Point Transformer V3 Extreme: 1st Place Solution for 2024 Waymo Open Dataset Challenge in Semantic Segmentation

Xiaoyang Wu[1,2] Xiang Xu[2] Lingdong Kong[3]
Liang Pan[4] Ziwei Liu[4] Tong He[2] Wanli Ouyang[2] Hengshuang Zhao[1]

[1]HKU [2]SH AI Lab [3]NUS [4]NTU

https://github.com/Pointcept/Pointcept

Abstract

In this technical report, we detail our first-place solution for the 2024 Waymo Open Dataset Challenge's semantic segmentation track. We significantly enhanced the performance of Point Transformer V3 on the Waymo benchmark by implementing cutting-edge, plug-and-play training and inference technologies. Notably, our advanced version, Point Transformer V3 Extreme, leverages multi-frame training and a no-clipping-point policy, achieving substantial gains over the original PTv3 performance. Additionally, employing a straightforward model ensemble strategy further boosted our results. This approach secured us the top position on the Waymo Open Dataset semantic segmentation leaderboard, markedly outperforming other entries.

1. Introduction

In recent years, the Waymo Open Dataset Challenge [12] has emerged as a premier arena for showcasing advancements in autonomous driving technologies. The 2024 iteration of this challenge continued to push the boundaries of what is achievable in 3D perception, leveraging the rich and diverse data provided by Waymo. The Waymo Open Dataset is characterized by its high-resolution LiDAR scans and comprehensive annotations, making it ideal for developing and testing cutting-edge 3D perception algorithms. This technical report presents our winning entry for the 2024 Waymo Open Dataset Challenges semantic segmentation track. Our approach builds upon the foundation of the Point Transformer V3 (PTv3) [15], known for its robustness and efficiency in handling point cloud data. We optimized PTv3 for the specific challenges posed by the Waymo dataset, implementing several plug-and-play training and inference technologies that significantly enhance performance.

Key to our strategy was the implementation of multi-frame training, which incorporates data from two previous frames to enrich the perception of current LiDAR frames.

Original		Extreme	
Config	Value	Config	Value
optimizer	AdamW	optimizer	AdamW
scheduler	Cosine	scheduler	Cosine
criteria	CrossEntropy (1) Lovasz [1] (1)	criteria	CrossEntropy (1) Lovasz [1] (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	12
datasets	Waymo	datasets	Waymo
warmup epochs	2	warmup epochs	2
epochs	50	epochs	50
frames	[0]	frames	[0, -1, -2]
model ensemble	×	model ensemble	✓

Table 1. **Training settings.**

Augmentations	Parameters	Original	Extreme
random rotate	axis: z, angle: [-1, 1], p: 0.5	✓	✓
point clip	range: [-75.2, -75.2, -4, 75.2, 75.2, 2]	✓	×
random scale	scale: [0.9, 1.1]	✓	✓
random flip	p: 0.5	✓	✓
random jitter	sigma: 0.005, clip: 0.02	✓	✓
grid sampling	grid size: 0.05	✓	✓

Table 2. **Data augmentations.**

This technique, combined with a no-clipping-point policy that avoids discarding data points outside a specified range, provided a deeper insight into the spatial and temporal aspects of the dataset. This enhanced version, termed Point Transformer V3 Extreme, achieved substantial performance improvements over the original PTv3 metrics reported in earlier works. Furthermore, by incorporating a simple yet effective model ensemble strategy, we were able to achieve unprecedented accuracy, securing the first-place position on the semantic segmentation leaderboard. The detailed parameter settings are presented in Tab. 1 and Tab. 2. Our methods outperformed other competitive entries by remarkable margins, demonstrating the potential of advanced transformer architectures in complex, real-world environments like those represented in the Waymo Open Dataset.

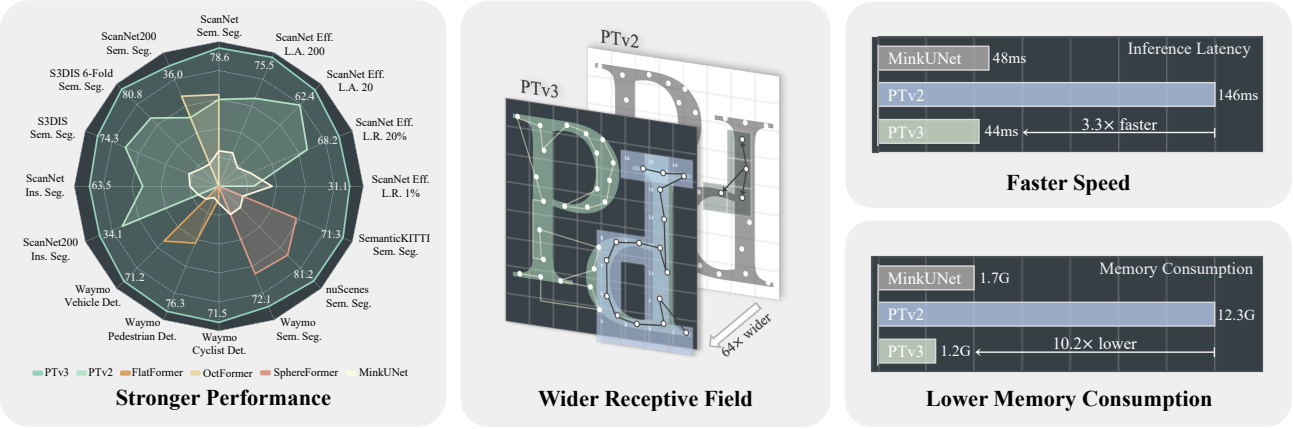

Figure 1. **Overview of Point Transformer V3 (PTv3).** Compared to its predecessor, PTv2 [14], our PTv3 shows superiority in the following aspects: 1. *Stronger performance.* PTv3 achieves state-of-the-art results across a variety of indoor and outdoor 3D perception tasks. 2. *Wider receptive field.* Benefit from the simplicity and efficiency, PTv3 expands the receptive field from 16 to 1024 points. 3. *Faster speed.* PTv3 significantly increases processing speed, making it suitable for latency-sensitive applications. 4. *Lower Memory Consumption.* PTv3 reduces memory usage, enhancing accessibility for broader situations.

2. Point Transformer V3 Extreme

This section first has a revisit of the Point Transformer V3 in Sec. 2.1. After that, we go through the details of additional training technologies in Sec. 2.2. The detailed parameter settings are presented in Tab. 1 and Tab. 2.

2.1. Revisit Point Transformer V3

Scaling principle. Enhanced with large-scale pre-training, SparseUNet [2] surpasses Point Transformers [14, 17] in accuracy while remaining efficient. Yet, Point Transformers fails to scale up due to limitations in efficiency, which inspired the hypothesis that model performance is more significantly influenced by scale than by complex design details. Backbone design should prioritize simplicity and efficiency over the accuracy of certain mechanisms. Efficiency enables scalability, which further brings a stronger accuracy.

Breaking the curse of permutation invariance Classical point cloud transformers build upon point-based backbones [10, 11], which treat point clouds as unstructured data and rely on neighboring query algorithms like the k-nearest neighbor (kNN). Yet kNN is extremely inefficient due to the difficulty in parallelization, which further raises the question of whether we really need the accurate neighbours queried by kNN. Considering that attention is adaptive to kernel shape, it is worth trading the accurate spatial proximity for additional scalability. Inspired by OctFormer [13] and FlatFormer [9], PTv3 abandoned the unstructured nature of the point cloud, exploring a strategy to turn unstructured sparse data into structured 1D data as language tokens while preserving necessary spatial proximity to attention.

Serialization & attention. Space-filling curves are paths that traverse every point within a high-dimensional discrete space, preserving spatial proximity to a certain extent. The serialization of point clouds involves sorting points according to the traversal order defined by a specific space-filling curve. This ordering effectively rearranges the points in a way that respects the spatial organization dictated by the curve, ensuring that neighboring points in the data structure are also spatially close. By reordering point clouds through serialization and incorporating necessary padding operations, the unordered point cloud is transformed into

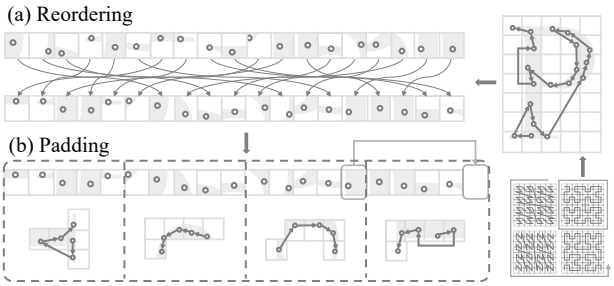

Figure 2. **Patch grouping.** (a) Reordering point cloud according to order derived from a specific serialization pattern. (b) Padding point cloud sequence by borrowing points from neighboring patches to ensure it is divisible by the designated patch size.

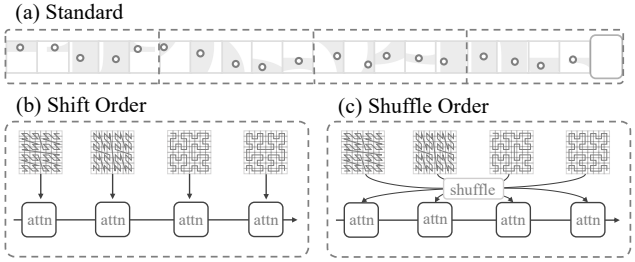

Figure 3. **Patch interaction.** (a) Standard patch grouping with a regular, non-shifted arrangement; (d) Shift Order where different serialization patterns are cyclically assigned to successive attention layers; (d) Shuffle Order, where the sequence of serialization patterns is randomized before being fed to attention layers.

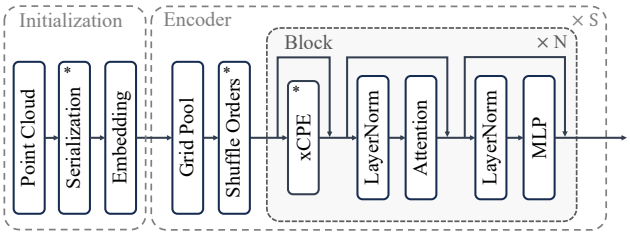

Figure 4. **Overall architecture.**

a structured format (see Fig. 2). Consequently, attention mechanisms optimized for structured data can be effectively applied to these serialized point clouds. To optimize performance across various benchmarks, PTv3 employs both local attention [8] and flash attention [4, 5]. For local attention, PTv3 facilitates patch interaction by utilizing various serialization patterns across different attention layers (see Fig. 3). Additionally, PTv3 adopts a sparse convolution layer, prepended with a skip connection, as conditional positional encoding [3, 13], named xCPE. The overall model architecture is visualized in Fig. 4.

2.2. Training Technologies

Multi-frame training. Perceiving distant ranges in a LiDAR point cloud, far from the center, is challenging due to insufficient sampling. An intuitive solution is to concatenate past LiDAR frames with the current frame after a coordinate alignment to supplement the less-sampled areas (see Fig. 5). Specifically, we incorporate two past labelled frames as additional references during both our training and inference processes, utilizing all of them for supervision during training for convenience.

Non-clipping proxy. However, merely enabling multi-frame training does not automatically result in significant enhancements for perception tasks. We have found that the full potential of multi-frame training is unlocked only when it is combined with a non-clipping strategy. Traditionally, clipping points to a specific range, such as [-75.2, -75.2, -4, 75.2, 75.2, 2] for the Waymo Open Dataset, was a necessary preprocessing step for perception tasks in outdoor scenarios. This was largely because the perception systems [6, 7, 16] for autonomous driving, which often rely on submanifold sparse convolution, struggle to effectively incorporate isolated points that frequently occur at distant ranges in open-space LiDAR point clouds. Unlike these systems, PTv3, which organizes point clouds into a structured 1D array, does not suffer from this disadvantage. Without the limitations imposed by a clipping proxy, PTv3 effectively leverages additional information from past frames, which significantly enhances the semantic segmentation mIoU on the Waymo Open Dataset validation split from 72.1% to 74.8%.

Model ensemble. One technique that consistently boosts model performance is model ensembling. In our approach, we independently train three PTv3 models and combine their predicted logits to form our final submission. It's important to note that we discourage using this technique for performance comparisons, especially on the validation split, as it can lead to unfair comparisons. We have limited the use of this technology to the Waymo Challenge test split. We also advise future researchers to refrain from using this technique for validation comparisons.

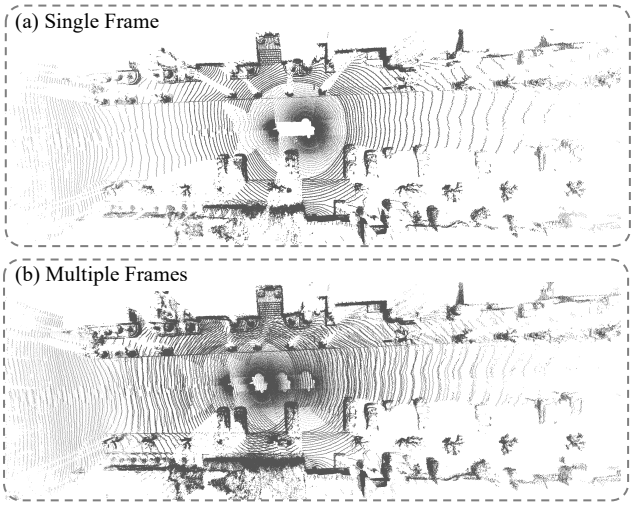

Figure 5. **Visualization of Multi-frames Concatenation.**

Sem. Seg.	PTv3 [15]		PTv3-EX	
	val	test	val	test
Model Ensemble	-	✓	-	✓
Params.	46.2M	46.2M×3	46.2M	46.2M×3
Training Latency	245ms	245ms×3	482ms	482ms×3
Inference Latency	132ms	132ms×3	253ms	253ms×3
Car	0.9447	0.9571	0.9463	0.9662
Truck	0.6207	0.6793	0.6283	0.7397
Bus	0.8665	0.7482	0.8920	0.7792
Other Vehicle	0.3582	0.3654	0.4857	0.3681
Motorcyclist	0.1630	0.0000	0.3946	0.1514
Bicyclist	0.7878	0.9010	0.8030	0.9203
Pedestrian	0.9120	0.9264	0.9162	0.9372
Sign	0.7235	0.7404	0.7664	0.7502
Traffic Light	0.3607	0.3373	0.4276	0.3465
Pole	0.7778	0.8157	0.8036	0.8254
Construction Cone	0.7562	0.6690	0.7405	0.6693
Bicycle	0.7821	0.6851	0.7772	0.7226
Motorcycle	0.9034	0.8070	0.9154	0.8263
Building	0.9606	0.9736	0.9636	0.9751
Vegetation	0.9189	0.8812	0.9242	0.8901
Tree Trunk	0.6860	0.7500	0.7069	0.7575
Curb	0.7152	0.7520	0.7226	0.7648
Road	0.9348	0.9306	0.9368	0.9330
Lane Marker	0.5712	0.4967	0.5726	0.5111
Other Ground	0.5206	0.5255	0.5248	0.5414
Walkable	0.8167	0.7357	0.8196	0.7538
Sidewalk	0.7872	0.8733	0.7891	0.8788
mIoU	0.7213	0.7068	**0.7480**	**0.7276**

Table 3. **Results on Waymo Open Dataset.** Latency and memory usage were assessed on a single RTX 4090 GPU, with the batch size fixed at 1 and models are trained with 4 NVIDIA a100 GPUs.

3. Conclusion

Enhanced with multi-frame training, a non-clipping strategy, and model ensembling, we have significantly extended the capabilities of Point Transformer V3. Specifically, on the Waymo Open Dataset, the validation mIoU increased from 72.1% to 74.8%, and the test mIoU rose from 70.7% to 72.8% (details provided in Tab. 3). We hope these technologies and results will inspire future research.

References

[1] Maxim Berman, Amal Rannen Triki, and Matthew B Blaschko. The lovász-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks. In *CVPR*, pages 4413–4421, 2018. 1

[2] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *CVPR*, 2019. 2

[3] Xiangxiang Chu, Zhi Tian, Bo Zhang, Xinlong Wang, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Conditional positional encodings for vision transformers. *arXiv:2102.10882*, 2021. 3

[4] Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv:2307.08691*, 2023. 3

[5] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *NeurIPS*, 2022. 3

[6] Lue Fan, Ziqi Pang, Tianyuan Zhang, Yu-Xiong Wang, Hang Zhao, Feng Wang, Naiyan Wang, and Zhaoxiang Zhang. Embracing single stride 3d object detector with sparse transformer. In *CVPR*, 2022. 3

[7] Chao Ma Guangsheng Shi, Ruifeng Li. Pillarnet: Real-time and high-performance pillar-based 3d object detection. *ECCV*, 2022. 3

[8] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *ICCV*, 2021. 3

[9] Zhijian Liu, Xinyu Yang, Haotian Tang, Shang Yang, and Song Han. Flatformer: Flattened window attention for efficient point cloud transformer. In *CVPR*, 2023. 2

[10] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017. 2

[11] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, 2017. 2

[12] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *CVPR*, 2020. 1

[13] Peng-Shuai Wang. Octformer: Octree-based transformers for 3D point clouds. *SIGGRAPH*, 2023. 2, 3

[14] Xiaoyang Wu, Yixing Lao, Li Jiang, Xihui Liu, and Hengshuang Zhao. Point transformer v2: Grouped vector attention and partition-based pooling. In *NeurIPS*, 2022. 2

[15] Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point transformer v3: Simpler, faster, stronger. In *CVPR*, 2024. 1, 3

[16] Tianwei Yin, Xingyi Zhou, and Philipp Krähenbühl. Center-based 3d object detection and tracking. In *CVPR*, 2021. 3

[17] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. Point transformer. In *ICCV*, 2021. 2