GEOMETRY-GUIDED CONSERVATIVE SOFT ORDERING FOR POINT CLOUD UNDERSTANDING

Anonymous authors
Paper under double-blind review

ABSTRACT

3D point clouds are inherently unordered, whereas state space models (SSMs) process ordered sequences, making token ordering critical for point cloud understanding. Existing methods derive fixed hard orders from geometry that cannot be optimized by the task loss, forcing SSMs to accommodate suboptimal ordering and potentially weakening the learned representations. To address this limitation, we propose Geometry-guided Conservative Soft Ordering (GCSO), which replaces one-to-one hard assignment with trainable conservative soft ordering based on geometry prior. Specifically, GCSO uses current token features and coordinates to predict a local offset for each scan position in a geometric base order. Then the offset centers the weights over neighboring tokens, whose weighted feature mixture forms the soft ordering. Since the weights vary continuously, the task loss can optimize the offset and the resulting feature-adaptive soft ordering end to end. Moreover, GCSO organizes the weights into a nonnegative doubly stochastic routing matrix, ensuring that each scan position receives one unit of assignment while each source token contributes one unit of participation. Experiments on shape classification and part segmentation support the effectiveness of GCSO for point cloud understanding.

1 INTRODUCTION

Point cloud understanding aims to learn effective representations from inherently unordered 3D points and has become fundamental to a wide range of 3D vision tasks. Hierarchical networks and dynamic graph methods capture local geometric structures through neighborhood aggregation (Zhang et al., 2022; Afham et al., 2022), while point transformers further model contextual interactions with attention mechanisms (Yu et al., 2022; Pang et al., 2022). More recently, state space models (SSMs) have emerged as an efficient alternative for long-range modeling due to their linear complexity with respect to sequence length (Liang et al., 2024; Wang et al., 2025). However, point clouds are inherently unordered, whereas SSMs operate on ordered sequences, requiring point tokens to be serialized before sequential modeling. Therefore, the resulting order governs sequential geometric interactions, making token ordering critical to SSM-based point cloud representation learning.

Existing SSM-based point cloud methods typically construct scan orders based on geometric structures. For example, PointMamba uses complementary space-filling curves (Liang et al., 2024), PCM employs coordinate-axis permutations (Zhang et al., 2025a), and SI-Mamba derives traversals from graph spectral information (Bahri et al., 2025). As illustrated in Figure 1(a–c), these rules provide geometric priors but yield hard token orders that cannot be optimized by the task loss. A natural extension is to predict a new hard order from the current representation, but converting predicted offsets into one-to-one token assignments is discrete, as shown in Figure 1(d). Task gradients therefore cannot revise the hard assignment, forcing the state space module to compensate for locally suboptimal orders during feature propagation and weakening the resulting representation.

Motivated by this observation, we propose Geometry-guided Conservative Soft Ordering (GCSO), as illustrated in Figure 1(e). Rather than assigning each scan position to a fixed source token, GCSO softly aggregates nearby tokens according to a geometric base order. Specifically, the current features and coordinates predict a local offset for each scan position. Relative distances between the shifted position and neighboring tokens are then converted into normalized weights, whose weighted

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.

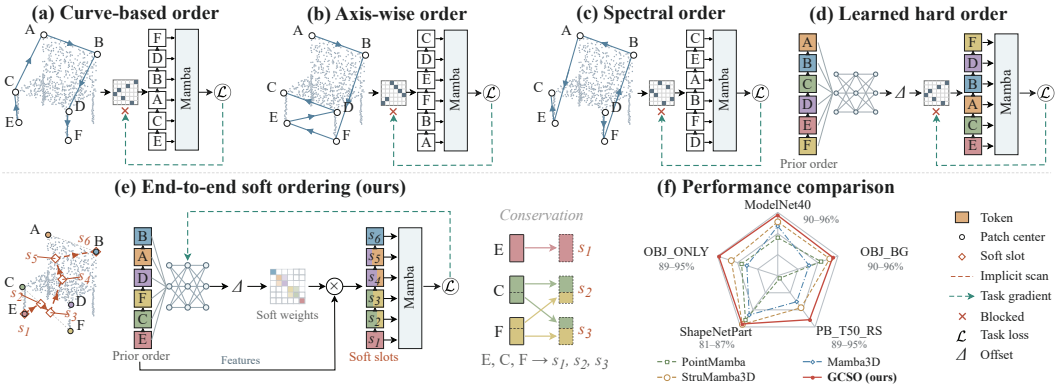

Figure 1: **From geometric hard orders to end-to-end soft ordering. (a–c)** Geometric rules produce fixed hard sequences with token-to-position assignments that task gradients cannot update. **(d)** A learned hard order requires a discrete token-to-position assignment that blocks task gradients. **(e)** GCSO converts feature- and geometry-conditioned offsets into continuous reading weights that mix source tokens into soft slots. The differentiable route allows the task loss to update the offset predictor, while conservation maintains unit total weight for every source token and scan position. **(f)** Classification OA and ShapeNetPart instance-averaged mIoU after self-supervised pre-training.

feature sum forms the Mamba input. Since these weights vary continuously with the predicted offsets, the task loss can directly optimize the offsets. Moreover, to preserve the assignment budget of discrete ordering, GCSO organizes the weights into a nonnegative routing matrix with unit row and column sums. As visualized by the conservation diagram in Figure 1(e), each scan position receives one unit of assignment, while each source token contributes one unit of participation. Overall, GCSO transforms geometric hard ordering into a trainable, conservative, and locally soft ordering.

The complete GCSO pipeline uses the resulting routing weights to mix source features into soft slots and back-project scan outputs to their original patch indices to preserve feature alignment for subsequent modules. Moreover, to capture local relationships missed by a single traversal, GCSO fuses the routed outputs from complementary geometric base orders before the feature update. Empirically, we evaluate GCSO on ModelNet40 (Wu et al., 2015) and ScanObjectNN (Uy et al., 2019). GCSO achieves the strongest listed results on all three ScanObjectNN variants in both settings; after pre-training, it also reaches 95.7% OA on ModelNet40 without voting and 96.0% with voting (see Table 1 and Figure 1(f)). Our contributions are:

- We analyze existing geometric ordering methods and identify a shared limitation, i.e., their hard token-to-position assignments cannot be updated by the task loss, forcing the state-space module to compensate for locally suboptimal scan organization.
- We propose GCSO, which replaces fixed hard assignments with trainable local soft ordering. Features and coordinates predict an offset position, and a continuous function converts the relative distances from this offset position to nearby tokens into weights that can be optimized end to end.
- Classification results on ModelNet40 and ScanObjectNN after self-supervised pre-training support the proposed method for point cloud understanding.

2 Related Work

2.1 Point Cloud Representation Learning

Point cloud networks encode unordered geometry through symmetric aggregation and hierarchical neighborhoods (Qi et al., 2017a;b), dynamic graphs, learned local transformations, and spatial kernels (Wang et al., 2019; Li et al., 2018; Thomas et al., 2019). Residual MLPs, geometric normalization, and scaling strategies provide efficient backbones (Ma et al., 2022; Qian et al., 2022), while attention models capture content-dependent interactions through global or local attention, grouped vector attention, and partition-based pooling (Guo et al., 2021; Zhao et al., 2021;

Wu et al., 2022). Octree windows and serialized neighborhoods further improve scalability by organizing spatial groups (Wang, 2023; Wu et al., 2024). Self-supervised methods learn transferable representations through masked token prediction, patch reconstruction, and multi-scale masking (Yu et al., 2022; Pang et al., 2022; Zhang et al., 2022), or through autoregressive prediction over ordered patches (Chen et al., 2023).

State space models provide an efficient alternative for long-range sequence modeling. S4 introduces structured parameterization, Mamba adds input-dependent selection, and Mamba-2 develops state space duality (Gu et al., 2022; Gu & Dao, 2024; Dao & Gu, 2024). Vision adaptations use bidirectional and multidirectional scans to process non-causal inputs (Zhu et al., 2024; Liu et al., 2024). Applying the same sequence modeling capability to point clouds requires an explicit serialization step that converts unordered local tokens into scan sequences. Point-cloud state space modeling therefore depends on both sequence construction before the scan and index restoration afterward. GCSO makes the sequence construction trainable within patch-based encoder layers while preserving the patch indices required by subsequent spatial aggregation.

2.2 Point Cloud Serialization and Differentiable Assignment

Point cloud serialization uses geometric rules to organize spatial relationships before state propagation. Existing models construct complementary sequences from space-filling curves or coordinate traversals and identify them with order indicators, prompts, or positional encodings (Liang et al., 2024; Zhang et al., 2025a). SI-Mamba derives traversals from patch-graph Laplacian spectra (Bahri et al., 2025). Other designs improve the surrounding local operators and state organization: Mamba3D combines Local Norm Pooling with state space modeling (Han et al., 2024); recent models introduce spatial state proxies, modified updates, sequence expansion and merging, or grouped state spaces (Wang et al., 2025; Qu et al., 2026); DM3D develops a deformable Mamba block that adapts spatial resampling with learned offsets (Liu et al., 2026); and local geometric enhancement, non-causal convolution, and node state increments further strengthen feature interaction (Zhou et al., 2025; Li et al., 2026; Xu et al., 2026). These methods broaden point-cloud state space modeling through prescribed sequence processing or adaptive sequence construction. GCSO focuses on conservative local soft assignment around a geometric base order, retaining explicit participation budgets for both source tokens and scan positions.

Differentiable sorting replaces discrete assignments with continuous relaxations that support gradient-based learning. Matrix balancing characterizes alternating row and column scaling (Sinkhorn & Knopp, 1967; Knight, 2008), while entropy-regularized transport and Gumbel-Sinkhorn support efficient transport computation and latent matching (Cuturi, 2013; Mena et al., 2018). Related sorting and ranking operators use row-stochastic relaxations, permutahedron projections, or optimal transport (Grover et al., 2019; Prillo & Eisenschlos, 2020; Blondel et al., 2020; Cuturi et al., 2019). Sparse Sinkhorn Attention learns latent reorderings for local-window context, and RPM-Net uses differentiable Sinkhorn matching for point cloud correspondence (Tay et al., 2020; Yew & Lee, 2020). Point-cloud scan routing poses a more specific problem than generic differentiable sorting. The sequence should adapt to the task without discarding its geometric reference or the token participation required by downstream spatial processing. GCSO formulates task-adaptive scan routing as conservative local soft ordering around a geometric base sequence.

3 Method

GCSO implements a differentiable and conservative relaxation of task-driven local permutations around a geometric base sequence. Specifically, we first formulate the hard ordering and relax it into a continuous soft assignment, followed by the overall architecture and training objective.

3.1 Preliminaries

Let $H \in \mathbb{R}^{G\times d}$ denote the d-dimensional point cloud features of G patches, and $C \in \mathbb{R}^{G\times 3}$ denote the centers of G patches. An assignment matrix $P \in \mathbb{R}^{G\times G}$ maps the patches to scan positions, while $\mathcal{M}$ denotes the state-space sequence module. The resulting scan branch F is defined as $F(H) = P^\top \mathcal{M}(PH)$. Here, PH forms the scan sequence, $\mathcal{M}(PH)$ propagates information along the sequence, and $P^\top$ maps the scanned features back to their original patch indices.

For a hard order, the assignment matrix P^{hard} is represented by

$$P^{\text{hard}} \in \mathcal{P} = \left\{ P \in \{0,1\}^{G\times G} : P\mathbf{1} = P^\top \mathbf{1} = \mathbf{1} \right\}. \tag{1}$$

The binary entries make P^{hard} discrete, while the unit row and column sums make it one-to-one. Each scan position therefore reads one source patch, and each source patch appears once. Geometric serialization fixes P^{hard} before task optimization, so the task loss cannot adapt the hard order.

To make the order task-driven, a natural approach is to infer a hard permutation from the current representation. A predictor parameterized by θ therefore produces assignment scores $S_\theta(H, C)$. The highest-scoring valid hard order is

$$P_\theta^{\text{hard}} = \arg\max_{P\in\mathcal{P}} \langle S_\theta(H,C), P \rangle_F . \tag{2}$$

The maximization enforces a valid one-to-one assignment. However, the selected permutation is piecewise constant with respect to the assignment scores, so $\partial P_\theta^{\text{hard}} / \partial S_\theta = 0$ almost everywhere and the task loss provides no gradient through the routing path. Thus, back-propagation cannot optimize the learned hard order through the task loss. Appendix A.1 provides the formal argument.

Our Geometry-guided Conservative Soft Ordering (GCSO) alleviates this discrete gradient barrier by replacing the hard order with a continuous soft assignment:

$$P_\theta^{\text{GCSO}}(H,C) \in \left\{ P \in \mathbb{R}_+^{G\times G} : P\mathbf{1} = P^\top \mathbf{1} = \mathbf{1} \right\}. \tag{3}$$

The soft assignment GCSO removes the binary constraint while retaining the unit row and column sums. These constraints preserve the reading budget of each scan position and the participation budget of each source patch, while the continuous entries allow task gradients to optimize the assignment. The following subsections describe how GCSO constructs and applies the layer-specific soft assignment P_ℓ. Appendix A.1 derives the complete gradient path.

3.2 Overall Architecture

As shown in Figure 2(a), given the input point cloud $\mathcal{X} \in \mathbb{R}^{N\times 3}$ containing N points, farthest point sampling (FPS) selects G patch centers $C \in \mathbb{R}^{G\times 3}$, and K-nearest neighbor (KNN) grouping constructs G center-relative patches, each containing K points. A shared Patch Encoder maps these patches to the initial features $F_0 \in \mathbb{R}^{G\times d}$.

Each of the L encoder layers, indexed by $\ell \in \{1, \ldots, L\}$, applies local aggregation $\mathcal{A}_\ell$ followed by conservative routed scan $\mathcal{S}_\ell$. As shown in Figure 2(b), $\mathcal{A}_\ell$ normalizes feature differences using a local root mean square scale, applies weighted pooling, and uses a multilayer perceptron (MLP) to aggregate information within neighborhoods defined by the patch centers (Han et al., 2024). The subsequent scan branch constructs routing from the aggregated features and patch centers.

To integrate the aggregation and scan branches, each encoder layer incorporates positional embeddings $E = \text{MLP}_{\text{pos}}(C)$ and residual connections. LayerNorm, denoted as $\text{LN}_{\ell,1}$ and $\text{LN}_{\ell,2}$, is also applied to the aggregation and scan branches, respectively, to stabilize the feature representations. Specifically, we denote the input and output features of layer ℓ as $F_{\ell-1}$ and F_ℓ. The position-enhanced features are denoted by V_ℓ, the aggregation-branch output by Z_ℓ, and the normalized scan input by H_ℓ. Accordingly, the output features F_ℓ is computed as follows:

$$\begin{aligned} V_\ell &= F_{\ell-1} + E, & Z_\ell &= V_\ell + \mathcal{A}_\ell(C, \text{LN}_{\ell,1}(V_\ell)), \\ H_\ell &= \text{LN}_{\ell,2}(Z_\ell), & F_\ell &= Z_\ell + \mathcal{S}_\ell(H_\ell, C). \end{aligned} \tag{4}$$

All layers reuse the same patch centers C, providing a consistent spatial reference for local aggregation and routed scanning. The final output F_L is passed to a classification or segmentation head.

3.3 Geometry-Guided Conservative Soft Ordering

Geometric prior and local parameterization. We adopt Morton ordering as the geometric prior and denote its permutation matrix by Π. Applying Π gives the ordered coordinates $\hat{C} = \Pi C$ and features $\hat{H}_\ell = \Pi H_\ell$. The corresponding coordinate and feature at scan position i are $\hat{c}_i = \hat{C}[i,:] \in \mathbb{R}^3$ and $\hat{h}_{\ell,i} = \hat{H}_\ell[i,:] \in \mathbb{R}^d$, respectively.

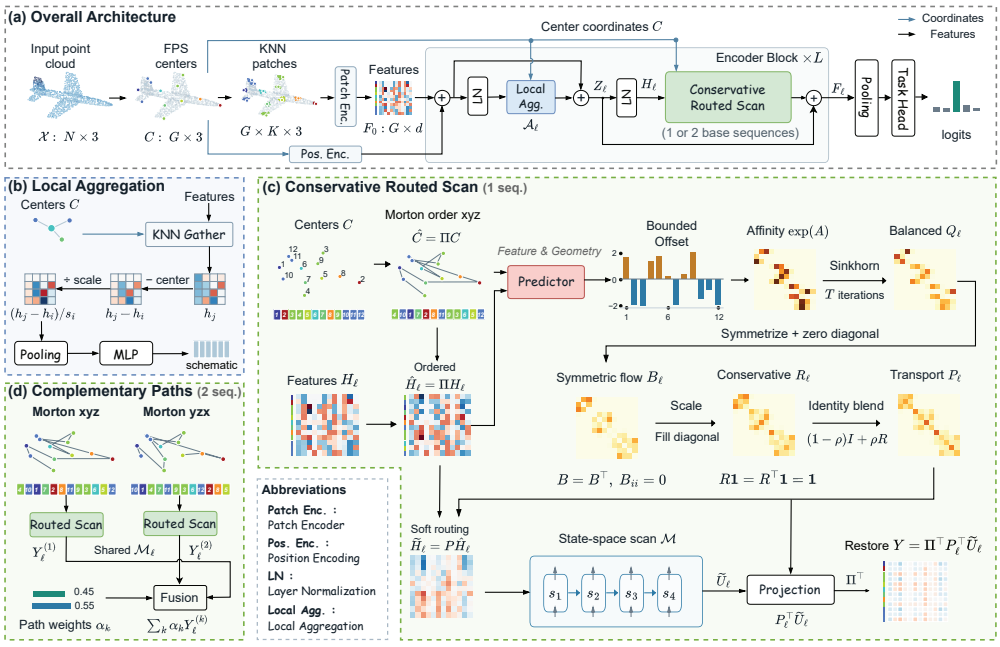

Figure 2: **Method overview.** **(a)** The architecture updates patch features through local aggregation and conservative routed scan with one or two base sequences. **(b)** Local aggregation combines features within neighborhoods defined by patch centers. **(c)** For one base sequence, conservative routed scan predicts local offsets from features and coordinates, constructs a doubly stochastic routing matrix for adaptive interaction with conserved patch participation weights, and projects scan outputs back to the original patch indices. **(d)** For two base sequences , the routed-scan module in (a) fuses the restored outputs of complementary Morton paths.

To construct the soft routing matrix P_ℓ, we first predict a local offset for each scan position from its feature $\hat{h}_\ell$ and standardized coordinate $\hat{c}$. As shown in Figure 2(c), the predictor f_θ maps these two inputs into a shared space, where their embeddings are summed and subsequently processed by a SiLU activation followed by a linear layer. The output of the predictor f_θ is bounded by $\tanh$, yielding a local offset $\delta_{\ell,i}$ for each scan position i in layer ℓ. We then construct the affinity matrix A_ℓ by measuring the relative distance between each candidate position j and the shifted position $i+\delta_{\ell,i}$, , as follows:

$$\delta_{\ell,i} = a\tanh f_\theta\left(\hat{h}_{\ell,i}, \mathrm{Std}(\hat{c}_i)\right), \qquad A_{\ell,ij} = \begin{cases} -(j-i-\delta_{\ell,i})^2/(2\sigma^2\tau), & |i-j|\le w, \\ -\infty, & \text{otherwise.} \end{cases} \tag{5}$$

Here, Std is coordinate standardization over patch centers. w denotes the radius of the local window, and $a \le w$ constrains each predicted offset within the local window. Parameters σ and τ control kernel width and temperature, respectively. Thus, $\delta_{\ell,i}$ shifts the preferred in-window position to obtain the affinity, whereas out-of-window positions get zero affinity.

Conservative routing matrix. Following the formulation in Section 3.1, we constrain the routing affinity matrix A_ℓ to have unit row and column sums. We first apply $T = 7$ Sinkhorn iterations (Sinkhorn & Knopp, 1967) to A_ℓ by alternately normalizing its rows and columns in log space and obtain $Q_\ell = \exp(\mathrm{Sinkhorn}_T(A_\ell))$. Q_ℓ provides an approximately balanced routing affinity. We then correct its remaining marginal errors and control the deviation from the base sequence:

$$\begin{aligned} B_\ell &= \tfrac{1}{2}(Q_\ell + Q_\ell^\top)\odot(\mathbf{1}\mathbf{1}^\top - I), & \kappa_\ell &= \max\big(1, \max_i (B_\ell\mathbf{1})_i\big), \\ R_\ell &= B_\ell/\kappa_\ell + \mathrm{Diag}(\mathbf{1} - B_\ell\mathbf{1}/\kappa_\ell), & P_\ell &= (1-\rho)I + \rho R_\ell. \end{aligned} \tag{6}$$

Here, I is the identity matrix, $\mathbf{1}$ is the all-ones column vector, $\odot$ denotes element-wise multiplication, and Diag constructs a diagonal matrix from a vector. The matrix B_ℓ extracts the symmetric off-diagonal routing weights, while κ_ℓ bounds the total outgoing weight. The diagonal term of R_ℓ

assigns the remaining weight to the original scan position, resulting in a nonnegative, symmetric, and doubly stochastic routing matrix. Finally, P_ℓ interpolates between R_ℓ and the identity matrix I through the learnable coefficient $\rho \in (0, 1)$, whose small initial value keeps the routing close to the geometric base sequence.

The resulting P_ℓ preserves the reading and participation budgets by maintaining unit row and column sums, while allowing task-driven interactions between nearby scan positions through continuous routing weights. This conservative formulation prevents excessive reuse or omission of individual tokens during routing, while retaining the flexibility to adapt the scan sequence to the downstream task. Appendix A provides a formal proof of these properties and discusses their scope.

3.4 Routed Scan and Complementary Path Fusion

Scanning and back-projection. As shown in Figure 2(c), after obtaining the routing matrix P_ℓ, we use it to mix the features before sequence modeling. The Mamba module $\mathcal{M}_\ell$ (Gu & Dao, 2024) then propagates the mixed features $\widetilde{H}_\ell$ along the routed sequence. For residual updates and spatial aggregation, the scan output is distributed back using the same participation weights $P_\ell^\top$ and $\Pi^\top$:

$$\widetilde{H}_\ell = P_\ell \Pi H_\ell, \qquad \widetilde{U}_\ell = \mathcal{M}_\ell(\widetilde{H}_\ell), \qquad Y_\ell = \Pi^\top P_\ell^\top \widetilde{U}_\ell. \tag{7}$$

where, $\widetilde{U}_\ell$ is the scan output, and Y_ℓ denotes the output features of layer ℓ.

Complementary paths. Since a single traversal may fail to capture certain local relationships, we extend GCSO to two complementary geometric paths and fuse their routed outputs. Specifically, as shown in Figure 2(d) in the 12-layer backbone, we enable $K_\ell = 2$ paths with different axis-interleaving sequences (Morton xyz and yzx) at encoder layers 4, 8, and 12 to capture complementary geometric dependencies and reduce reliance on a single traversal sequence, while the remaining layers use $K_\ell = 1$ path with Morton xyz sequence. Each path independently performs the routing and back-projection operations described above based on its corresponding base sequence.

Moreover, for the $K_\ell = 2$ complementary paths, we fuse the restored features from each path using the path length and path weights defined below. For path k, let $\hat{C}_k = \Pi_k C = [\hat{c}_{k,1}; \ldots; \hat{c}_{k,G}]$ denote the ordered patch centers. We quantify path continuity by the mean distance between consecutive centers. To account for variations in input scale, we normalize this distance by the global center scale s_C, estimated from the coordinate-wise dispersion of the patch centers. The path length $\mathcal{D}$ and the path weights α are defined as

$$\mathcal{D}(\Pi_k) = \frac{\sum_{i=1}^{G-1} \|\hat{c}_{k,i+1} - \hat{c}_{k,i}\|_2}{(G-1)\max(s_C, \varepsilon)}, \qquad \alpha_{\ell,k} = \frac{\exp[-\mathcal{D}(\Pi_k)/\tau_p]}{\sum_{r=1}^{K_\ell} \exp[-\mathcal{D}(\Pi_r)/\tau_p]}. \tag{8}$$

The temperature $\tau_p > 0$ controls weight concentration, and $\varepsilon > 0$ ensures numerical stability. The weighted fusion gives greater weight to paths with shorter mean geometric steps to reduce the influence of long spatial jumps.

3.5 Training Objective

We jointly optimize the task loss with continuity and entropy regularization. The continuity term $\mathcal{C}$ encourages shorter geometric steps between scan positions. Since the continuous relaxation may produce uniform assignments that blur local differences, the entropy term $\mathcal{H}$ penalizes nearly uniform routing and encourages concentrated reads. For path k at layer ℓ,

$$\mathcal{C}_\ell^{(k)}(P) = \mathcal{D}(P_\ell^{(k)} \Pi_k C), \qquad \mathcal{H}_\ell^{(k)} = -\frac{\sum_{i,j} P_{\ell,ij}^{(k)} \log \max(P_{\ell,ij}^{(k)}, \varepsilon)}{G \log(\max(2, \min(G, 2w+1)))}. \tag{9}$$

Local windows gives at most $\min(G, 2w+1)$ candidates per row. We normalize entropy by the logarithm of the candidate count to keep scales comparable across window sizes and clamp the count to 2 to ensure a positive denominator.

Let $\mathcal{R}$ denote the set of layers with routing enabled. The training objective is

$$\mathcal{L} = \mathcal{L}_{\text{task}} + \frac{\lambda}{|\mathcal{R}|} \sum_{\ell \in \mathcal{R}} \sum_{k=1}^{K_\ell} \text{sg}(\alpha_{\ell,k}) \left[\mathcal{C}_\ell^{(k)} + \eta \mathcal{H}_\ell^{(k)} \right], \tag{10}$$

Table 1: **Shape classification performance and computational complexity after self-supervised pre-training.** OA (%) is reported on ScanObjectNN and ModelNet40. ModelNet40 uses 1,024 points; #P and FLOPs are reported in millions and billions, respectively.

Method	Venue	ScanObjectNN			ModelNet40 (1k pts)		#P (M)	FLOPs (G)
		OBJ_BG	OBJ_ONLY	PB_T50_RS	w/o Vote	w/ Vote		
Point-BERT (2022)	CVPR 22	87.43	88.12	83.07	92.7	93.2	23.8	4.8
Point-MAE (2022)	ECCV 22	92.77	91.22	89.04	92.7	93.8	23.8	4.8
PointGPT-S (2023)	NeurIPS 23	93.39	92.43	89.17	93.3	94.0	29.2	5.7
Point-FEMAE (2024)	AAAI 24	95.18	93.29	90.22	94.0	94.5	27.4	3.6
PointMamba (2024)	NeurIPS 24	94.32	92.60	89.31	93.6	94.1	12.3	3.6
Mamba3D (2024)	ACM MM 24	93.12	92.08	92.05	94.7	95.1	16.9	3.9
SAST (2025)	CVPR 25	94.32	91.91	89.10	93.4	–	12.3	3.6
StruMamba3D (2025)	ICCV 25	95.18	93.63	92.75	95.1	95.4	15.8	4.0
Point-PQAE (2025b)	ICCV 25	95.0	93.6	89.6	94.0	–	22.1	–
GCSO (ours)		**95.52**	**94.84**	**94.20**	**95.7**	**96.0**	16.9	3.9

For downstream fine-tuning, $\mathcal{L}_{\text{task}}$ uses classification or segmentation targets. Self-supervised pre-training uses masked point reconstruction, as described in Section 4.1. The coefficients λ and η set overall regularization and relative entropy weight, respectively, and sg denotes stop-gradient.

4 Experiments

To verify the effectiveness of the proposed approach for point cloud understanding, we evaluate shape classification after self-supervised pre-training. We further evaluate the transfer of the pre-trained representations to part segmentation and few-shot classification.

4.1 Implementation Details

We follow the masked autoencoding protocol of Point-MAE (Pang et al., 2022), pre-training on the ShapeNet-55 training split without labels. ShapeNet-55 covers 55 categories (Chang et al., 2015). Each input has 1,024 points, grouped by FPS and KNN into 64 patches of 32 points; 60% of the patches are randomly masked. The 12-layer encoder has feature dimension 384. We pre-train for 300 epochs with AdamW, learning rate 0.001, weight decay 0.05, batch size 128, and cosine decay with a 10-epoch warm-up. We then replace the reconstruction decoder with a task head and fine-tune the encoder and head. Full-data classification uses 300 epochs, batch size 32, and learning rate 0.0005, retaining the optimizer, weight decay, and schedule. The downstream loss includes the routing regularizers in Equation (10).

4.2 Datasets and Evaluation Protocols

Shape Classification. Table 1 compares shape classification performance after self-supervised pre-training. GCSO achieves 95.52%, 94.84%, and 94.20% on ScanObjectNN, exceeding the strongest listed baselines by 0.34, 1.21, and 1.45 percentage points, respectively. On ModelNet40, GCSO reaches 95.7% without voting and 96.0% with voting, both 0.6 points above StruMamba3D. Additional classification results under supervised training from scratch are provided in Appendix C.

Part segmentation. ShapeNetPart (Yi et al., 2016) contains 16 object categories and 50 part labels. We use 14,007 training/validation shapes and 2,874 test shapes, sample 2,048 points per shape, and follow the evaluation protocol of StruMamba3D (Wang et al., 2025). We report instance-averaged mIoU (mIoU_i).

Few-shot classification. Following Point-MAE and StruMamba3D (Pang et al., 2022; Wang et al., 2025), we evaluate 5-way and 10-way tasks on ModelNet40 with 10 or 20 labeled support shapes per class. Each fold uses 20 disjoint query shapes per class. We report mean OA and standard deviation over ten independently sampled folds for each setting.

4.3 Comparison with Existing Methods

Shape Classification. Table 1 compares shape classification performance after self-supervised pre-training. GCSO achieves 95.52%, 94.84%, and 94.20% on ScanObjectNN, exceeding the strongest listed baselines by 0.34, 1.21, and 1.45 percentage points, respectively. On ModelNet40, GCSO reaches 95.7% without voting and 96.0% with voting, both 0.6 points above StruMamba3D.

Table 2: **Part segmentation on ShapeNetPart.** Instance-averaged mIoU_i (%). Bold marks the best result.

Method	$\mathrm{mIoU}_i \uparrow$
Point-BERT (2022)	85.6
MaskPoint (2022)	86.0
Point-MAE (2022)	86.1
Point-M2AE (2022)	86.5
PointGPT-S (2023)	86.2
PointMamba (2024)	86.2
Mamba3D (2024)	85.6
StruMamba3D (2025)	**86.7**
GCSO (ours)	**86.7**

Table 3: **Few-shot classification on ModelNet40.** OA (%), reported as mean $\pm$ standard deviation over ten folds. Bold marks the best mean.

Method	5-way		10-way	
	10-shot	20-shot	10-shot	20-shot
Point-BERT (2022)	94.6 ± 3.6	93.9 ± 3.1	86.4 ± 5.4	91.3 ± 4.6
MaskPoint (2022)	95.0 ± 3.7	97.2 ± 1.7	91.4 ± 4.0	92.7 ± 5.1
Point-MAE (2022)	96.3 ± 2.5	97.8 ± 1.8	92.6 ± 4.1	93.4 ± 3.5
Point-M2AE (2022)	96.8 ± 1.8	98.3 ± 1.4	92.3 ± 4.5	95.0 ± 3.0
PointGPT-S (2023)	96.8 ± 2.0	98.6 ± 1.1	92.6 ± 4.6	95.2 ± 3.4
PointMamba (2024)	96.9 ± 2.0	99.0 ± 1.1	93.0 ± 4.4	95.6 ± 3.2
Mamba3D (2024)	96.4 ± 2.2	98.2 ± 1.2	92.4 ± 4.1	95.2 ± 2.9
StruMamba3D (2025)	97.5 ± 2.3	99.1 ± 1.4	93.5 ± 3.7	**96.1** ± 3.5
GCSO (ours)	**97.9** ± 1.9	**99.4** ± 0.7	**93.6** ± 4.0	96.0 ± 2.9

Part Segmentation. As shown in Table 2, GCSO reaches 86.7% mIoU_i, matching StruMamba3D at the reported precision. GCSO improves over Mamba3D by 1.1 percentage points, Point-MAE by 0.6 points, and Point-M2AE by 0.2 points.

Few-Shot Classification. Table 3 reports the few-shot results. GCSO achieves 97.9% and 99.4% mean OA in the 5-way settings, improving over StruMamba3D by 0.4 and 0.3 percentage points. In the 10-way settings, GCSO obtains 93.6% and 96.0%. Increasing the support set from 10 to 20 shapes per class raises GCSO accuracy by 1.5 points for 5-way and 2.4 points for 10-way tasks. GCSO achieves the highest listed mean in three of the four settings, supporting the transfer of its pre-trained representations to classification with limited labels.

4.4 Ablation Studies

We conduct ablation studies on ScanObjectNN PB_T50_RS and ModelNet40 without voting under the same pre-training and fine-tuning protocol as the complete model. The experiments follow the construction of GCSO from a geometric base order to trainable local reading, feature-conditioned adaptation, conservative routing, and complementary paths. Table 4 evaluates the main designs introduced in Sections 3.1, 3.3, and 3.4, as well as the operations used to construct the conservative routing matrix in Equation (6).

Main components. The geometric hard order provides the reference configuration in Table 4. Introducing the coordinate-driven local offset in Equation (5) makes the local reading pattern trainable and improves OA from 92.37% to 93.06% on PB_T50_RS and from 94.7% to 95.0% on ModelNet40. Conditioning the offset predictor f_θ on current-layer features further raises the results to 93.23% and 95.1%, indicating that current features help adapt the reading pattern to the current representation. The conservative matrix improves the results to 93.82% and 95.4% by preserving unit reading and participation totals during local feature assignment. Complementary xyz and yzx paths expose additional local candidate relationships and yield 94.20% and 95.7%.

Conservative routing. The right panel of Table 4 examines the three operations in Equation (6). Sinkhorn normalization refines the banded affinities into the preliminary assignment Q_ℓ. Symmetric diagonal completion converts the off-diagonal flow into the doubly stochastic matrix R_ℓ, and identity blending controls the amount of feature redistribution in P_ℓ. Removing the three operations individually gives 93.72%, 94.00%, and 93.61% OA on PB_T50_RS, respectively. Combining them

Table 4: **Ablation studies on the main GCSO designs and conservative routing.** OA (%) on ScanObjectNN PB_T50_RS (Scan) and ModelNet40 without voting (MN40). Left: contributions of local offsets, feature conditioning, conservative routing, and complementary paths. Right: effects of Sinkhorn normalization, symmetric completion, and identity blending.

	Offset	Feature	Conserv.	Paths	Scan	MN40
A	×	×	×	×	92.37	94.7
B	✓	×	×	×	93.06	95.0
C	✓	✓	×	×	93.23	95.1
D	✓	✓	✓	×	93.82	95.4
E	✓	✓	✓	✓	**94.20**	**95.7**

	Sinkhorn	Complete	Blend	Scan	MN40
R1	×	✓	✓	93.72	95.3
R2	✓	×	✓	94.00	95.5
R3	✓	✓	×	93.61	95.2
R4	✓	✓	✓	**94.20**	**95.7**

achieves the highest OA on both datasets, which supports their complementary roles in conservative routing. Appendix B further evaluates geometric path weighting and the routing regularizers.

4.5 Analysis of Learned Routing Behavior

We examine how the learned offset predictor and the conservative transport vary across network depth. Figure 3 summarizes statistics from 128 test shapes using a trained 12-layer GCSO checkpoint; the shapes are sampled by class with a fixed random seed. For scan position i, the plotted offset is the expected displacement of source indices relative to that position under row i of the routing matrix P_ℓ.

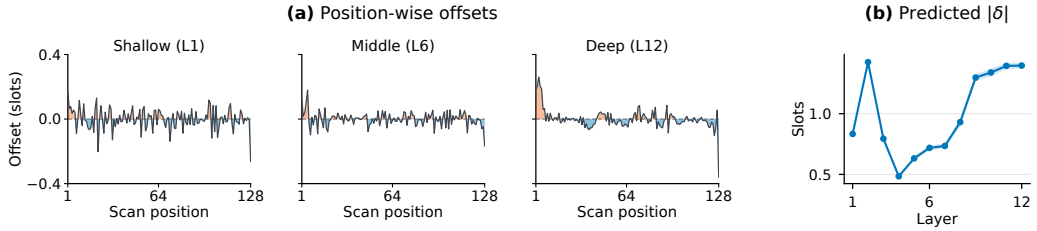

Figure 3: **Learned offsets on ScanObjectNN PB_T50_RS.** (a) Unaggregated position-wise offsets at shallow, middle, and deep layers for a representative test shape. Orange and blue indicate positive and negative displacement, respectively. (b) Layerwise mean magnitude of the predicted pre-transport offset. Colored shading denotes the 95% confidence interval.

The offsets in Figure 3(a) vary across depth and fluctuate around zero, showing that the transport introduces localized bidirectional adjustments. The shallow layer exhibits more frequent local fluctuations, the middle layer is comparatively concentrated near zero, and the deep layer retains sparse larger adjustments, indicating that the routed reading pattern evolves with representation depth.

Figure 3(b) provides the complementary pre-transport view. The mean predicted offset magnitude ranges from 0.48 to 1.42 scan slots and increases again in the upper layers after a mid-depth minimum. Together, the two panels show that GCSO learns depth-dependent offsets and converts them into bounded, position-specific routing adjustments.

5 Conclusion

In this work, we introduced Geometry-guided Conservative Soft Ordering (GCSO), a differentiable alternative to fixed geometric serialization for point-cloud state space models. GCSO learns local adjustments around a geometric base order while conserving the assignment budgets of scan positions and source tokens. Experiments on ModelNet40 and ScanObjectNN show consistent improvements under supervised and self-supervised settings, indicating that end-to-end optimization of scan organization is effective for point cloud understanding.

AI USE STATEMENT

Generative AI tools were used to improve the readability and linguistic presentation of the manuscript and to assist in searching for and identifying potentially relevant literature. All retrieved references were independently reviewed and verified by the authors using the original sources.

REFERENCES

Mohamed Afham, Isuru Dissanayake, Dinithi Dissanayake, Amaya Dharmasiri, Kanchana Thilakarathna, and Ranga Rodrigo. Crosspoint: Self-supervised cross-modal contrastive learning for 3d point cloud understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9902–9912, 2022.

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 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11799–11809, 2025.

Mathieu Blondel, Olivier Teboul, Quentin Berthet, and Josip Djolonga. Fast Differentiable Sorting and Ranking. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pp. 950–959, 2020.

Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. 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. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Marco Cuturi. Sinkhorn Distances: Lightspeed Computation of Optimal Transport. In *Advances in Neural Information Processing Systems*, volume 26, 2013.

Marco Cuturi, Olivier Teboul, and Jean-Philippe Vert. Differentiable Ranking and Sorting using Optimal Transport. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Tri Dao and Albert Gu. Transformers are SSMs: Generalized Models and Efficient Algorithms Through Structured State Space Duality. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 10041–10071, 2024.

Aditya Grover, Eric Wang, Aaron Zweig, and Stefano Ermon. Stochastic Optimization of Sorting Networks via Continuous Relaxations. In *International Conference on Learning Representations*, 2019.

Albert Gu and Tri Dao. Mamba: Linear-Time Sequence Modeling with Selective State Spaces. In *First Conference on Language Modeling*, 2024.

Albert Gu, Karan Goel, and Christopher Re. Efficiently Modeling Long Sequences with Structured State Spaces. In *International Conference on Learning Representations*, 2022.

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. doi: 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. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pp. 4995–5004, 2024. doi: 10.1145/3664647.3681173.

Philip A. Knight. The Sinkhorn-Knopp Algorithm: Convergence and Applications. *SIAM Journal on Matrix Analysis and Applications*, 30(1):261–275, 2008. doi: 10.1137/060659624.

Qihui Li, Qiliang Du, Lianfang Tian, Yihua Shao, and Guoyu Lu. Enhancing point cloud feature representation via historical node state increments in graph neural networks. *Pattern Recognition*, 172:112603, 2026. doi: 10.1016/j.patcog.2025.112603.

Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. PointCNN: Convolution On X-Transformed Points. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Dingkang Liang, Xin Zhou, Wei Xu, Xingkui Zhu, Zhikang Zou, Xiaoqing Ye, Xiao Tan, and Xiang Bai. PointMamba: A Simple State Space Model for Point Cloud Analysis. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Bin Liu, Chunyang Wang, Xuelian Liu, and Ge Zhang. DM3D: Deformable Mamba via Offset-Guided Differentiable Scanning for Point Cloud Understanding. *arXiv preprint arXiv:2512.03424*, 2026. URL https://arxiv.org/abs/2512.03424.

Haotian Liu, Mu Cai, and Yong Jae Lee. Masked Discrimination for Self-Supervised Learning on Point Clouds. In *European Conference on Computer Vision*, pp. 657–675, 2022.

Yue Liu, Yunjie Tian, Yuzhong Zhao, Hongtian Yu, Lingxi Xie, Yaowei Wang, Qixiang Ye, Jianbin Jiao, and Yunfan Liu. VMamba: Visual State Space Model. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

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. In *International Conference on Learning Representations*, 2022.

Gonzalo Mena, David Belanger, Scott Linderman, and Jasper Snoek. Learning Latent Permutations with Gumbel-Sinkhorn Networks. In *International Conference on Learning Representations*, 2018.

Yatian Pang, Wenxiao Wang, Francis E. H. Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked Autoencoders for Point Cloud Self-supervised Learning. In *European Conference on Computer Vision*, 2022.

Sebastian Prillo and Julian Eisenschlos. SoftSort: A Continuous Relaxation for the argsort Operator. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pp. 7793–7802, 2020.

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 Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Advances in Neural Information Processing Systems*, volume 30, 2017b.

Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. PointNeXt: Revisiting PointNet++ with Improved Training and Scaling Strategies. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Kanglin Qu, Pan Gao, Qun Dai, and Yuanhao Sun. HydraMamba: Multi-head state space model for global point cloud learning. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 333–342, 2025. doi: 10.1145/3746027.3755114.

Kanglin Qu, Pan Gao, Qun Dai, Zhanzhi Ye, Rui Ye, and Yuanhao Sun. CloudMamba: Grouped Selective State Spaces for Point Cloud Analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 8659–8667, 2026. doi: 10.1609/aaai.v40i11.37818. Issue 11.

Richard Sinkhorn and Paul Knopp. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 21(2):343–348, 1967. doi: 10.2140/pjm.1967.21.343.

Yi Tay, Dara Bahri, Liu Yang, Donald Metzler, and Da-Cheng Juan. Sparse Sinkhorn Attention. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pp. 9438–9447, 2020.

Hugues Thomas, Charles R. Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, Francois Goulette, and Leonidas J. Guibas. KPConv: Flexible and Deformable Convolution for Point Clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6411–6420, 2019. doi: 10.1109/ICCV.2019.00651.

Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Duc 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*, 2019.

Chuxin Wang, Yixin Zha, Wenfei Yang, and Tianzhu Zhang. StruMamba3D: Exploring Structural Mamba for Self-supervised Point Cloud Representation Learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 28546–28555, 2025.

Peng-Shuai Wang. OctFormer: Octree-based Transformers for 3D Point Clouds. *ACM Transactions on Graphics*, 42(4), 2023. doi: 10.1145/3592131.

Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic Graph CNN for Learning on Point Clouds. *ACM Transactions on Graphics*, 38(5), 2019. doi: 10.1145/3326362.

Xiaoyang Wu, Yixing Lao, Li Jiang, Xihui Liu, and Hengshuang Zhao. Point Transformer V2: Grouped Vector Attention and Partition-based Pooling. In *Advances in Neural Information Processing Systems*, volume 35, 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4840–4851, 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. doi: 10.1109/CVPR.2015.7298801.

Ke Xu, Xinpu Liu, Yinghui Gao, Qingyong Hu, Xinjie Wang, Hanyun Wang, and Yulan Guo. PointMamba++: Rethinking Ordering and Convolution Strategy of State Space Model for Point Cloud Analysis. *IET Computer Vision*, 20(1):e70064, 2026. doi: 10.1049/cvi2.70064.

Zi Jian Yew and Gim Hee Lee. RPM-Net: Robust Point Matching Using Learned Features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11824–11833, 2020.

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*, 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.

Yaohua Zha, Huizhen Ji, Jinmin Li, Rongsheng Li, Tao Dai, Bin Chen, Zhi Wang, and Shu-Tao Xia. Towards compact 3D representations via point feature enhancement masked autoencoders. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 6962–6970, 2024. doi: 10.1609/aaai.v38i7.28522.

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. In *Advances in Neural Information Processing Systems*, volume 35, 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*, volume 39, pp. 10121–10130, 2025a. doi: 10.1609/aaai.v39i10.33098. Issue 10.

Xiangdong Zhang, Shaofeng Zhang, and Junchi Yan. Towards more diverse and challenging pre-training for point cloud learning: Self-supervised cross reconstruction with decoupled views. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2025b.

Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip H. S. Torr, and Vladlen Koltun. Point Transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16259–16268, 2021.

Bowen Zhou, Lixin Zhan, and Jie Jiang. LFE-PointMamba: Point cloud learning via local feature enhancement and state space model. *Knowledge-Based Systems*, 329:114350, 2025. doi: 10.1016/j.knosys.2025.114350.

Lianghui Zhu, Bencheng Liao, Qian Zhang, Xinlong Wang, Wenyu Liu, and Xinggang Wang. Vision Mamba: Efficient Visual Representation Learning with Bidirectional State Space Model. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pp. 62429–62442, 2024.

A Differentiability and Weight Conservation in GCSO

We establish the gradient path of soft ordering, prove the conservative construction, and derive the resulting feature-redistribution identities. All routing derivatives below hold the input features and coordinates fixed, isolating the contribution of the routing parameters to the task gradient.

A.1 Differentiability of Hard and Soft Ordering

Let $\hat{H} = \Pi H \in \mathbb{R}^{G\times d}$ denote features in a geometric base order and let $\mathcal{M}$ be the state-space module. For an assignment matrix P, define

$$F(\hat{H}) = P^\top \mathcal{M}(P\hat{H}). \tag{11}$$

A hard assignment belongs to $\mathcal{P} = \{P \in \{0,1\}^{G\times G} : P\mathbf{1} = P^\top\mathbf{1} = \mathbf{1}\}$. Let $\mathcal{P}_w = \{P \in \mathcal{P} : P_{ij} = 0 \text{ for } |i-j| > w\}$ be its local subset. Learned scores $S_\theta(\hat{H}, \hat{C})$ yield

$$P_\theta^{\text{hard}} = \arg\max_{P\in\mathcal{P}_w} \langle S_\theta(\hat{H}, \hat{C}), P\rangle_F. \tag{12}$$

Hard-assignment gradient. Suppose P_* uniquely maximizes the score at S and the feasible set contains another assignment. Finiteness of the set gives a positive margin

$$\gamma = \min_{\substack{P\in\mathcal{P}_w \\ P\neq P_*}} \langle S, P_* - P\rangle_F > 0, \qquad \|P_* - P\|_F \le \sqrt{2G}. \tag{13}$$

For any perturbation E with $\|E\|_F < \gamma/\sqrt{2G}$, the Cauchy–Schwarz inequality gives, for every $P \neq P_*$,

$$\begin{aligned}\langle S+E, P_* - P\rangle_F &= \langle S, P_* - P\rangle_F + \langle E, P_* - P\rangle_F \\ &\ge \gamma - \|E\|_F\|P_* - P\|_F \ge \gamma - \sqrt{2G}\|E\|_F > 0.\end{aligned} \tag{14}$$

Hence the selected assignment is constant in an open neighborhood of S. Ties lie in a finite union of hyperplanes $\langle S, P - P'\rangle_F = 0$ for distinct assignments, a set of Lebesgue measure zero in score space. A singleton feasible set is constant everywhere. Ordinary differentiation through the hard selection therefore gives

$$\begin{aligned}D_S P_\theta^{\text{hard}} &= 0 \quad \text{almost everywhere in score space,} \\ \nabla_\theta \mathcal{L}_{\text{task}}|_{\text{routing}} &= (D_\theta S_\theta)^*(D_S P_\theta^{\text{hard}})^* \nabla_P \mathcal{L}_{\text{task}} = 0.\end{aligned} \tag{15}$$

Here, D denotes a derivative and * its adjoint under the Frobenius inner product; the chain-rule identity applies at differentiable score points. A fixed geometric assignment likewise has $D_\theta P_{\text{geo}}^{\text{hard}} = 0$ for routing parameters θ.

Soft-assignment gradient. Relaxing binary entries retains the unit row and column budgets in the convex set

$$\mathcal{B}_G = \{P \in \mathbb{R}_+^{G\times G} : P\mathbf{1} = P^\top\mathbf{1} = \mathbf{1}\}. \tag{16}$$

GCSO parameterizes a banded, symmetric matrix $P = P_\theta^{\text{GCSO}}(\hat{H}, \hat{C})$ in $\mathcal{B}_G$. Write $\widetilde{H} = P\hat{H}$, $\widetilde{U} = \mathcal{M}(\widetilde{H})$, and $\hat{Y} = P^\top\widetilde{U}$. At a differentiable point of $\mathcal{M}$, the product and chain rules give

$$\mathrm{d}\widetilde{H} = (\mathrm{d}P)\hat{H}, \qquad \mathrm{d}\hat{Y} = (\mathrm{d}P)^\top\widetilde{U} + P^\top D\mathcal{M}_{\widetilde{H}}[(\mathrm{d}P)\hat{H}]. \tag{17}$$

Let $G_{\hat{Y}} = \nabla_{\hat{Y}}\mathcal{L}_{\text{task}}$ and $G_{\widetilde{H}} = D\mathcal{M}^*_{\widetilde{H}}[PG_{\hat{Y}}]$. Taking inner products with the output gradient yields

$$\begin{aligned}\mathrm{d}\mathcal{L}_{\text{task}} &= \langle G_{\hat{Y}}, (\mathrm{d}P)^\top\widetilde{U}\rangle_F + \langle PG_{\hat{Y}}, D\mathcal{M}_{\widetilde{H}}[(\mathrm{d}P)\hat{H}]\rangle_F \\ &= \langle \widetilde{U}G_{\hat{Y}}^\top, \mathrm{d}P\rangle_F + \langle G_{\widetilde{H}}\hat{H}^\top, \mathrm{d}P\rangle_F, \\ \nabla_P\mathcal{L}_{\text{task}} &= \widetilde{U}G_{\hat{Y}}^\top + G_{\widetilde{H}}\hat{H}^\top.\end{aligned} \tag{18}$$

The two terms account for back-projection and the routed scan input, respectively. The next derivation connects this matrix gradient to the learned offsets.

A.2 Gradient Through the Conservative Parameterization

We omit the layer index and write $z_i = f_\theta(\hat{h}_i, \mathrm{Std}(\hat{c}_i))$. On the fixed support $\Omega_w = \{(i,j) : |i-j| \le w\}$, the offset and log-affinity derivatives are

$$\delta_i = a \tanh z_i, \qquad \frac{\partial \delta_i}{\partial z_i} = a(1 - \tanh^2 z_i), \qquad \frac{\partial A_{ij}}{\partial \delta_r} = \begin{cases} (j - i - \delta_i)/(\sigma^2 \tau), & r = i, \\ 0, & r \neq i. \end{cases} \tag{19}$$

For one row-normalization step, let $L'_{ij} = L_{ij} - \log \sum_{k:(i,k)\in\Omega_w} \exp L_{ik}$ and $p_{ik} = \exp L_{ik} / \sum_{s:(i,s)\in\Omega_w} \exp L_{is}$. Its differential, and that of a column-normalization step with column softmax weights q_{kj}, are

$$\mathrm{d}L'_{ij} = \mathrm{d}L_{ij} - \sum_{k:(i,k)\in\Omega_w} p_{ik}\, \mathrm{d}L_{ik}, \qquad \mathrm{d}L''_{ij} = \mathrm{d}L'_{ij} - \sum_{k:(k,j)\in\Omega_w} q_{kj}\, \mathrm{d}L'_{kj}. \tag{20}$$

Each row and column has an allowed diagonal entry. For $\sigma, \tau > 0$ and finite predictor outputs, the normalizers are positive and a finite composition of these steps is smooth on the allowed entries. If $L^{(T)}$ is the normalized log-matrix, then $Q = \exp L^{(T)}$ satisfies $\mathrm{d}Q = Q \odot \mathrm{d}L^{(T)}$ on Ω_w, while masked entries and their differentials remain zero.

For the conservative correction, define $J = \mathbf{1}\mathbf{1}^\top - I$ and $b = B\mathbf{1}$. Direct differentiation gives

$$\begin{aligned} \mathrm{d}B &= \tfrac{1}{2}(\mathrm{d}Q + \mathrm{d}Q^\top) \odot J, & \mathrm{d}b &= (\mathrm{d}B)\mathbf{1}, \\ \mathrm{d}R &= \frac{\mathrm{d}B}{\kappa} - \frac{B}{\kappa^2}\mathrm{d}\kappa - \mathrm{Diag}\left(\frac{\mathrm{d}b}{\kappa} - \frac{b}{\kappa^2}\mathrm{d}\kappa\right), & & \\ \mathrm{d}P &= \rho\, \mathrm{d}R + (R - I)\, \mathrm{d}\rho, & \mathrm{d}\rho &= \rho(1-\rho)\, \mathrm{d}\beta. \end{aligned} \tag{21}$$

On a branch where $\max_i b_i < 1$, $\mathrm{d}\kappa = 0$. If $b_m > 1$ is the unique maximum, $\mathrm{d}\kappa = \mathrm{d}b_m$. At ties, the maximum admits the standard subgradient $\mathrm{d}\kappa = \sum_{m\in\mathcal{I}} \lambda_m \mathrm{d}b_m$, where $\mathcal{I}$ indexes the active maxima, $\lambda_m \ge 0$, and $\sum_m \lambda_m = 1$; an active constant branch contributes zero. Since $\kappa \ge 1$, the correction is locally Lipschitz and differentiable almost everywhere as a function of Q.

Combining the derivatives gives the routing contribution to the predictor gradient and the gradient of the blending parameter:

$$\begin{aligned} g_i &= \left\langle \nabla_P \mathcal{L}_{\mathrm{task}}, \frac{\partial P}{\partial \delta_i} \right\rangle_F, \\ \nabla_\theta \mathcal{L}_{\mathrm{task}}|_{\mathrm{routing}} &= \sum_i g_i\, a(1 - \tanh^2 z_i)\, \nabla_\theta f_\theta(\hat{h}_i, \mathrm{Std}(\hat{c}_i)), \\ \frac{\partial \mathcal{L}_{\mathrm{task}}}{\partial \beta} &= \rho(1-\rho)\, \langle \nabla_P \mathcal{L}_{\mathrm{task}}, R - I \rangle_F . \end{aligned} \tag{22}$$

These expressions provide an explicit end-to-end gradient path from the task loss to both the offset predictor and the routing strength through the implemented conservative construction.

A.3 Weight Conservation and Local Support

Proposition 1 (Conservative local routing). Let $Q \in \mathbb{R}^{G\times G}$ be finite and nonnegative, with $Q_{ij} = 0$ outside Ω_w. For $0 \le \rho \le 1$, the construction in Eq. (6) yields

$$P \ge 0, \qquad P = P^\top, \qquad P\mathbf{1} = P^\top \mathbf{1} = \mathbf{1}, \qquad P_{ij} = 0 \text{ if } |i-j| > w. \tag{23}$$

Proof. Suppressing the layer index, write the construction entrywise:

$$\begin{aligned} B_{ij} &= \begin{cases} (Q_{ij} + Q_{ji})/2, & i \neq j, \\ 0, & i = j, \end{cases} \qquad b_i = \sum_j B_{ij}, \qquad \kappa = \max(1, \max_i b_i), \\ R_{ij} &= \begin{cases} B_{ij}/\kappa, & i \neq j, \\ 1 - b_i/\kappa, & i = j, \end{cases} \qquad P_{ij} = \begin{cases} \rho B_{ij}/\kappa, & i \neq j, \\ 1 - \rho b_i/\kappa, & i = j. \end{cases} \end{aligned} \tag{24}$$

Nonnegativity and symmetry follow from

$$B_{ij} = B_{ji} \geq 0, \qquad 0 \leq b_i \leq \kappa, \qquad R_{ii} = 1 - b_i/\kappa \geq 0, \qquad P_{ii} = 1 - \rho b_i/\kappa \geq 1 - \rho \geq 0. \tag{25}$$

The diagonal remainder completes each row to one:

$$\sum_j R_{ij} = 1 - \frac{b_i}{\kappa} + \frac{\sum_{j \neq i} B_{ij}}{\kappa} = 1 - \frac{b_i}{\kappa} + \frac{b_i}{\kappa} = 1. \tag{26}$$

Since $R = R^\top$, the column sums also equal one. Identity blending preserves both marginals:

$$\begin{aligned} P\mathbf{1} &= (1-\rho)I\mathbf{1} + \rho R\mathbf{1} = \mathbf{1}, \\ P^\top\mathbf{1} &= (1-\rho)I\mathbf{1} + \rho R^\top\mathbf{1} = \mathbf{1}. \end{aligned} \tag{27}$$

Finally, for $|i - j| > w$, symmetry of the support implies

$$Q_{ij} = Q_{ji} = 0 \implies B_{ij} = 0 \implies R_{ij} = P_{ij} = 0. \tag{28}$$

This proves the proposition. The algebra holds for every finite Sinkhorn iteration count because the correction uses only nonnegativity and local support of Q. In particular, the final marginal identities are exact in real arithmetic, independently of the preliminary marginal residuals.

A.4 Conservative Feature Redistribution

Corollary 1 (Reading and participation budgets). For any feature matrix $\hat{H} \in \mathbb{R}^{G \times d}$, Proposition 1 gives

$$\begin{aligned} (P\hat{H})_i &= \sum_j P_{ij}\hat{H}_j, \qquad \sum_j P_{ij} = 1, \qquad \sum_i P_{ij} = 1, \\ \mathbf{1}^\top P\hat{H} &= \sum_j \left(\sum_i P_{ij} \right) \hat{H}_j = \sum_j \hat{H}_j = \mathbf{1}^\top \hat{H}. \end{aligned} \tag{29}$$

Each slot is a convex combination of its local sources, and each source contributes one unit of total weight across slots. The channel sum, and therefore the channel mean, is preserved by the routing operation. The identities hold pointwise for input-dependent $P = P(\hat{H}, \hat{C})$.

Corollary 2 (Controlled departure from the base order). Let $\| \cdot \|_\infty$ be the induced maximum absolute row-sum norm. The entrywise form of P yields

$$\begin{aligned} (P\hat{H} - \hat{H})_i &= \frac{\rho}{\kappa} \sum_{j \neq i} B_{ij}(\hat{H}_j - \hat{H}_i), \\ \|P - I\|_\infty &= \max_i \left(\frac{\rho b_i}{\kappa} + \sum_{j \neq i} \frac{\rho B_{ij}}{\kappa} \right) = 2\rho \max_i \frac{b_i}{\kappa} \leq 2\rho. \end{aligned} \tag{30}$$

Thus the routed update exchanges local feature differences with a strength controlled by ρ. For fixed R and $\hat{H}$, continuity of $\mathcal{M}$ gives the base-scan limit

$$\lim_{\rho \to 0} P = I, \qquad \lim_{\rho \to 0} P^\top \mathcal{M}(P\hat{H}) = \mathcal{M}(\hat{H}). \tag{31}$$

A.5 Participation-Weighted Back-Projection

Let $\widetilde{U}_\ell$ be the scan output and Π the geometric base permutation. With $T_\ell = P_\ell \Pi$, forward routing and back-projection form an adjoint pair:

$$\begin{aligned} Y_\ell &= T_\ell^\top \widetilde{U}_\ell = \Pi^\top P_\ell^\top \widetilde{U}_\ell, \\ \langle T_\ell H_\ell, \widetilde{U}_\ell \rangle_F &= \operatorname{tr}(H_\ell^\top T_\ell^\top \widetilde{U}_\ell) = \langle H_\ell, T_\ell^\top \widetilde{U}_\ell \rangle_F. \end{aligned} \tag{32}$$

In base order, the update to source j uses exactly its forward participation weights:

$$(P_\ell^\top \widetilde{U}_\ell)_j = \sum_i P_{\ell,ij} \widetilde{U}_{\ell,i}, \qquad \sum_i P_{\ell,ij} = 1. \tag{33}$$

Since $\Pi \mathbf{1} = \mathbf{1}$ and $P_\ell \mathbf{1} = \mathbf{1}$, redistribution of the scan outputs preserves their channel sums:

$$\begin{aligned} \mathbf{1}^\top Y_\ell &= \mathbf{1}^\top \Pi^\top P_\ell^\top \widetilde{U}_\ell = (P_\ell \Pi \mathbf{1})^\top \widetilde{U}_\ell \\ &= \mathbf{1}^\top \widetilde{U}_\ell. \end{aligned} \tag{34}$$

The transpose implements participation-weighted aggregation, and $\Pi^\top$ restores the patch indices used by the residual branch and subsequent spatial aggregation. The conservation identities apply separately to routing the input features and redistributing the scan outputs.

B Additional Ablation Studies

This appendix complements Table 4 by isolating two designs that stabilize and extend the conservative local relaxation: complementary path fusion in Section 3.4 and routing regularization in Section 3.5. All results use the same self-supervised pre-training and downstream fine-tuning protocol as the main ablation study. Table 5(a) corresponds to the path-fusion definition in Section 3.4 and Equation (8), while Table 5(b) corresponds to Equations (9) and (10).

Table 5: **Additional analysis of GCSO.** OA (%) without voting. (a) Complementary paths and geometric fusion. (b) Continuity ($\mathcal{C}$) and entropy ($\mathcal{H}$) regularizers. Bold marks the best result.

Config.	xyz	yzx	Fusion	PB_T50_RS	MN40
P1	✓	×	–	93.79	95.4
P2	×	✓	–	93.65	95.2
P3	✓	✓	×	94.03	95.5
P4	✓	✓	✓	**94.20**	**95.7**

Config.	$\mathcal{C}$	$\mathcal{H}$	PB_T50_RS	MN40
L1	×	×	93.41	95.1
L2	✓	×	93.68	95.3
L3	×	✓	93.75	95.3
L4	✓	✓	**94.20**	**95.7**

Complementary paths. The first table separates the two permutations Π_k used by the path-fusion definition in Section 3.4. The xyz and yzx paths individually reach 93.79% and 93.65% OA on PB_T50_RS, confirming that each axis order forms an effective base sequence. Equal averaging of their restored outputs increases OA to 94.03%, showing that the two orders expose complementary local candidates. Replacing equal averaging with the geometric weights $\alpha_{\ell,k}$ from Equation (8) further raises OA to 94.20% on PB_T50_RS and 95.7% on ModelNet40.

Routing regularizers. The second table maps $\mathcal{C}_\ell^{(k)}$ and $\mathcal{H}_\ell^{(k)}$ directly to the two terms in Equation (9). Either term improves the model without routing regularization. The continuity term keeps the soft relaxation near geometrically short transitions, while the entropy term promotes selective rather than nearly uniform local reads. Combining both terms reaches 94.20% OA on PB_T50_RS and 95.7% on ModelNet40, supporting their complementary roles in controlling the local soft relaxation.

C Supervised Classification Results

For supervised classification, GCSO is trained from scratch on the labeled training sets, while the competing results retain their reported settings.

Table 6 reports the supervised classification results. GCSO reaches 93.63%, 93.63%, and 92.78% OA on the three ScanObjectNN variants, exceeding the strongest listed baselines by 1.38, 2.24, and 4.48 percentage points, respectively. GCSO also obtains 94.0% on ModelNet40 without voting, matching the best listed result. These results support the effectiveness of GCSO without pre-training.

Table 6: **Shape classification performance and computational complexity under supervised learning.** OA (%) is reported on ScanObjectNN and ModelNet40. ModelNet40 uses 1,024 points; #P and FLOPs are reported in millions and billions, respectively.

Method	Venue	ScanObjectNN			ModelNet40 (1k pts)		#P (M)	FLOPs (G)
		OBJ_BG	OBJ_ONLY	PB_T50_RS	w/o Vote	w/ Vote		
PointNet (2017a)	CVPR 17	73.3	79.2	68.0	89.2	–	3.5	0.5
PointNet++ (2017b)	NeurIPS 17	82.3	84.3	77.9	90.7	–	1.5	1.7
DGCNN (2019)	TOG 19	82.8	86.2	78.1	92.9	–	1.8	2.4
PointNeXt (2022)	NeurIPS 22	–	–	87.7	92.9	–	1.4	3.6
PointMamba (2024)	NeurIPS 24	88.30	87.78	82.48	–	–	12.3	3.6
HydraMamba (2025)	ACM MM 25	–	–	88.3	**94.0**	–	–	–
SAST (2025)	CVPR 25	92.25	91.39	87.30	92.7	–	12.3	3.6
PCM (2025a)	AAAI 25	–	–	88.1	93.4	–	34.2	45.0
GCSO (ours)		**93.63**	**93.63**	**92.78**	**94.0**	–	16.9	3.9