Appendix

For a thorough understanding of our Point Transformer V3 (PTv3), we have compiled a detailed Appendix. The table of contents below offers a quick overview and will guide to specific sections of interest.

Contents

A. Limitations and Broader Impacts

Attention mechanisum. In prioritizing efficiency, PTv3 reverts to utilizing dot-product attention, which has been well-optimized through engineering efforts. However, we do note a reduction in convergence speed and a limitation in further scaling depth compared to vector attention. This issue also observed in recent advancements in transformer technology [93], is attributed to "attention sinks" stemming from the dot-product and softmax operations. Consequently, our findings reinforce the need for continued exploration of attention mechanisms.

Scaling parameters. PTv3 transcends the existing trade-offs between accuracy and efficiency, paving the way for investigating 3D transformers at larger parameter scales within given computational resources. While this exploration remains a topic for future work, current point cloud transformers already demonstrate an over-capacity for existing tasks. We advocate for a combined approach that scales up both the model parameters and the scope of data and tasks (e.g., learning from all available data, multi-task frameworks, and multi-modality tasks). Such an integrated strategy could fully unlock the potential of scaling in 3D representation learning.

Multiple modalities. Point cloud serialization provides a robust methodology for transforming n-dimensional data into a structured 1D format, effectively preserving spatial proximity. This technique can similarly be applied to image data, enabling its conversion into a language-style 1D structure that PTv3 can efficiently encode. This capability opens new avenues for the development of multimodal models that bridge 2D and 3D spaces, fostering opportunities for large-scale, synergistic pre-training that integrates both image and point cloud data.

Scratch		Joint Training [92]	
Config	Value	Config	Value
optimizer	AdamW	optimizer	AdamW
scheduler	Cosine	scheduler	Cosine
criteria	CrossEntropy (1) Lovasz [4] (1)	criteria	CrossEntropy (1) Lovasz [4] (1)
learning rate	5e-3	learning rate	5e-3
block lr scaler	0.1	block lr scaler	0.1
weight decay	5e-2	weight decay	5e-2
batch size	12	batch size	24
datasets	ScanNet / S3DIS / Struct.3D	datasets	ScanNet (2) S3DIS (1) Struct.3D (4)
warmup epochs	40	warmup iters	6k
epochs	800	iters	120k

Table 12. **Indoor semantic segmentation settings.**

Scratch		Joint Training [92]	
Config	Value	Config	Value
optimizer	AdamW	optimizer	AdamW
scheduler	Cosine	scheduler	Cosine
criteria	CrossEntropy (1) Lovasz [4] (1)	criteria	CrossEntropy (1) Lovasz [4] (1)
learning rate	2e-3	learning rate	2e-3
block lr scaler	1e-1	block lr scaler	1e-1
weight decay	5e-3	weight decay	5e-3
batch size	12	batch size	24
datasets	NuScenes / Sem.KITTI / Waymo	datasets	NuScenes (1) Sem.KITTI (1) Waymo (1)
warmup epochs	2	warmup iters	9k
epochs	50	iters	180k

Table 13. **Outdoor semantic segmentation settings.**

Ins. Seg.		Obj. Det	
Config	Value	Config	Value
framework	PointGroup [35]	framework	CenterPoint [102]
optimizer	AdamW	optimizer	Adam
scheduler	Cosine	scheduler	Cosine
learning rate	5e-3	learning rate	3e-3
block lr scaler	1e-1	block lr scaler	1e-1
weight decay	5e-2	weight decay	1e-2
batch size	12	batch size	16
datasets	ScanNet	datasets	Waymo
warmup epochs	40	warmup epochs	0
epochs	800	epochs	24

Table 14. **Other downstream tasks settings.**

B. Implementation Details

Our implementation primarily utilizes Pointcept [15], a specialized codebase focusing on point cloud perception and

Config	Value
serialization pattern	Z + TZ + H + TH
patch interaction	Shift Order + Shuffle Order
positional encoding	xCPE
embedding depth	2
embedding channels	32
encoder depth	[2, 2, 6, 2]
encoder channels	[64, 128, 256, 512]
encoder num heads	[4, 8, 16, 32]
encoder patch size	[1024, 1024, 1024, 1024]
decoder depth	[1, 1, 1, 1]
decoder channels	[64, 64, 128, 256]
decoder num heads	[4, 4, 8, 16]
decoder patch size	[1024, 1024, 1024, 1024]
down stride	[×2, ×2, ×2, ×2]
mlp ratio	4
qkv bias	True
drop path	0.3

Table 15. **Model settings.**

Augmentations	Parameters	Indoor	Outdoor
random dropout	dropout ratio: 0.2, p: 0.2	✓	-
random rotate	axis: z, angle: [-1, 1], p: 0.5	✓	✓
	axis: x, angle: [-1 / 64, 1 / 64], p: 0.5	✓	-
	axis: y, angle: [-1 / 64, 1 / 64], p: 0.5	✓	-
random scale	scale: [0.9, 1.1]	✓	✓
random flip	p: 0.5	✓	✓
random jitter	sigma: 0.005, clip: 0.02	✓	✓
elastic distort	params: [[0.2, 0.4], [0.8, 1.6]]	✓	-
auto contrast	p: 0.2	✓	-
color jitter	std: 0.05; p: 0.95	✓	-
grid sampling	grid size: 0.02 (indoor), 0.05 (outdoor)	✓	✓
sphere crop	ratio: 0.8, max points: 128000	✓	-
normalize color	p: 1	✓	-

Table 16. **Data augmentations.**

representation learning. For tasks involving outdoor object detection, we employ OpenPCDet [76], which is tailored for LiDAR-based 3D object detection. The specifics of our implementation are detailed in this section.

B.1. Training Settings

Indoor semantic segmentation. The settings for indoor semantic segmentation are outlined in Tab. 12. The two leftmost columns describe the parameters for training from scratch using a single dataset. To our knowledge, this represents the first initiative to standardize training settings across different indoor benchmarks with a unified approach. The two rightmost columns describe the parameters for multi-dataset joint training [92] with PTv3, maintaining similar settings to the scratch training but with an increased batch size. The numbers in brackets indicate the relative weight assigned to each dataset (criteria) in the mix.

Outdoor semantic segmentation. The configuration for outdoor semantic segmentation, presented in Tab. 13, follows a similar format to that of indoor. We also standardize the training settings across three outdoor datasets. Notably,

Block	BN	LN	BN	LN
Pooling	BN	LN	LN	BN
Perf.	76.7	76.1	75.6	**77.3**

Table 17. **Nomalization layer.**

Block	Traditional	Post-Norm	Pre-Norm
Perf.	76.6	72.3	**77.3**

Table 18. **Block structure.**

PTv3 operates effectively without the need for point clipping within a specific range, a step that is typically essential in current models. Furthermore, we extend our methodology to multi-dataset joint training with PTv3, employing settings analogous to scratch training but with augmented batch size. The numbers in brackets represent the proportional weight assigned to each dataset in the training mix.

Other Downstream Tasks. We outline our configurations for indoor instance segmentation and outdoor object detection in Tab. 14. For indoor instance segmentation, we use PointGroup [35] as our foundational framework, a popular choice in 3D representation learning [30, 91, 92, 94]. Our configuration primarily follows PointContrast [94], with necessary adjustments made for PTv3 compatibility. Regarding outdoor object detection, we adhere to the settings detailed in FlatFormer [51] and implement CenterPoint as our base framework to assess PTv3's effectiveness. It's important to note that PTv3 is versatile and can be integrated with various other frameworks due to its backbone nature.

B.2. Model Settings

As briefly described in Sec. 4.3, here we delve into the detailed model configurations of our PTv3, which are comprehensively listed in Tab. 15. This table serves as a blueprint for components within serialization-based point cloud transformers, encapsulating models like OctFormer [83] and FlatFormer [51] within the outlined frameworks, except for certain limitations discussed in Sec. 2. Specifically, OctFormer can be interpreted as utilizing a single z-order serialization with patch interaction enabled by Shift Dilation. Conversely, FlatFormer can be characterized by its window-based serialization approach, facilitating patch interaction through Shift Order.

B.3. Data Augmentations

The specific configurations of data augmentations implemented for PTv3 are detailed in Tab. 16. We unify augmentation pipelines for both indoor and outdoor scenarios separately, and the configurations are shared by all tasks within each domain. Notably, we observed that PTv3 does not depend on point clipping within a specific range, a process often crucial for existing models.

Methods	Year	Val	Test
○ PointNet++ [63]	2017	53.5	55.7
○ 3DMV [16]	2018	-	48.4
○ PointCNN [46]	2018	-	45.8
○ SparseConvNet [25]	2018	69.3	72.5
○ PanopticFusion [55]	2019	-	52.9
○ PointConv [88]	2019	61.0	66.6
○ JointPointBased [11]	2019	69.2	63.4
○ KPConv [77]	2019	69.2	68.6
○ PointASNL [97]	2020	63.5	66.6
○ SegGCN [44]	2020	-	58.9
○ RandLA-Net [32]	2020	-	64.5
○ JSENet [33]	2020	-	69.9
○ FusionNet [104]	2020	-	68.8
○ FastPointTransformer [58]	2022	72.4	-
○ SratifiedTranformer [40]	2022	74.3	73.7
○ PointNeXt [64]	2022	71.5	71.2
○ LargeKernel3D [9]	2023	73.5	73.9
○ PointMetaBase [47]	2023	72.8	71.4
○ PointConvFormer [89]	2023	74.5	74.9
○ OctFormer [83]	2023	75.7	76.6
○ Swin3D [101]	2023	77.5	77.9
• + Supervised [101]	2023	76.7	77.9
○ MinkUNet [13]	2019	72.2	73.6
• + PC [94]	2020	74.1	-
• + CSC [30]	2021	73.8	-
• + MSC [91]	2023	75.5	-
• + GC [81]	2024	75.7	-
• + PPT [92]	2024	76.4	76.6
○ OA-CNNs [60]	2024	76.1	75.6
○ PTv1 [106]	2021	70.6	-
○ PTv2 [90]	2022	75.4	74.2
○ PTv3 (Ours)	2024	77.5	77.9
• + PPT [92]	2024	**78.6**	**79.4**

Table 19. **ScanNet V2 semantic segmentation.**

Methods	Year	Area5	6-fold
○ PointNet [62]	2017	41.1	47.6
○ SegCloud [75]	2017	48.9	-
○ TanConv [74]	2018	52.6	-
○ PointCNN [46]	2018	57.3	65.4
○ ParamConv [85]	2018	58.3	-
○ PointWeb [105]	2019	60.3	66.7
○ HPEIN [34]	2019	61.9	-
○ KPConv [77]	2019	67.1	70.6
○ GACNet [82]	2019	62.9	-
○ PAT [100]	2019	60.1	-
○ SPGraph [42]	2018	58.0	62.1
○ SegGCN [44]	2020	63.6	-
○ PAConv [96]	2021	66.6	-
○ StratifiedTransformer [40]	2022	72.0	-
○ PointNeXt [64]	2022	70.5	74.9
○ SuperpointTransformer [65]	2023	68.9	76.0
○ PointMetaBase [47]	2023	72.0	77.0
○ Swin3D [101]	2023	72.5	76.9
• + Supervised [101]	2023	74.5	79.8
○ MinkUNet [13]	2019	65.4	65.4
• + PC [94]	2020	70.3	-
• + CSC [30]	2021	72.2	-
• + MSC [91]	2023	70.1	-
• + GC [81]	2024	72.0	-
• + PPT [92]	2024	72.7	78.1
○ PTv1 [106]	2021	70.4	65.4
○ PTv2 [90]	2022	71.6	73.5
○ PTv3 (Ours)	2024	73.4	77.7
• + PPT [92]	2024	**74.7**	**80.8**

Table 20. **S3DIS semantic segmentation.**

C. Additional Ablations

In this section, we present further ablation studies focusing on macro designs of PTv3, previously discussed in Sec. 4.3.

C.1. Nomalization Layer

Previous point transformers employ Batch Normalization (BN), which can lead to performance variability depending on the batch size. This variability becomes particularly problematic in scenarios with memory constraints that require small batch sizes or in tasks demanding dynamic or varying batch sizes. To address this issue, we have gradually transitioned to Layer Normalization (LN). Our final, empirically determined choice is to implement Layer Normalization in the attention blocks while retaining Batch Normalization in the pooling layers (see Tab. 17).

C.2. Block Structure

Previous point transformers use a traditional block structure that sequentially applies an operator, a normalization layer, and an activation function. While effective, this approach can sometimes complicate training deeper models due to issues like vanishing gradients or the need for careful initialization and learning rate adjustments [95]. Consequently, we explored adopting a more modern block structure, such as pre-norm and post-norm. The pre-norm structure, where a normalization layer precedes the operator, can stabilize training by ensuring normalized inputs for each layer [12]. In contrast, the post-norm structure places a normalization layer right after the operator, potentially leading to faster convergence but with less stability [80]. Our experimental results (see Tab. 18) indicate that the pre-norm structure is more suitable for our PTv3, aligning with findings in recent transformer-based models [95].

D. Additional Comparision

In this section, we expand upon the combined results table for semantic segmentation (Tab. 5 and Tab. 7) from our main paper, offering a more detailed breakdown of results alongside the respective publication years of previous works. This comprehensive result table is designed to assist readers in tracking the progression of research efforts in 3D representation learning. Marker ○ refers to the result from a model trained from scratch, and • refers to the result from a pre-trained model.

Methods	Year	Val	Test
○ SPVNAS [73]	2020	64.7	66.4
○ Cylinder3D [108]	2021	64.3	67.8
○ PVKD [31]	2022	-	71.2
○ 2DPASS [98]	2022	69.3	72.9
○ WaffleIron [61]	2023	68.0	70.8
○ SphereFormer [41]	2023	67.8	74.8
○ RangeFormer [39]	2023	67.6	73.3
○ MinkUNet [13]	2019	63.8	-
• + PPT [92]	2024	71.4	-
○ OA-CNNs [60]	2024	70.6	-
○ PTv2 [90]	2022	70.3	72.6
○ PTv3 (Ours)	2024	70.8	74.2
• + M3Net [48]	2024	72.0	75.1
• + PPT [92]	2024	**72.3**	**75.5**

Table 21. **SemanticKITTI semantic segmentation.**

Methods	Year	Val	Test
○ SPVNAS [73]	2020	77.4	-
○ Cylinder3D [108]	2021	76.1	77.2
○ PVKD [31]	2022	-	76.0
○ 2DPASS [98]	2022	-	80.8
○ SphereFormer [41]	2023	78.4	81.9
○ RangeFormer [39]	2023	78.1	80.1
○ MinkUNet [13]	2019	73.3	-
• + PPT [92]	2024	78.6	-
○ OA-CNNs [60]	2024	78.9	-
○ PTv2 [90]	2022	80.2	82.6
○ PTv3 (Ours)	2024	80.4	82.7
• + M3Net [48]	2024	80.9	**83.1**
• + PPT [92]	2024	**81.2**	83.0

Table 22. **NuScenes semantic segmentation.**

D.1. Indoor Semantic Segmentation

We conduct a detailed comparison of pre-training technologies and backbones on the ScanNet v2 [17] (see Tab. 19) and S3DIS [2] (see Tab. 20) datasets. ScanNet v2 comprises 1,513 room scans reconstructed from RGB-D frames, divided into 1,201 training scenes and 312 for validation. In this dataset, model input point clouds are sampled from the vertices of reconstructed meshes, with each point assigned a semantic label from 20 categories (e.g., wall, floor, table). The S3DIS dataset for semantic scene parsing includes 271 rooms across six areas from three buildings. Following a common practice [63, 75, 106], we withhold area 5 for testing and perform a 6-fold cross-validation. Different from ScanNet v2, S3DIS densely sampled points on mesh surfaces, annotated into 13 categories. Consistent with standard practice [63]. We employ the mean class-wise intersection over union (mIoU) as the primary evaluation metric for indoor semantic segmentation.

D.2. Outdoor Semantic Segmentation

We extend our comprehensive evaluation of pre-training technologies and backbones to outdoor semantic segmentation tasks, focusing on the SemanticKITTI [3](see Tab. 21) and NuScenes [5] (see Tab. 22) datasets. SemanticKITTI is derived from the KITTI Vision Benchmark Suite and consists of 22 sequences, with 19 for training and the remaining 3 for testing. It features richly annotated LiDAR scans, offering a diverse array of driving scenarios. Each point in this dataset is labeled with one of 28 semantic classes, encompassing various elements of urban driving environments. NuScenes, on the other hand, provides a large-scale dataset for autonomous driving, comprising 1,000 diverse urban driving scenes from Boston and Singapore. For outdoor semantic segmentation, we also employ the mean class-wise intersection over union (mIoU) as the primary evaluation metric for outdoor semantic segmentation.