE, SDOA, and SPM. To clarify the source of these gains, we conduct component-wise ablations. As shown in Table 4, removing MSPE causes the largest individual degradation, decreasing OA by 1.2%, 1.6%, and 1.7% on BG, ONLY, and RS, and by 1.6% on ModelNet40. Removing SDOA or SPM also reduces performance, confirming the effectiveness of local and global modeling. Removing all three modules causes larger drops of 6.6%, 6.9%, and 5.9% on ScanObjectNN and 2.9% on ModelNet40. We further compare the full SNN with an ANN counterpart obtained by replacing spiking neurons with ReLU activations. The ANN version achieves slightly higher accuracy on ScanObjectNN and comparable performance on ModelNet40, but consumes 42.1 mJ, compared with only 6.2 mJ and 5.5 mJ for the SNN version. Taken together, the three modules provide complementary gains, while spike-driven computation retains nearly unchanged accuracy with substantially lower energy consumption.

Analysis of SNN Timestep. The number of time steps determines how long the network integrates spike information and directly affects feature representation, motivating us to study its influence on point cloud performance. As shown in Table 5, increasing the time step from 1 to 4 improves OA from 94.2% to 95.5% and mAcc from 90.8% to 92.2%. More time steps allow the network to accumulate richer spike responses, while increasing it to 6 slightly reduces OA and mAcc to 95.3% and 91.9%, suggesting redundant temporal responses. Overall, four time steps provide the best balance between temporal integration and spike computation.

Analysis of different point scales. The total token number L produced by the MSPE module determines how much multi-scale geometric information is retained and affects the balance between representation capacity and feature redundancy. We therefore examine how different token numbers influence model performance. As shown in Table 6, increasing L from 64 to 128 improves OA from 93.3% to 95.5% and mAcc from 90.6% to 92.2%. However, further increasing L to 192 and 256 reduces OA to 95.1% and 94.5%, respectively. This non-monotonic trend indicates that simply increasing the token number does not continuously improve point cloud representation. The choice of $L = 128$ provides a suitable balance between multi-scale geometry and token redundancy.

Analysis of Mamba Strategy. Since the SPM module relies on state-space modeling for global feature interaction, we investigate how different sequence strategies affect its performance. NOrder keeps the input token sequence without additional ordering, while Shuffle randomly permutes the sequence; the prefix Bi denotes a bidirectional Mamba SSM. As shown in Table 7, NOrder achieves the best performance with 95.5% OA and 92.2% mAcc. Bi-NOrder performs similarly with 95.4% OA and 92.0% mAcc, whereas Shuffle and Bi-Shuffle reduce OA to 94.8% and 94.9%. Random permutation may disturb useful geometric relations, while bidirectional modeling brings limited gains. NOrder offers the best accuracy with a simpler unidirectional SSM design.

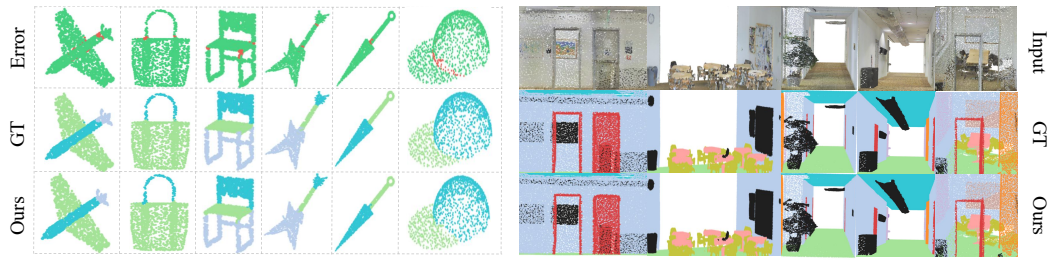


Figure 3: Visualization of Part Segmentation on ShapeNetPart and Semantic Segmentation on S3DIS.

Visualization. Segmentation visualization provides an intuitive way to examine whether 3DSMT++ can capture fine-grained object structures and complex scene-level geometry. Figure 3 presents qualitative results on ShapeNetPart and S3DIS. For ShapeNetPart, the predictions of 3DSMT++ closely match the ground truth across different object categories, while most errors appear only around part boundaries or small local structures. On S3DIS, our predictions also preserve the main semantic structures of complex indoor scenes and show high consistency with the ground truth. These visual results further support the effectiveness of 3DSMT++ in modeling both fine-grained local geometry and large-scale scene structures.

Energy Calculation. Energy efficiency is a key advantage of spike-driven computation. Following the energy estimation protocol of 3DSMT, we convert the computation of spiking layers into spike-driven accumulation (AC) operations. For the l-th spiking layer, the number of spike operations (SOPs) is estimated as

$$\mathrm{SOPs}_l = fr_l \times T \times \mathrm{FLOPs}(l), \tag{30}$$

where fr_l denotes the firing rate of the l-th layer, T is TimeStep, and $\mathrm{FLOPs}(l)$ denotes the corresponding operations. In Figure 4, the SDOA and SPM modules maintain sparse firing across different blocks. Such sparse activity reduces effective SOPs and supports energy-efficient spike-driven computation. Accordingly, the total inference energy is estimated as

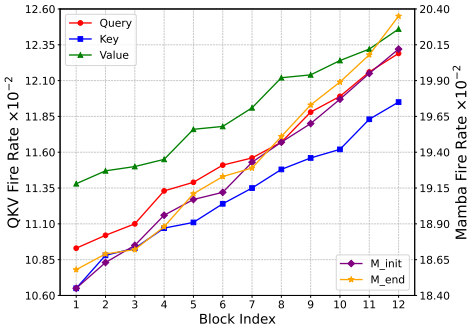


Figure 4: The firing rates of Q, K, V and M_{init}, M_{end}.

$$E = E_{\mathrm{MAC}} \times \mathrm{FLOPs}_{\mathrm{NS}} + E_{\mathrm{AC}} \times \sum_{l=1}^{N_s} \mathrm{SOPs}_l, \tag{31}$$

where $\mathrm{FLOPs}_{\mathrm{NS}}$ denotes the small number of non-spiking operations located at the model input and task head. Note that this estimation focuses on the computational energy of the neural network and does not account for additional hardware-level overheads. We use $E_{\mathrm{MAC}} = 4.6$ pJ and $E_{\mathrm{AC}} = 0.9$ pJ for one MAC and AC operation, respectively (Kundu et al., 2021; Horowitz, 2014). Combined with sparse spike activity, low-cost AC operations enable 3DSMT++ to achieve lower energy consumption while maintaining strong point cloud representation.

5 Conclusion

In this paper, we present 3DSMT++, a hybrid Spiking Mamba-Transformer for high-performance and energy-efficient point cloud analysis. To improve the representation capability of spiking point cloud models, 3DSMT++ combines MSPE with pre-training for complementary multi-scale geometry, SDOA for distance-aware local interactions, and SPM for efficient global modeling. Together, these designs strengthen point cloud representation while retaining sparse spike-driven computation. Extensive experiments on classification and segmentation benchmarks show that 3DSMT++ achieves state-of-the-art performance among SNN methods, remains competitive with strong ANN models, and maintains favorable computational and energy efficiency. 3DSMT++ further narrows the performance gap between SNNs and ANNs, providing an effective solution for high-performance and energy-efficient 3D perception.

REFERENCES

Iro Armeni, Ozan Sener, Amir R Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 3d semantic parsing of large-scale indoor spaces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1534–1543, 2016.

Ali Bahri, Moslem Yazdanpanah, Mehrdad Noori, Sahar Dastani, Milad Cheraghalikhani, Gustavo Adolfo Vargas Hakim, David Osowiechi, Farzad Beizaee, Ismail Ben Ayed, and Christian Desrosiers. Spectral informed mamba for robust point cloud processing. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 11799–11809, 2025.

Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

Guangyan Chen, Meiling Wang, Yi Yang, Kai Yu, Li Yuan, and Yufeng Yue. Pointgpt: Auto-regressively generative pre-training from point clouds. *Advances in Neural Information Processing Systems*, 36, 2024.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=ztn8FCR1td.

Wei Fang, Zhaofei Yu, Yanqi Chen, Tiejun Huang, Timothée Masquelier, and Yonghong Tian. Deep residual learning in spiking neural networks. *Advances in neural information processing systems*, 34:21056–21069, 2021.

Wei Fang, Yanqi Chen, Jianhao Ding, Zhaofei Yu, Timothée Masquelier, Ding Chen, Liwei Huang, Huihui Zhou, Guoqi Li, and Yonghong Tian. Spikingjelly: An open-source machine learning infrastructure platform for spike-based intelligence. *Science Advances*, 9(40):eadi1480, 2023.

Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R. Martin, and Shi-Min Hu. Pct: Point cloud transformer. *Computational Visual Media*, 7(2):187–199, 2021. ISSN 2096-0433. doi: 10.1007/s41095-021-0229-5. URL http://dx.doi.org/10.1007/s41095-021-0229-5.

Xu Han, Yuan Tang, Zhaoxuan Wang, and Xianzhi Li. Mamba3d: Enhancing local features for 3d point cloud analysis via state space model. *arXiv preprint arXiv:2404.14966*, 2024.

Mark Horowitz. 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)*, pp. 10–14. IEEE, 2014.

Yifan Hu, Lei Deng, Yujie Wu, Man Yao, and Guoqi Li. Advancing spiking neural networks toward deep residual learning. *IEEE transactions on neural networks and learning systems*, 36(2):2353–2367, 2024.

Souvik Kundu, Massoud Pedram, and Peter A Beerel. Hire-snn: Harnessing the inherent robustness of energy-efficient deep spiking neural networks by training with crafted input noise. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 5209–5218, 2021.

Wenrui Li, Xiaopeng Hong, Ruiqin Xiong, and Xiaopeng Fan. Spikemba: Multi-modal spiking saliency mamba for temporal video grounding. *arXiv preprint arXiv:2404.01174*, 2024.

Dingkang Liang, Xin Zhou, Xinyu Wang, Xingkui Zhu, Wei Xu, Zhikang Zou, Xiaoqing Ye, and Xiang Bai. Pointmamba: A simple state space model for point cloud analysis. *arXiv preprint arXiv:2402.10739*, 2024.

Haotian Liu, Mu Cai, and Yong Jae Lee. Masked discrimination for self-supervised learning on point clouds. *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022.

Yijie Lu, Zhiyi Pan, Renrui Zhang, Yanhao Jia, Ronggang Wang, and Zhaokun Zhou. Spiking discrepancy transformer for point cloud analysis. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=7Brnh0aNFn.

Xu Ma, Can Qin, Haoxuan You, Haoxi Ran, and Yun Fu. Rethinking network design and local geometry in point cloud: A simple residual mlp framework. *arXiv preprint arXiv:2202.07123*, 2022.

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

Yatian Pang, Wenxiao Wang, Francis EH Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked autoencoders for point cloud self-supervised learning. In *European conference on computer vision*, pp. 604–621. Springer, 2022.

K. L. Priddy. *Artificial Neural Networks: An Introduction*. PNWD - SA - 6806. SPIE Press, Bellingham, Washington, 2005.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 652–660, 2017a.

Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017b.

Xuerui Qiu, Man Yao, Jieyuan Zhang, Yuhong Chou, Ning Qiao, Shibo Zhou, Bo Xu, and Guoqi Li. Efficient 3d recognition with event-driven spike sparse convolution. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 20086–20094, 2025.

Xuerui Qiu, Shaowei Gu, Peixi Wu, JiaKui Hu, Yaozhi Wen, Yuqi Pan, Xinhao Luo, Bo XU, and Guoqi Li. SVL: Empowering spiking neural networks for efficient 3d open-world understanding. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=Ai3a79cTvr.

Dayong Ren, Zhe Ma, Yuanpei Chen, Weihang Peng, Xiaode Liu, Yuhan Zhang, and Yufei Guo. Spiking pointnet: Spiking neural networks for point clouds. *Advances in Neural Information Processing Systems*, 36, 2024.

Kaiwen Tang, Jiaqi Zheng, Yuze Jin, Yupeng Qiu, Guangda Sun, Zhanglu Yan, and Weng-Fai Wong. Spikyspace: A spiking state space model for energy-efficient time series forecasting. *arXiv preprint arXiv:2601.02411*, 2026.

Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1588–1597, 2019.

Shuai Wang, Malu Zhang, Dehao Zhang, Ammar Belatreche, Yichen Xiao, Yu Liang, Yimeng Shan, Qian Sun, Enqi Zhang, and Yang Yang. Spiking vision transformer with saccadic attention. *arXiv preprint arXiv:2502.12677*, 2025.

Peixi Wu, Bosong Chai, Hebei Li, Menghua Zheng, Yansong Peng, Zeyu Wang, Xuan Nie, Yueyi Zhang, and Xiaoyan Sun. Spiking point transformer for point cloud classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 21563–21571, 2025a.

Peixi Wu, Bosong Chai, Menghua Zheng, Wei Li, Zhangchi Hu, Jie Chen, Zheyu Zhang, Hebei Li, and Xiaoyan Sun. Efficient spiking point mamba for point cloud analysis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 26393–26403, 2025b.

Xiaoyang Wu, Yixing Lao, Li Jiang, Xihui Liu, and Hengshuang Zhao. Point transformer v2: Grouped vector attention and partition-based pooling. *Advances in Neural Information Processing Systems*, 35:33330–33342, 2022.

Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point transformer v3: Simpler, faster, stronger. In *CVPR*, 2024.

Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1912–1920, 2015.

Man Yao, Jiakui Hu, Zhaokun Zhou, Li Yuan, Yonghong Tian, Bo Xu, and Guoqi Li. Spike-driven transformer. *Advances in neural information processing systems*, 36:64043–64058, 2023.

Man Yao, JiaKui Hu, Tianxiang Hu, Yifan Xu, Zhaokun Zhou, Yonghong Tian, Bo Xu, and Guoqi Li. Spike-driven transformer v2: Meta spiking neural network architecture inspiring the design of next-generation neuromorphic chips. In *International conference on learning representations*, volume 2024, pp. 52885–52907, 2024.

Man Yao, Xuerui Qiu, Tianxiang Hu, Jiakui Hu, Yuhong Chou, Keyu Tian, Jianxing Liao, Luziwei Leng, Bo Xu, and Guoqi Li. Scaling spike-driven transformer with efficient spike firing approximation training. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(4): 2973–2990, 2025.

Li Yi, Vladimir G Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. A scalable active framework for region annotation in 3d shape collections. *ACM Transactions on Graphics (ToG)*, 35(6):1–12, 2016.

Xumin Yu, Lulu Tang, Yongming Rao, Tiejun Huang, Jie Zhou, and Jiwen Lu. Point-bert: Pre-training 3d point cloud transformers with masked point modeling. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 19313–19322, 2022.

Shangbo Yuan, Jie Xu, Ping Hu, Xiaofeng Zhu, and Na Zhao. Graph smoothing for enhanced local geometry learning in point cloud analysis, 2026. URL https://arxiv.org/abs/2601.11102.

Renrui Zhang, Ziyu Guo, Peng Gao, Rongyao Fang, Bin Zhao, Dong Wang, Yu Qiao, and Hongsheng Li. Point-m2ae: multi-scale masked autoencoders for hierarchical point cloud pre-training. *Advances in neural information processing systems*, 35:27061–27074, 2022.

Tao Zhang, Haobo Yuan, Lu Qi, Jiangning Zhang, Qianyu Zhou, Shunping Ji, Shuicheng Yan, and Xiangtai Li. Point cloud mamba: Point cloud learning via state space model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 10121–10130, 2025.

Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 16259–16268, 2021.

Chenlin Zhou, Liutao Yu, Zhaokun Zhou, Zhengyu Ma, Han Zhang, Huihui Zhou, and Yonghong Tian. Spikingformer: Spike-driven residual learning for transformer-based spiking neural network. *arXiv e-prints*, pp. arXiv–2304, 2023.

Zhaokun Zhou, Yuesheng Zhu, Chao He, Yaowei Wang, Shuicheng Yan, Yonghong Tian, and Li Yuan. Spikformer: When spiking neural network meets transformer. *arXiv preprint arXiv:2209.15425*, 2022.

Zhiming Zhou, Yong He, Chaoxu Mu, Qiaoyun Wu, and Ajmal Saeed Mian. QPoint: End-to-end lightweight point cloud processing via robust quaternion feature learning. In *Forty-third International Conference on Machine Learning*, 2026a. URL https://openreview.net/forum?id=xxifbmZe0b.

Zhiming Zhou, Yong He, Chaoxu Mu, Qiaoyun Wu, and Ajmal Saeed Mian. Spikenet: Sparse spike-driven mask vector transformer for energy-efficient and stable spiking point cloud processing. In *Forty-third International Conference on Machine Learning*, 2026b. URL https://openreview.net/forum?id=7BpcmBjQL0.

Zhiming Zhou, Yong He, Qiaoyun Wu, Chaoxu Mu, and Ajmal Saeed Mian. 3DSMT: A hybrid spiking mamba-transformer for point cloud analysis. In *The Fourteenth International Conference on Learning Representations*, 2026c.

A APPENDIX

The supplementary material presents the following sections to strengthen the main manuscript:

A1. Preliminaries.

A2. Theoretical Analysis.

A3. Dataset Introduction.

A4. Experiment Details.

A5. More Experiment.

A1 PRELIMINARIES

A1.1 INTEGRATE-AND-FIRE NEURON

Spiking neurons are the basic computational units of SNNs. Following 3DSMT, we adopt the Integrate-and-Fire (IF) neuron to convert membrane potentials into binary spike representations. At each time step t, the IF neuron integrates the input current, generates a spike according to the firing threshold, and updates its membrane potential as

$$u^t = V^{t-1} + I^t, \quad S^t = \Theta(u^t - V_{\text{th}}), \quad V^t = u^t(1 - S^t) + V_{\text{reset}} S^t, \tag{1,2,3}$$

where V^{t-1} denotes the membrane potential from the previous time step, I^t is the input current, u^t is the membrane potential after integration, and V_{reset} denotes the reset potential. $\Theta(\cdot)$ is the Heaviside step function:

$$\Theta(x) = \begin{cases} 1, & x \geq 0, \\ 0, & x < 0. \end{cases} \tag{4}$$

When u^t reaches the firing threshold V_{th}, the neuron emits a spike with $S^t = 1$ and resets its membrane potential to V_{reset}; otherwise, $S^t = 0$ and the membrane potential is preserved. This mechanism enables sparse information transmission in the following spike-driven modules.

A1.2 SPIKING SELF-ATTENTION

Self-attention models feature dependencies through the interactions among query, key, and value features. Given an input feature $\mathbf{X} \in \mathbb{R}^{T \times N \times D}$, vanilla self-attention first generates floating-point query, key, and value as

$$\mathbf{Q}_{\mathcal{F}} = \mathbf{X}\mathbf{W}_Q, \quad \mathbf{K}_{\mathcal{F}} = \mathbf{X}\mathbf{W}_K, \quad \mathbf{V}_{\mathcal{F}} = \mathbf{X}\mathbf{W}_V, \tag{5,6,7}$$

where $\mathbf{W}_Q$, $\mathbf{W}_K$, and $\mathbf{W}_V$ are learnable projection matrices. The standard self-attention is formulated as

$$\text{SA}(\mathbf{Q}_{\mathcal{F}}, \mathbf{K}_{\mathcal{F}}, \mathbf{V}_{\mathcal{F}}) = \text{Softmax}\left(\frac{\mathbf{Q}_{\mathcal{F}}\mathbf{K}_{\mathcal{F}}^{\text{T}}}{\sqrt{d}}\right)\mathbf{V}_{\mathcal{F}}, \tag{8}$$

where d denotes the feature dimension of each attention head. To adapt self-attention to spike-driven computation, Spikformer introduces Spiking Self-Attention (SSA), where query, key, and value are converted into binary spike representations:

$$\mathbf{Q} = \text{SN}_Q(\text{BN}(\mathbf{X}\mathbf{W}_Q)), \quad \mathbf{K} = \text{SN}_K(\text{BN}(\mathbf{X}\mathbf{W}_K)), \quad \mathbf{V} = \text{SN}_V(\text{BN}(\mathbf{X}\mathbf{W}_V)). \tag{9,10,11}$$

SSA removes the Softmax normalization and directly performs interaction among the spike-form $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$:

$$\text{SSA}'(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{SN}\big(\mathbf{Q}\mathbf{K}^{\text{T}}\mathbf{V} \cdot s\big), \tag{12}$$

where s is a scaling factor. The final output is obtained through a linear projection followed by normalization and spike activation:

$$\text{SSA}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{SN}\big(\text{BN}\big(\text{Linear}\big(\text{SSA}'(\mathbf{Q}, \mathbf{K}, \mathbf{V})\big)\big)\big), \tag{13}$$

Since $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ are binary spike features, SSA avoids Softmax and enables sparse spike-driven feature interaction.

A1.3 State Space Model and Mamba

State Space Models (SSMs) provide an efficient framework for sequence modeling by mapping an input sequence to an output sequence through a latent state. Given an input $\mathbf{x}(t)$, a continuous SSM is formulated as

$$\frac{d\mathbf{h}(t)}{dt} = \mathbf{A}\mathbf{h}(t) + \mathbf{B}\mathbf{x}(t), \quad \mathbf{y}(t) = \mathbf{C}\mathbf{h}(t) + \mathbf{D}\mathbf{x}(t), \tag{14,15}$$

where $\mathbf{h}(t)$ denotes the latent state, $\mathbf{A}$ is the state transition matrix, $\mathbf{B}$ and $\mathbf{C}$ control the input and output projections, respectively, and $\mathbf{D}$ denotes the direct residual connection. For discrete sequences, the continuous system can be discretized using the zero-order hold (ZOH) rule with a timescale parameter Δ:

$$\overline{\mathbf{A}} = \exp(\Delta\mathbf{A}), \quad \overline{\mathbf{B}} = (\Delta\mathbf{A})^{-1}\left(\exp(\Delta\mathbf{A}) - \mathbf{I}\right)\Delta\mathbf{B}. \tag{16,17}$$

The resulting discrete SSM is

$$\mathbf{h}_k = \overline{\mathbf{A}}\mathbf{h}_{k-1} + \overline{\mathbf{B}}\mathbf{x}_k, \quad \mathbf{y}_k = \mathbf{C}\mathbf{h}_k + \mathbf{D}\mathbf{x}_k. \tag{18,19}$$

Conventional SSMs use input-independent parameters, which limits their ability to selectively preserve or suppress information. Mamba introduces a selective SSM (S6) by making $\mathbf{B}$, $\mathbf{C}$, and Δ dependent on the input:

$$\mathbf{B}_k = s_B(\mathbf{x}_k), \quad \mathbf{C}_k = s_C(\mathbf{x}_k), \quad \Delta_k = \text{softplus}(\Delta_0 + s_\Delta(\mathbf{x}_k)), \tag{20,21,22}$$

where $s_B(\cdot)$, $s_C(\cdot)$, and $s_\Delta(\cdot)$ are learnable input projections. Accordingly, the discretized parameters become input-dependent:

$$\overline{\mathbf{A}}_k = \exp(\Delta_k\mathbf{A}), \quad \overline{\mathbf{B}}_k = (\Delta_k\mathbf{A})^{-1}\left(\exp(\Delta_k\mathbf{A}) - \mathbf{I}\right)\Delta_k\mathbf{B}_k. \tag{23,24}$$

The selective state transition is therefore expressed as

$$\mathbf{h}_k = \overline{\mathbf{A}}_k\mathbf{h}_{k-1} + \overline{\mathbf{B}}_k\mathbf{x}_k, \quad \mathbf{y}_k = \mathbf{C}_k\mathbf{h}_k + \mathbf{D}\mathbf{x}_k. \tag{25,26}$$

Through these input-dependent state-space parameters, Mamba selectively controls information propagation along the sequence and provides efficient global sequence modeling.

A2 Theoretical Analysis

This section further analyzes the main properties of 3DSMT++ following its feature-processing pipeline. We first discuss the token and parameter efficiency of MSPE, then provide a Laplacian interpretation of SDOA and analyze the sequence-order sensitivity of SPM. Finally, we discuss the computational complexity of the overall backbone.

A2.1 Token and Parameter Efficiency of MSPE

MSPE captures complementary geometry at multiple spatial scales without requiring a proportional increase in token length or embedding parameters. Given an input point cloud $\mathbf{P}$, progressive sampling produces fine, medium, and coarse point sets:

$$\mathbf{P}_f = \text{FPS}(\mathbf{P}, N_f), \quad \mathbf{P}_m = \text{FPS}(\mathbf{P}_f, N_m), \quad \mathbf{P}_c = \text{FPS}(\mathbf{P}_m, N_c). \tag{27,28,29}$$

Thus, the sampled point sets form a progressive hierarchy, $\mathbf{P}_c \subset \mathbf{P}_m \subset \mathbf{P}_f \subset \mathbf{P}$. Let $\phi_\theta(\cdot)$ denote the spiking patch embedding function shared by all scales. The multi-scale features are written as

$$\mathbf{F}_s = \phi_\theta(\mathbf{E}_s),\ s \in \{f, m, c\}, \quad \mathbf{F}_{\text{ms}} = \text{Concat}(\mathbf{F}_f, \mathbf{F}_m, \mathbf{F}_c), \quad L = L_f + L_m + L_c. \tag{30,31,32}$$

Therefore, MSPE allocates a prescribed total token budget L across different spatial resolutions instead of constructing a full-length sequence at every scale. The following backbone still processes a single sequence of length L, so multi-scale modeling mainly increases geometric diversity rather than sequence redundancy. Moreover, all scales share the same embedding parameters. Denoting the parameter set by θ, we have

$$\mathcal{P}_{\text{MSPE}} = |\theta|, \quad \mathcal{P}_{\text{ind}} = |\theta_f| + |\theta_m| + |\theta_c| = 3|\theta|, \tag{33,34}$$

where $\mathcal{P}_{\text{ind}}$ corresponds to three independent embedding networks with the same architecture. Parameter sharing thus introduces complementary multi-scale representations without a proportional increase in embedding parameters.

A2.2 Laplacian Interpretation of Spiking Distance-aware Offset Attention

The offset-attention mechanism in PCT Guo et al. (2021) interprets the difference between an input feature and its attention response as a discrete graph-Laplacian transformation. Let $\mathbf{F} \in \mathbb{R}^{L\times C}$ denote the input feature, $\mathbf{A} \in \mathbb{R}^{L\times L}$ the normalized attention matrix, and $\mathbf{V} = \mathbf{F}\mathbf{W}_v$ the value feature. The self-attention response is

$$\mathbf{F}_{\text{sa}} = \mathbf{A}\mathbf{V} = \mathbf{A}\mathbf{F}\mathbf{W}_v. \tag{35}$$

By absorbing the linear value projection into the subsequent feature transformation, the offset can be written as

$$\mathbf{F} - \mathbf{F}_{\text{sa}} \approx \mathbf{F} - \mathbf{A}\mathbf{F} = (\mathbf{I} - \mathbf{A})\mathbf{F}. \tag{36}$$

Because the rows of $\mathbf{A}$ are normalized, its corresponding degree matrix is $\mathbf{D} = \mathbf{I}$. Therefore,

$$(\mathbf{I} - \mathbf{A})\mathbf{F} = (\mathbf{D} - \mathbf{A})\mathbf{F} = \mathbf{L}_{\mathbf{A}}\mathbf{F}. \tag{37}$$

where $\mathbf{L}_{\mathbf{A}}$ is the graph Laplacian induced by the attention affinity. The offset consequently emphasizes feature variations relative to the attention-aggregated context.

SDOA preserves this Laplacian-inspired offset structure but replaces the dense attention response with a spike-driven vector operator. Given $\mathbf{F} = \mathbf{S}_2$, we define

$$\mathbf{\Phi}_\theta(\mathbf{F}) = \text{SN}\left((\mathbf{Q} - \mathbf{K}) \odot \mathbf{V}\right), \tag{38}$$

where $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$ are learned projections of $\mathbf{F}$. The SDOA offset is then

$$\mathbf{F}_{\text{off}} = \mathbf{F} - \mathbf{\Phi}_\theta(\mathbf{F}) = (\mathcal{I} - \mathbf{\Phi}_\theta)[\mathbf{F}]. \tag{39}$$

This operator difference generalizes the Laplacian form $(\mathbf{I}-\mathbf{A})\mathbf{F}$ to a data-dependent spiking feature operator. The equivalence concerns the offset structure: PCT uses an explicit normalized affinity matrix, whereas SDOA implements the attention response through channel-wise vector subtraction and multiplication. In this way, SDOA retains the feature-difference property of offset attention without explicitly constructing a dense attention matrix.

A2.3 Sequence-order Sensitivity of SPM

Point clouds are naturally unordered, whereas state-space models propagate information along a sequence. Therefore, different permutations of the same point tokens are generally not equivalent for SPM. Using the recurrent state-space formulation introduced in the preliminaries, we write

$$\mathbf{h}_k = \overline{\mathbf{A}}_k\mathbf{h}_{k-1} + \overline{\mathbf{B}}_k\mathbf{x}_k, \quad \mathbf{y}_k = \mathbf{C}_k\mathbf{h}_k. \tag{40,41}$$

Ignoring the initial-state term for clarity, the hidden state can be expanded as

$$\mathbf{h}_k = \sum_{j=1}^{k}\left(\prod_{r=j+1}^{k}\overline{\mathbf{A}}_r\right)\overline{\mathbf{B}}_j\mathbf{x}_j. \tag{42}$$

Thus, the contribution of $\mathbf{x}_j$ to a later state depends not only on the token itself, but also on the state transitions between positions j and k. For a permutation π, the corresponding state becomes

$$\mathbf{h}^{\pi}_k = \sum_{j=1}^{k}\left(\prod_{r=j+1}^{k}\overline{\mathbf{A}}_{\pi(r)}\right)\overline{\mathbf{B}}_{\pi(j)}\mathbf{x}_{\pi(j)}, \quad \mathcal{M}(\mathbf{P}_\pi\mathbf{X}) \neq \mathbf{P}_\pi\mathcal{M}(\mathbf{X}), \tag{43,44}$$

in general, where $\mathcal{M}(\cdot)$ denotes the state-space operator and $\mathbf{P}_\pi$ is a permutation matrix. Changing the token order therefore changes the state-transition path and the effective propagation relation among tokens. In 3DSMT++, SPM receives the token sequence after multi-scale embedding and local geometric modeling. NOrder keeps this sequence without introducing an additional ordering operation, whereas Shuffle randomly changes the state-transition path among the same point features. Such random permutations may disturb the feature organization already formed by MSPE and SDOA, making global state propagation less stable. This does not imply that the original point order is a canonical geometric ordering or that SPM is permutation invariant. Instead, NOrder simply avoids introducing an additional arbitrary permutation. This interpretation is consistent with Table 7, where NOrder performs better than Shuffle, while bidirectional NOrder provides only a marginal change.

A2.4 LINEAR COMPLEXITY OF 3DSMT++

We finally analyze the computational complexity with respect to the total token number L. Let C denote the feature dimension, K the neighborhood size in SDOA, K_c the causal convolution kernel size, D_s the state dimension of SPM, and M the number of Spiking Hybrid Blocks. We treat these architecture settings and the SNN timestep as constants when analyzing the asymptotic scaling with L. The cost of FPS and KNN search is implementation dependent, so we focus on the learnable feature computation after neighborhood construction. For the three main components, the computational costs can be summarized as

$$\begin{aligned} \mathcal{C}_{\text{MSPE}} &= \mathcal{O}\left(\sum_{s\in\{f,m,c\}} L_s K_s C\right), \\ \mathcal{C}_{\text{SDOA}} &= \mathcal{O}\big(LKC + LC^2\big), \\ \mathcal{C}_{\text{SPM}} &= \mathcal{O}\big(LC^2 + LK_cC + LCD_s\big). \end{aligned} \tag{45,46,47}$$

Since $L_f + L_m + L_c = L$ and the neighborhood sizes are fixed, $\mathcal{C}_{\text{MSPE}}$ grows linearly with L. In SDOA, the vector operation

$$(\mathbf{Q} - \mathbf{K}) \odot \mathbf{V} \tag{48}$$

requires only $\mathcal{O}(LC)$ computation and avoids the $\mathcal{O}(L^2C)$ pairwise interaction of dense dot-product attention. Combining SDOA and SPM, one Spiking Hybrid Block has

$$\mathcal{C}_{\text{SHB}} = \mathcal{O}\big(L(KC + C^2 + K_cC + CD_s)\big), \quad \mathcal{C}_{\text{Backbone}} = \mathcal{O}(M\mathcal{C}_{\text{SHB}}) = \mathcal{O}(L), \tag{49,50}$$

when M, K, K_c, C, and D_s are fixed with respect to the sequence length. Moreover, SDOA and SPM do not construct an $L \times L$ attention matrix. Their main token activations therefore scale linearly with the sequence length, while dense self-attention requires quadratic attention memory. These properties allow 3DSMT++ to combine multi-scale embedding, local geometric interaction, and global state-space modeling without introducing quadratic token complexity.

A2.5 REPRESENTATION ENRICHMENT OVER 3DSMT

3DSMT establishes a hybrid spike-driven framework by combining local offset attention with Mamba-based global modeling. However, its geometric representation is mainly constructed from a single spatial granularity, while its local and global interactions remain limited in explicitly exploiting spatial relevance and efficient state-space propagation. In contrast, 3DSMT++ does not simply increase model depth or capacity, but further strengthens its representation mechanism from three aspects: multi-scale geometric representation, distance-aware local propagation, and global state interaction tailored to the unordered nature of point clouds. First, MSPE constructs the multi-scale representation

$$F_{\text{ms}} = \text{Concat}(F_f, F_m, F_c), \tag{51}$$

where the fine-scale representation F_f is explicitly preserved as a subrepresentation of F_{ms}, while F_m and F_c further introduce geometric contexts at medium and coarse spatial scales. Therefore, the subsequent backbone can jointly exploit complementary local details and structural information across different spatial granularities, thereby extending the spatial range of point cloud geometric representation. Second, SDOA explicitly conditions local feature propagation on Euclidean distance. For two neighbors a and b of the same center point i, since

$$g(d) = \frac{1}{d + \epsilon} \tag{52}$$

is monotonically decreasing with respect to the distance, we have

$$\hat{d}_{ia} < \hat{d}_{ib} \quad \Rightarrow \quad w_{ia} > w_{ib}. \tag{53}$$

Thus, neighbors spatially closer to the center point receive larger propagation weights. This explicitly incorporates geometric relevance into local feature interaction, allowing different neighboring points to contribute differently to the center representation according to their spatial relations. Third, SPM further organizes global state interaction according to the unordered nature of point clouds. Since point clouds do not possess fixed sequential semantics, effective global state propagation is

closely related not only to state-space modeling itself, but also to the organization of local geometric features established by preceding modules. SPM therefore directly follows the feature organization produced by MSPE and SDOA without introducing additional random token reordering. Meanwhile, SPM structurally separates state evolution from local context encoding: the state-transition branch A directly controls state evolution, whereas X_0, B_0, and C_0 are first processed by causal convolution and spike activation,

$$X_s = \mathrm{SN}(\mathrm{CConv}(X_0)), B_s = \mathrm{SN}(\mathrm{CConv}(B_0)), C_s = \mathrm{SN}(\mathrm{CConv}(C_0)), \tag{54,55,56}$$

followed by

$$Y = \mathcal{M}_2(X_s; A, B_s, C_s), \qquad O = X + \mathrm{Linear}_o(\mathrm{SN}(Y) \odot dt). \tag{57}$$

In this way, the local geometric organization extracted by preceding modules can be continuously propagated into global modeling, while the spike-based modulation branch dt selectively regulates the resulting state responses, forming a more coherent local-to-global information interaction process. This design does not rely on constructing an additional artificial point ordering, but instead preserves the continuity of the established geometric feature organization during state propagation. Overall, MSPE, SDOA, and SPM further enhance 3DSMT from the perspectives of geometric representation space, local feature propagation rules, and global state interaction mechanisms, respectively. Therefore, 3DSMT++ does not merely increase network depth or parameter capacity, but systematically changes how multi-scale geometric information, local spatial relations, and global dependencies are represented, propagated, and integrated in the spike-driven point cloud network.

A3 Dataset Introduction

We evaluate 3DSMT++ on four representative point cloud benchmarks covering object classification, few-shot classification, part segmentation, and large-scale indoor semantic segmentation. ModelNet40 and ShapeNetPart are synthetic datasets constructed from clean CAD models, while ScanObjectNN and S3DIS contain real-world scanned point clouds with more complex geometry, noise, occlusion, and background clutter.

ScanObjectNN. ScanObjectNN (Uy et al., 2019) is a real-world object-level point cloud dataset collected from indoor scene scans. It contains 2,902 unique object instances from 15 categories and is more challenging than synthetic CAD datasets because scanned objects may be incomplete, noisy, and mixed with background points. We evaluate object classification on three commonly used variants: OBJ_ONLY, OBJ_BG, and PB_T50_RS. OBJ_ONLY contains only the segmented object points, while OBJ_BG retains surrounding background points within the object bounding box. PB_T50_RS further introduces bounding-box perturbations with translation, rotation, and scaling, forming the most challenging setting. In the commonly used processed split, OBJ_ONLY and OBJ_BG contain 2,309 training and 581 test samples, while PB_T50_RS expands the data through perturbations to 11,416 training and 2,882 test samples. The classification task requires predicting one object category for each input point cloud.

ModelNet40. ModelNet40 (Wu et al., 2015) is a widely used synthetic benchmark for 3D shape classification. It consists of 12,311 clean CAD models from 40 object categories, including chairs, tables, airplanes, and other common objects. The standard split contains 9,843 models for training and 2,468 models for testing. Compared with ScanObjectNN, ModelNet40 contains complete and relatively clean object geometry without real-world scanning noise or background clutter. The task is object-level classification, where each point cloud is assigned to one of the 40 semantic categories.

Few-shot Classification on ModelNet40. We further use ModelNet40 for few-shot classification to evaluate representation generalization when only a small number of labeled samples are available. Following the n-way m-shot protocol, n object categories are randomly selected and only m labeled samples from each category are used for training. We evaluate four settings with $n \in \{5, 10\}$ and $m \in \{10, 20\}$, corresponding to 5-way 10-shot, 5-way 20-shot, 10-way 10-shot, and 10-way 20-shot classification. For each setting, 20 additional samples per selected category are used for testing, and the mean accuracy and standard deviation are reported over 10 independent runs.

ShapeNetPart. ShapeNetPart (Yi et al., 2016) is a synthetic object-level dataset for fine-grained part segmentation. It contains 16,881 CAD shapes from 16 object categories, with 50 part labels in total. Each object category contains two to six semantic parts, such as the wings and body of an

Table A1: Training configurations for pre-training and different downstream tasks.

Parameter	Pre-training	ModelNet40	ScanObjectNN	Few-shot	ShapeNetPart	S3DIS
Optimizer	AdamW	AdamW	AdamW	AdamW	AdamW	AdamW
Learning Rate	1×10^{-3}	5×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}	1×10^{-2}
Weight Decay	5×10^{-2}	5×10^{-2}	5×10^{-2}	5×10^{-2}	5×10^{-2}	1×10^{-4}
Batch Size	192	24	24	24	32	16
Training Epochs	300	300	300	300	350	100
Initial Point Number	2048	1024	2048	1024	2048	24000

airplane or the seat and legs of a chair. Following the commonly used split, 14,006 shapes are used for training and 2,874 shapes for testing. Unlike object classification, part segmentation requires predicting a semantic part label for every point, providing a direct evaluation of fine-grained local geometric representation.

S3DIS. S3DIS (Armeni et al., 2016) is a large-scale real-world indoor point cloud dataset collected from three buildings. It contains six large indoor areas with 271 rooms and approximately 273 million points. Each point is associated with 3D coordinates, RGB information, and one of 13 semantic labels covering structural elements, furniture, and other indoor objects. We follow the widely used Area 5 protocol, where Area 5 is used for testing and the remaining five areas are used for training. The semantic segmentation task requires assigning a semantic category to every point in a complete indoor scene, making S3DIS considerably more challenging than object-level datasets in terms of scene scale, geometric complexity, and point density.

A4 Experiment Details

Classification and Pre-training. For object classification and few-shot classification, we use AdamW with an initial learning rate of 5×10^{-4} and a weight decay of 5×10^{-2}. The models are trained for 300 epochs with a batch size of 24. We use 1,024 input points for ModelNet40 and few-shot classification, and 2,048 points for ScanObjectNN. For masked pre-training, the model is trained for 300 epochs using AdamW with an initial learning rate of 1×10^{-3}, a weight decay of 5×10^{-2}, and a batch size of 192. Each pre-training sample contains 2,048 input points, and random masking with a ratio of 0.6 is applied.

Point Cloud Segmentation. For part segmentation on ShapeNetPart, each object is represented by 2,048 points. The model is optimized using AdamW with an initial learning rate of 5×10^{-4} and a weight decay of 5×10^{-2}, with a batch size of 32 for 350 epochs. For large-scale indoor semantic segmentation on S3DIS, the model is trained for 100 epochs using AdamW with an initial learning rate of 1×10^{-2}, a weight decay of 1×10^{-4}, and a batch size of 16. Each training sample starts from 24,000 input points. The detailed configurations for all tasks are summarized in Table A1.

A5 More Experiment

Effect of Spiking Neuron. The choice of spiking neuron directly affects spike generation and temporal feature propagation. We therefore compare IF and LIF neurons on ModelNet40. As shown in Table A2, the IF neuron achieves 95.5% OA and 92.2% mAcc, outperforming LIF by 0.4% and 0.3%, respectively. Point clouds mainly contain static spatial geometry, while the temporal dimension in the SNN is introduced by spike-based processing rather than inherent temporal dynamics in the input. Therefore, the membrane leakage in LIF may be less necessary for point cloud representation, whereas IF can preserve the accumulated geometric responses until firing with simpler dynamics. This result indicates that the simple integrate-and-fire mechanism is sufficient and more suitable for effective point cloud representation in 3DSMT++. We therefore adopt the IF neuron in all experiments.

Effect of Neuron Threshold. The firing threshold controls when a neuron emits spikes and therefore influences the sparsity and information transmission of spike features. Table A3 compares two threshold settings. Increasing V_{th} from 0.5 to 1.0 improves OA from 94.8% to 95.5% and mAcc from 91.5% to 92.2%. A lower threshold causes neurons to fire more easily and may introduce less

Table A2: Effect of different spiking neurons on ModelNet40.

Neuron	OA(%)	mAcc(%)
LIF	95.1	91.9
IF	**95.5**	**92.2**

Table A3: Effect of neuron threshold on ModelNet40.

V_{th}	OA(%)	mAcc(%)
0.5	94.8	91.5
1.0	**95.5**	**92.2**

Table A4: Effect of the number of SHBs on ModelNet40.

M	OA(%)	mAcc(%)
4	93.9	90.2
8	94.4	90.8
12	**95.5**	**92.2**
16	94.5	91.4

Table A5: Effect of **K** neighborhood size in SDOA on ModelNet40.

K	OA(%)	mAcc(%)
2	93.7	90.3
4	**95.5**	**92.2**
6	94.9	91.6
8	94.3	91.0

selective spike responses, while $V_{\text{th}} = 1.0$ provides a better balance between spike activation and feature representation. We therefore set $V_{\text{th}} = 1.0$ throughout the experiments.

Effect of SHB Number. The number of Spiking Hybrid Blocks determines the depth of local-global feature interaction in the backbone. We vary the block number M from 4 to 16, as reported in Table A4. Increasing M from 4 to 12 steadily improves performance, with OA increasing from 93.9% to 95.5% and mAcc from 90.2% to 92.2%. This shows that deeper stacking allows SDOA and SPM to progressively refine local geometric features and global interactions. However, further increasing M to 16 decreases OA and mAcc to 94.5% and 91.4%, respectively, suggesting that excessive depth may introduce redundant feature transformations without additional representation gains. Therefore, we use $M = 12$ as the default setting.

Effect of Neighborhood Size in SDOA. The neighborhood size K determines the spatial range used for local geometric interaction in SDOA. As shown in Table A5, increasing K from 2 to 4 substantially improves OA from 93.7% to 95.5% and mAcc from 90.3% to 92.2%, indicating that an overly small neighborhood provides insufficient local context. When K is further increased to 6 and 8, performance gradually decreases to 94.9%/91.6% and 94.3%/91.0%, respectively. A larger neighborhood may introduce less relevant points into local propagation and weaken the emphasis on fine-grained local structures. These results show that $K = 4$ provides the best balance between local geometric context and neighborhood relevance, and is therefore used in SDOA.

Comparison with 3DSMT. To further clarify the difference between 3DSMT++ and 3DSMT, we directly compare their full models and replace each enhanced module in 3DSMT++ with its corresponding module from 3DSMT. As shown in Table A6, the full SNN version of 3DSMT++ outperforms 3DSMT by 3.1%, 2.9%, and 2.5% on BG, ONLY, and RS, respectively, and improves OA by 0.8% on ModelNet40. Similar gains are also observed in the ANN counterparts, showing that the improvement mainly comes from stronger point cloud representation rather than the spiking mechanism alone. More importantly, replacing MSPE with SPE decreases OA by 0.9%, 1.1%, and 1.2% on the three ScanObjectNN variants and by 0.6% on ModelNet40. Replacing SDOA with SLOA or SPM with SMB also consistently reduces performance across the two benchmarks. These module-level comparisons demonstrate that the enhanced designs in 3DSMT++ capture richer geometric information than their counterparts in 3DSMT. Therefore, 3DSMT++ is not a simple incremental extension of 3DSMT, but further strengthens point cloud representation through more effective geometric feature extraction and interaction. The consistent gains from these module replacements further show that the improvement is achieved through stronger geometric modeling throughout the representation pipeline, rather than by a single isolated modification. This also explains the clear performance advantage of 3DSMT++ over 3DSMT under both ANN and SNN settings.

Table A6: Comparison between 3DSMT++ and 3DSMT with module-level replacements on ScanObjectNN and ModelNet40. MSPE→SPE, SDOA→SLOA, and SPM→SMB denote replacing the enhanced modules in 3DSMT++ with their corresponding modules from 3DSMT. OA and mAcc are reported in %, and Energy in millijoule (mJ).

Method	Type	ScanObjectNN				ModelNet40		
		BG (OA) ↑	ONLY (OA) ↑	RS (OA) ↑	Energy ↓	OA↑	mAcc↑	Energy↓
Full:3DSMT	ANN	91.2(↓ 3.6)	90.4(↓ 2.8)	91.0(↓ 2.4)	36.3	94.9(↓ 0.7)	92.0(↓ 0.4)	36.3
Full:3DSMT++	ANN	94.8	93.2	93.4	42.1	95.6	92.4	42.1
Full:3DSMT	SNN	90.8(↓ 3.1)	89.7(↓ 2.9)	90.4(↓ 2.5)	4.9	94.7(↓ 0.8)	91.8(↓ 0.4)	4.3
Full:3DSMT++	SNN	**93.9**	**92.6**	**92.9**	6.2	**95.5**	**92.2**	5.5
MSPE→SPE	SNN	93.0(↓ 0.9)	91.5(↓ 1.1)	91.7(↓ 1.2)	4.8	94.9(↓ 0.6)	91.9(↓ 0.3)	4.2
SDOA→SLOA	SNN	93.3(↓ 0.6)	92.2(↓ 0.4)	92.5(↓ 0.4)	5.5	95.1(↓ 0.4)	92.0(↓ 0.2)	4.9
SPM→SMB	SNN	93.1(↓ 0.8)	91.9(↓ 0.7)	92.3(↓ 0.6)	5.3	95.0(↓ 0.5)	92.0(↓ 0.2)	4.8

A6 LLM USAGE STATEMENT

In accordance with the relevant ICLR policy regarding the use of Large Language Models (LLMs), the authors hereby declare that the core academic content of this research, including the formulation of research ideas, methodology design, experimental implementation and data analysis, derivation of results, and formulation of conclusions, was independently completed by the authors.

During the paper writing process, AI tools were used only for language polishing and expression refinement of selected parts of the manuscript to improve grammatical accuracy, readability, and fluency. AI tools were not involved in the conception of the research, methodology design, experimental execution, result analysis, or formulation of scientific conclusions. All AI-assisted revisions were further reviewed and verified by the authors.

The authors take full responsibility for the authenticity, accuracy, originality, and academic integrity of all content in this paper. The use of AI tools was strictly limited to language-level assistance and did not affect the independence or scientific integrity of the research.

A7 ETHICS STATEMENT

We confirm that all authors have read and comply with the ICLR Code of Ethics. Our work does not involve human subjects, sensitive data collection or release, potentially harmful methodologies or applications, conflicts of interest, discrimination/bias/fairness concerns, privacy/security issues, legal compliance challenges, or research integrity matters that require explicit addressing.

A8 REPRODUCIBILITY STATEMENT

To ensure the reproducibility of our results, we will make our implementation code available as supplementary material. The datasets used (e.g., ModelNet40, ScanObjectNN, ShapeNetPart for point cloud tasks) are publicly accessible, with relevant details provided in the main text. Key experimental settings (including hyperparameters like SNN threshold and timestep optimized in our analysis) and procedures are thoroughly documented in the manuscript. Any theoretical derivations supporting our work are included in the appendix.