

Supplementary Materials:

MAMBA3D: Enhancing Local Features for 3D Point Cloud Analysis via State Space Model

Anonymous Authors

In the appendix, we first show the t-SNE visualization of feature vectors in Appendix A. Then, we present additional experimental results, including linear SVM classification results in Appendix B.1.1, detailed part segmentation results in Appendix B.1.2, and further ablation studies in Appendix B.2. We provide more implementation details in Appendix C. Finally, we discuss the limitations of the MAMBA3D model and potential directions for future improvement in Appendix D.

A VISUALIZATION

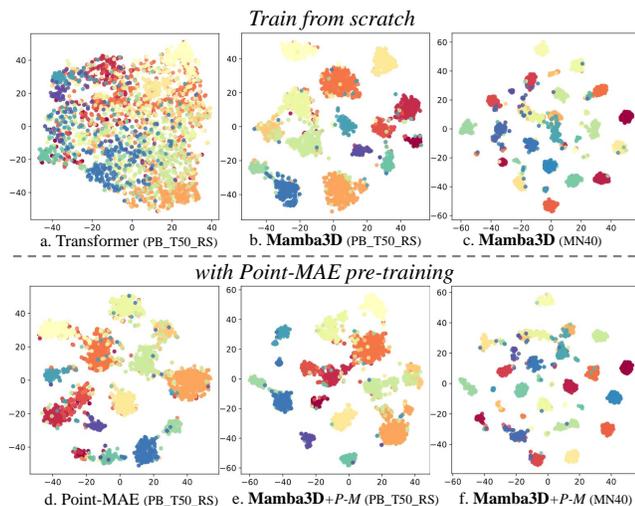


Figure 1: Visualization of feature distributions. We show the t-SNE visualization of feature vectors learned by MAMBA3D and Transformer. PB_T50_RS: ScanObjectNN (PB_T50_RS) dataset. P-M: Point-MAE strategy. MN40: ModelNet40 dataset.

We visualize the feature distribution using the t-SNE method [13], as depicted in Fig. 1. When trained from scratch, it’s evident that Transformer [14] struggles to effectively learn a well-separated feature distribution, as shown in Fig. 1.a. In contrast, the features learned by our MAMBA3D, as presented in Fig. 1.b and Fig. 1.c, form clear and distinct clusters that correspond to different categories. This demonstrates the superior ability of MAMBA3D to effectively learn and capture discriminative feature distributions from 3D point clouds, even when trained from scratch. When using the Point-MAE [8] pre-training strategy, the features learned by Transformer (Fig. 1.d) are less dispersed compared to MAMBA3D (Fig. 1.e and Fig. 1.f). Interestingly, the feature distribution of the pre-trained Transformer still falls short of the distribution achieved by MAMBA3D trained from scratch, further showcasing the efficient and powerful point cloud learning capability of MAMBA3D. The

results indicate that MAMBA3D effectively learns point cloud feature distribution even from scratch, highlighting the efficiency of our LNP and bi-SSM modules. Furthermore, pre-training enhances MAMBA3D’s performance.

B ADDITIONAL EXPERIMENTS

B.1 Additional Evaluation

Table 1: Linear SVM Classification on ModelNet40 dataset. We compare with methods using the \bullet plain Transformer architectures, \circ dedicated architectures for 3D understanding, and \star Mamba-based architectures. ModelNet40: overall accuracy (%) on ModelNet40 dataset. #P: model parameters (M). #F: FLOPs (G).

Method	ModelNet40 (%) \uparrow	#P \downarrow	#F \downarrow
\circ 3D-GAN [17]	83.3	-	-
\circ SO-Net [7]	87.3	-	-
\circ FoldingNet [19]	88.4	-	-
\circ 3D-PointCapsNet [22]	88.9	-	-
\circ VIP-GAN [6]	90.2	-	-
\star MAMBA3D	91.4	16.9	3.9
\circ PointNet+OcCo [15]	88.7	3.5	0.5
\circ PointNet+CrossPoint [1]	89.1	3.5	0.5
\circ DGCNN+Jigsaw [11]	90.6	1.8	2.4
\circ DGCNN+OcCo [15]	90.7	1.8	2.4
\circ DGCNN+CrossPoint [1]	91.2	1.8	2.4
\bullet Point-BERT [21]	87.4	22.1	4.8
\bullet Transformer+OcCo [15]	89.2	22.1	4.8
\bullet Point-MAE [8]	91.0	22.1	4.8
\star MAMBA3D +P-M [8]	91.5 +0.5	16.9	3.9

B.1.1 Linear SVM Classification. We conduct linear SVM classification [2] experiments on the ModelNet40 dataset [18], and the results are shown in Table 1. Linear SVM classification is used to evaluate the discriminative quality of pre-trained features. We utilize the pre-trained features to fit a linear SVM classifier on the training set, with point cloud input size $N=1024$. The results show that when trained from scratch, MAMBA3D achieves 91.4% overall accuracy (OA), surpassing dedicated architectures like FoldingNet [19] and VIP-GAN [6]. When using the Point-MAE [8] pre-training strategy, MAMBA3D achieves 91.5%, surpassing Point-BERT [21] and Point-MAE by 4.1% and 0.5%, respectively, and also outperforming DGCNN+OcCo’s 91.2% [15]. The linear SVM experiments demonstrate that MAMBA3D can effectively learn point cloud features, leading to excellent discriminative ability.

Table 2: Detailed part segmentation results on the ShapeNetPart dataset. We report the mean IoU across all part categories $mIoU_C$ (%) and the mean IoU across all instances $mIoU_I$ (%), as well as the IoU (%) for each category.

Methods	$mIoU_C$	$mIoU_I$	aero	bag	cap	car	chair	e-phone	guitar	knife	lamp	laptop	motor	mug	pistol	rocket	s-board	table
<i>Supervised Learning Only</i>																		
○ PointNet[9]	80.39	83.7	83.4	78.7	82.5	74.9	89.6	73.0	91.5	85.9	80.8	95.3	65.2	93.0	81.2	57.9	72.8	80.6
○ PointNet++[10]	81.85	85.1	82.4	79.0	87.7	77.3	<u>90.8</u>	71.8	91.0	85.9	83.7	95.3	71.6	94.1	81.3	58.7	76.4	82.6
○ DGCNN[16]	82.33	<u>85.2</u>	<u>84.0</u>	83.4	86.7	77.8	90.6	74.7	<u>91.2</u>	<u>87.5</u>	82.8	95.7	66.3	94.9	81.1	63.5	74.5	82.6
● Transformer [14]	<u>83.42</u>	85.1	82.9	85.4	87.7	<u>78.8</u>	90.5	80.8	91.1	87.7	85.3	95.6	<u>73.9</u>	<u>94.9</u>	<u>83.5</u>	61.2	74.9	80.6
★ MAMBA3D	83.74	85.7	84.8	<u>83.7</u>	<u>86.7</u>	80.0	91.2	<u>79.2</u>	91.4	87.2	<u>84.8</u>	95.2	76.6	95.0	84.7	<u>63.3</u>	<u>74.9</u>	<u>80.9</u>
<i>With Self-supervised Pre-training</i>																		
● OcCo [15]	83.42	85.1	83.3	85.2	88.3	<u>79.9</u>	90.7	74.1	<u>91.9</u>	<u>87.6</u>	84.7	95.4	75.5	94.4	84.1	63.1	75.7	80.8
● Point-BERT [21]	84.11	85.6	84.3	84.8	88.0	79.8	<u>91.0</u>	81.7	91.6	87.9	85.2	95.6	<u>75.6</u>	94.7	84.3	63.4	76.3	81.5
● Point-MAE [8]	-	86.1	84.3	85.0	88.3	80.5	91.3	78.5	92.1	87.4	86.1	96.1	75.2	94.6	<u>84.7</u>	<u>63.5</u>	<u>77.1</u>	82.4
★ MAMBA3D+P-B	84.13	<u>85.7</u>	84.3	84.0	<u>87.9</u>	79.8	90.9	<u>80.6</u>	91.3	87.2	86.1	95.4	76.1	94.7	84.8	64.3	77.6	81.2

Table 3: Ablation on feature dimension. Overall accuracy (%) on ScanObjectNN (OBJ_ONLY) dataset without voting, model parameters (M), and FLOPs (G) are reported.

Dimension	OBJ_ONLY (%) \uparrow	Params (M) \downarrow	FLOPs (G) \downarrow
192	90.5	4.9	2.1
256	91.4	8.1	2.6
384	92.1	16.9	3.9
512	84.2	29.1	5.6
768	88.5	63.6	10.2

B.1.2 Detailed Part Segmentation. Detailed part segmentation results on the ShapeNetPart dataset are presented in Table 2. When trained from scratch, MAMBA3D achieves higher $mIoU_C$ and $mIoU_I$ than Transformer [14], by +0.3% and +0.6%, respectively. It outperforms Transformer in 9 out of 16 categories. When using the Point-BERT [21] pre-training strategy, MAMBA3D surpasses Point-BERT in both $mIoU_C$ and $mIoU_I$, showing higher results in 6 categories compared to Point-MAE [8]. Particularly in the challenging *motor* category, MAMBA3D outperforms Point-BERT and Point-MAE by +0.9% and +0.8%, respectively. MAMBA3D demonstrates greater efficiency, with parameters and FLOPs reduced by -17.8% and -31.4% compared to Transformer (23.0M/11.8G vs. 27.1M/15.5G). Furthermore, we observe that existing masked point modeling pre-training strategies are less suitable for sequential models like MAMBA3D, yielding higher gains for Transformer compared to MAMBA3D.

B.2 Additional Ablation Study

B.2.1 Ablation on feature dimension. We conduct ablation experiments on the intermediate feature dimension of the model on the ScanObjectNN (OBJ_ONLY) dataset, and the results are shown in Table 3. The results indicate that as the feature dimension increases, both the model’s parameters and FLOPs increase, while the overall accuracy reaches its maximum of 92.1% at a dimension of 384.

B.2.2 Ablation on encoder depth. We explore the impact of encoder layers on the model, and the results on the ScanObjectNN (OBJ_ONLY) dataset are shown in Table 4. As the number of layers increases, both the model’s parameters and FLOPs gradually

Table 4: Ablation on encoder depth. Overall accuracy (%) on ScanObjectNN (OBJ_ONLY) dataset without voting, model parameters (M), and FLOPs (G) are reported.

Depth	OBJ_ONLY (%) \uparrow	Params (M) \downarrow	FLOPs (G) \downarrow
2	90.7	3.5	2.3
4	91.9	6.2	2.6
6	91.6	8.9	2.9
8	90.9	11.6	3.3
10	91.7	14.2	3.6
12	92.1	16.9	3.9

increase, with the overall accuracy (OA) reaching a maximum of 92.1% at 12 layers. It is worth noting that even with 4 layers, the model’s OA still reaches 91.9%. However, the parameters and FLOPs decrease by 172.6% and 50%, respectively, further demonstrating that MAMBA3D can accurately and efficiently learn point features.

C IMPLEMENTATION DETAILS

We present more implementation details in Table 5. During pre-training, we follow the training settings of Point-MAE¹ [8] or Point-BERT² [21], and pre-train on the ShapeNetCore training data, which is a subset of ShapeNet [3]. Pre-training takes approximately 24 hours on a single NVIDIA RTX 3090 GPU. For fine-tuning and training from scratch on downstream tasks, we adhere to the parameter settings of Point-MAE, as specified in Table 5. Specifically, we use rotation as data augmentation [4] in multiple tasks, and a higher dropout rate of 0.2/0.3 to better train MAMBA3D.

In Algorithm 1, we present the Pytorch-style pseudo-code for the Local Norm Pooling (LNP) block. The learnable parameters in LNP block include scale vectors γ and shift vectors β in linear transformation, and a shared MLP layer, utilizing only 0.3M parameters in total. For the bi-SSM block implementation, we’ve taken reference from the Mamba³ [5] and Vision Mamba⁴ [23].

¹<https://github.com/Pang-Yatian/Point-MAE>

²<https://github.com/lulutang0608/Point-BERT>

³<https://github.com/state-spaces/mamba>

⁴<https://github.com/hustvl/Vim>

Table 5: Training recipes for pre-training and downstream fine-tuning/training from scratch. Note that P-M and P-B represent Point-MAE and Point-BERT strategy, respectively.

Config	Pretraining (P-M/P-B)	Object Classification		Part Segmentation	Few-shot Learning
	ShapeNet [3]	ScanObjectNN [12]	ModelNet [18]	ShapeNetPart [20]	ModelNet [18]
optimizer	AdamW	AdamW	AdamW	AdamW	AdamW
learning rate	1e-3 / 5e-4	5e-4	5e-4	2e-4	5e-4
weight decay	5e-2	5e-2	5e-2	5e-2	5e-2
learning rate scheduler	cosine	cosine	cosine	cosine	cosine
training epochs	300	300	300	300	300
warmup epochs	10 / 3	10	10	10	10
batch size	128	32	32	16	32
drop path rate	0.1	0.1	0.2	0.1	0.3
number of points	1024	2048	1024	2048	1024
number of point patches	64	128	128	128	128
point patch size	32	32	32	32	32
augmentation	Rotation	Rotation	Scale&Trans	Scale&Center	Rotation
GPU device	RTX 3090	RTX 3090	RTX 3090	RTX 3090	RTX 3090

Algorithm 1: Pytorch-Style Pseudocode of the Local Norm Pooling (LNP) Block

```

1 # B: batch size
2 # L: number of tokens
3 # C: channel dimension
4 # k: number of local neighborhood in LNP
5 # input: central point feature F_C [B,L,C]
6 # output: updated token embeddings [B,L,C]
7
8 F_C = F_C.unsqueeze(2).repeat(1, 1, k, 1) # [B,L,k,C]
9 # local patch graph construction
10 F_K = kNN(F_C, k=4) - F_C # [B,L,k,C]
11 # K-norm
12 F_K_tilde = F_K / (std(F_K) + 1e-5) # [B,L,k,C]
13 F_K_widehat = concat([F_K_tilde, F_C], dim=-1) # [B,L,k,2C]
14 gamma = parameter(ones([1, 1, 1, 2C])) # [B,L,k,2C]
15 beta = parameter(zeros([1, 1, 1, 2C])) # [B,L,k,2C]
16 F_K_widehat = F_K_widehat * broadcast(gamma)
17               + broadcast(beta) # [B,L,k,2C]
18 # K-pooling
19 e_x = exp(F_K_widehat) # [B,L,k,2C]
20 F_C_widehat = (F_K_widehat * e_x).mean(-2)
21               / e_x.mean(-2) # [B,L,2C]
22 # share_mlp for dimension alignment
23 out = shared_mlp(F_C_widehat) # [B,L,C]
24 return out

```

D LIMITATIONS

While MAMBA3D demonstrates efficient learning of point cloud features, it exhibits certain limitations. Our observations indicate that pre-training strategies such as Point-BERT and Point-MAE, which rely on masked point modeling (MPM), provide limited performance gain for MAMBA3D or even lead to performance degradation. This arises from the fact that, despite point cloud sequences lacking inherent order, sequential models like Mamba still require some level of sequential information. However, random masking strategies like MPM struggle to adequately provide this sequential context,

thereby hindering the effectiveness of pre-training in enhancing Mamba’s learning capacity. Another limitation pertains to Mamba’s segmentation performance, which requires further enhancement. While it surpasses Transformer, it falls short of achieving state-of-the-art (SOTA) results. Exploring pyramid-like architectures holds promise for addressing this issue. Finally, while we’ve primarily employed single-modal pre-training strategies, most large-scale point cloud models are multimodal. We believe that integrating multimodal strategies could further enhance MAMBA3D.

REFERENCES

- [1] Mohamed Afham, Isuru Dissanayake, Dinithi Dissanayake, Amaya Dharmasiri, Kanchana Thilakarathna, and Ranga Rodrigo. 2022. CrossPoint: Self-Supervised Cross-Modal Contrastive Learning for 3D Point Cloud Understanding. In *IEEE/CVF Conf. Comput. Vis. Pattern Recog. (CVPR)*.
- [2] Bernhard E. Boser, Isabelle Guyon, and Vladimir Vapnik. 1992. A Training Algorithm for Optimal Margin Classifiers. In *ACM Conf. Comput. Learn. Theory (COLT)*. ACM, 144–152.
- [3] Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qi-Xing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. 2015. ShapeNet: An Information-Rich 3D Model Repository. *CoRR* abs/1512.03012 (2015). arXiv:1512.03012
- [4] Runpei Dong, Zekun Qi, Linfeng Zhang, Junbo Zhang, Jianjian Sun, Zheng Ge, Li Yi, and Kaisheng Ma. 2023. Autoencoders as Cross-Modal Teachers: Can Pretrained 2D Image Transformers Help 3D Representation Learning?. In *Int. Conf. Learn. Represent. (ICLR)*.
- [5] Albert Gu and Tri Dao. 2023. Mamba: Linear-Time Sequence Modeling with Selective State Spaces. *arXiv preprint arXiv:2312.00752* (2023).
- [6] Zhizhong Han, Mingyang Shang, Yu-Shen Liu, and Matthias Zwicker. 2019. View Inter-Prediction GAN: Unsupervised Representation Learning for 3D Shapes by Learning Global Shape Memories to Support Local View Predictions. In *AAAI Conf. Artif. Intell. (AAAI)*, Vol. 33. 8376–8384.
- [7] Jiaxin Li, Ben M Chen, and Gim Hee Lee. 2018. SO-Net: Self-Organizing Network for Point Cloud Analysis. In *IEEE/CVF Conf. Comput. Vis. Pattern Recog. (CVPR)*. 9397–9406.
- [8] Yatian Pang, Wenxiao Wang, Francis E. H. Tay, Wei Liu, Yonghong Tian, and Li Yuan. 2022. Masked Autoencoders for Point Cloud Self-supervised Learning. In *Eur. Conf. Comput. Vis. (ECCV)*.
- [9] Charles Ruizhongtai Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. 2017. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *IEEE/CVF Conf. Comput. Vis. Pattern Recog. (CVPR)*. 77–85.
- [10] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J. Guibas. 2017. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Adv. Neural Inform. Process. Syst. (NeurIPS)*. 5099–5108.

349	[11] Jonathan Sauder and Bjarne Sievers. 2019. Self-Supervised Deep Learning on Point Clouds by Reconstructing Space. In <i>Adv. Neural Inform. Process. Syst. (NeurIPS)</i> . 12942–12952.	
350		
351	[12] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. 2019. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In <i>IEEE/CVF Conf. Comput. Vis. Pattern Recog. (CVPR)</i> . 1588–1597.	
352		
353		
354	[13] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing Data using t-SNE. <i>J. Mach. Learn. Res. (JMLR)</i> 9, 11 (2008).	
355		
356	[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In <i>Adv. Neural Inform. Process. Syst. (NeurIPS)</i> . 5998–6008.	
357		
358	[15] Hanchen Wang, Qi Liu, Xiangyu Yue, Joan Lasenby, and Matt J Kusner. 2021. Unsupervised point cloud pre-training via occlusion completion. In <i>Int. Conf. Comput. Vis. (ICCV)</i> . 9782–9792.	
359		
360	[16] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. 2019. Dynamic Graph CNN for Learning on Point Clouds. <i>ACM Trans. Graph.</i> 38, 5 (2019), 146:1–146:12.	
361		
362	[17] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. 2016. Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling. <i>Adv. Neural Inform. Process. Syst. (NeurIPS)</i> 29 (2016).	
363		
364		
365		
366		
367		
368		
369		
370		
371		
372		
373		
374		
375		
376		
377		
378		
379		
380		
381		
382		
383		
384		
385		
386		
387		
388		
389		
390		
391		
392		
393		
394		
395		
396		
397		
398		
399		
400		
401		
402		
403		
404		
405		
406		
	[18] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 2015. 3d shapenets: A deep representation for volumetric shapes. In <i>IEEE/CVF Conf. Comput. Vis. Pattern Recog. (CVPR)</i> . 1912–1920.	407
		408
	[19] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. 2018. FoldingNet: Point Cloud Auto-Encoder via Deep Grid Deformation. In <i>IEEE/CVF Conf. Comput. Vis. Pattern Recog. (CVPR)</i> . 206–215.	409
		410
	[20] Li Yi, Vladimir G Kim, Duygu Ceylan, I-Chao Shen, Mengyan Yan, Hao Su, Cewu Lu, Qixing Huang, Alla Sheffer, and Leonidas Guibas. 2016. A scalable active framework for region annotation in 3d shape collections. <i>ACM Trans. Graph.</i> 35, 6 (2016), 1–12.	411
		412
		413
	[21] Xumin Yu, Lulu Tang, Yongming Rao, Tiejun Huang, Jie Zhou, and Jiwen Lu. 2022. Point-BERT: Pre-training 3D Point Cloud Transformers with Masked Point Modeling. In <i>IEEE/CVF Conf. Comput. Vis. Pattern Recog. (CVPR)</i> .	414
		415
	[22] Yongheng Zhao, Tolga Birdal, Haowen Deng, and Federico Tombari. 2019. 3D Point Capsule Networks. In <i>IEEE/CVF Conf. Comput. Vis. Pattern Recog. (CVPR)</i> . 1009–1018.	416
		417
		418
	[23] Lianghui Zhu, Bencheng Liao, Qian Zhang, Xinlong Wang, Wenyu Liu, and Xingang Wang. 2024. Vision Mamba: Efficient Visual Representation Learning with Bidirectional State Space Model. <i>arXiv preprint arXiv:2401.09417</i> (2024).	419
		420
		421
		422
		423
		424
		425
		426
		427
		428
		429
		430
		431
		432
		433
		434
		435
		436
		437
		438
		439
		440
		441
		442
		443
		444
		445
		446
		447
		448
		449
		450
		451
		452
		453
		454
		455
		456
		457
		458
		459
		460
		461
		462
		463
		464