Multi-Scale Neighborhood Occupancy Masked Autoencoder for Self-Supervised Learning in LiDAR Point Clouds

Supplementary Material

Table 7. Training settings for semantic segmentation NOMAE pretraining and fine-tuning

Pretraining		SemSeg [38]	
Config	Value	Config	Value
optimizer	AdamW	optimizer	AdamW
scheduler	Cosine	scheduler	Cosine
criteria	BCE (1)	criteria	CE (1) Lovasz (1)
learning rate	2e-3	learning rate	2e-3
block lr scaler	0.1	block lr scaler	0.1
weight decay	5e-2	weight decay	5e-2
batch size	8	batch size	12
Datasets	nuScenes / Waymo	Datasets	nuScenes / Waymo
Masking Ratio	0.7 / 0.85	Masking Ratio	- / -
HMG	$[5, 10, 20, 40]$cm	HMG	-
LLRD	-	LLRD	0.65
warmup epochs	2	warmup epochs	2
epochs	50	epochs	50

Table 8. Training settings for object detection NOMAE pretraining and finetuning

Pretraining		ObjDet [23]	
Config	Value	Config	Value
optimizer	AdamW	optimizer	AdamW
scheduler	Cosine	scheduler	Cosine
criteria	BCE (1)	criteria	Focal loss (2) L1 loss (0.25)
learning rate	2e-3	learning rate	1e-3
block lr scaler	0.1	block lr scaler	0.2
weight decay	5e-2	weight decay	1e-2
batch size	8	batch size	16
Datasets	nuScenes	Datasets	nuScenes
Size	100%		20%
Masking Ratio	0.85	Masking Ratio	-
HMG	$[5, 10, 20, 40]cm$	HMG	-
LLRD	-	LLRD	0.65
warmup epochs	2	warmup epochs	8
epochs	50	epochs	20

6. Implementation Details

In this section, we report further details of our implementation for both pretraining and finetuning. We use the Pointcept [11] framework for pretraining and finetuning semantic segmentation. We performed the fine-tuning for object detection in the MMDetection3D framework [10].

6.1. Training Settings

In Tab. 7, we report the training details for both pretraining and semantic segmentation finetuning. For the criteria, the weights between brackets represent the weight of the loss in the final objective. During finetuning, we employ LLRD [17], more specifically, we decay the learning rate of every block in the encoder exponentially with a factor of 0.65 compared to the next block. For the downsampling blocks, we utilize the same learning rate as the next transformer block in the architecture.

In Tab. 8, we highlight the pretraining and finetuning details for object detection experiments. We follow the standard 10 sweeps aggregation for nuScenes object detection [23]. To this end, we pretrain NOMAE on the aggregated point clouds and use a masking ratio of 85% to accommodate for the denser point cloud. For other training settings, we utilize the same implementation details as our object detection baseline UVTR [23].

6.2. Model Settings

For the encoder $\mathbf{E}$, we use the same implementation details as the Encoder of PTv3 [37]. For the upsampling module $\mathbf{M}_u$, we use a PTv3 decoder as a baseline and experiment with different sizes. We found empirically that reducing the size of the upsampling module improves the downstream performance after finetuning. Hence, we utilize a single transformer block per resolution in the upsampling module.

The neighboring decoder uses sparse convolution layers to generate representations for neighboring voxels $\mathcal{V}_n$ and a single sub-manifold convolution layer to convert the representation into occupancy predictions. Sec. 7 gives further details of the hyperparameter choices of the neighborhood decoder.

6.3. Data Augmentations

We adopt common data augmentation methods used in [23, 37] during fine-tuning semantic segmentation and object detection. Pretraining uses the same data augmentations as semantic segmentation finetuning. Details about the data augmentations used are reported in Tab. 10.

7. Further Ablations

In this section, we present further ablation studies on the nuScenes semantic segmentation validation set. We evaluate the settings by freezing weights in the encoder after pretraining and using NonLP as described in Sec. 4.2.

Table 9. Detailed results on NuScenes val and test set compared with the baseline PTv3 [37].

Dataset	Method	mIoU	fwIoU	barrier	bicycle	bus	car	construction	motocycle	pedstrain	Traffic cone	trailer	truck	drivable	Other flat	sidewalk	terrian	manmade	vegetation
Val	PTv3(Baseline)	80.4	-	**81**	54	96	92	52	89	84	72	**74**	85	**97**	**76**	**77**	76	91	90
	NOMAE (Ours)	**81.8**	-	**81**	**56**	**97**	**95**	**63**	**90**	**85**	**73**	**74**	**88**	**97**	75	**77**	**77**	**92**	**90**
Test	PTv3(Baseline)	**82.7**	91.1	83	**72**	**93**	92	**71**	**90**	83	77	86	75	**98**	**69**	81	77	92	89
	NOMAE (Ours)	82.6	**91.5**	**87**	50	92	**93**	**71**	**90**	**86**	**82**	**89**	**77**	**98**	**69**	**83**	**78**	**93**	**90**

Table 10. Data augmentation details for NOMAE pretraining and finetuning for both semantic segmentation and object detection.

Augmentations	Parameters	Pretrain	SemSeg	ObjDet
random rotate	{axis: z, p: 0.5, angle: [-1, 1]π}	✓	✓	-
random rotate	{axis: z, p: 1.0, angle: [-1 / 8, 1 / 8]π}	-	-	✓
random scale	scale: [0.9, 1.1]	✓	✓	-
random scale	scale: [0.95, 1.05]	-	-	✓
random flip	p: 0.5	✓	✓	✓
random jitter	sigma: 0.005, clip: 0.02	✓	✓	-
point clip	range: [-51.2,-51.2,-5, 51.2,51.2,3]m	-	-	✓

Table 11. Results with varying batch size during NOMAE pretraining step.

	Batch Size					
	4	**8**	**16**	**32**	**64**	**128**
Single Scale $2^s = 1$	-	-	64.9	64.3	63.8	61.9
NOMAE	73.8	**74.8**	73.5	73.3	-	-

Pretraining Batch Size: This experiment investigates the effect of different batch sizes for our SSL task. Tab. 11 compares the downstream performance of models pretrained with different batch sizes and the same number of epochs (leading to a different number of parameter update steps). It can be observed that for NOMAE, the performance increases with decreasing the batch size (even for a single scale pretraining), and it reaches a maximum of 8 examples per batch before further decreasing. We attribute the higher performance at lower batch sizes to an increased number of parameter update steps. The results also indicate that NOMAE can benefit from a longer training schedule.

Neighboring Decoder Design: This experiment investigates the effect of design choices for the neighboring decoder. Fig. 7 shows the general architecture of the neighboring decoder. It consists of multiple sparse convolution layers to yield representations for $\mathcal{V}_\text{n}$ from $\mathcal{V}_\text{v}$, followed by submanifold convolution layers to predict the occupancy from the representation.

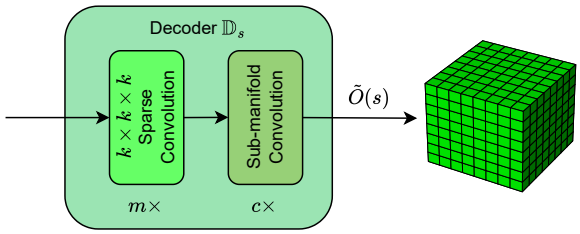

Figure 7. General architecture of a Neighborhood occupancy decoder $\mathbb{D}_s$ at scale s.

First, we vary both the network depth m and the size of the neighborhood expansion per layer k (layer receptive field). The total size of the neighborhood is given by

$$n = 2m(k-1) + 1. \tag{5}$$

Tab. 12 shows that a shallower decoder helps to learn a better representation compared to a deeper one with the same neighborhood size.

Furthermore, we investigate the number of sparse submanifold convolution layers c in the decoder. $c = 0$ indicates that the sparse convolutions must directly predict the neighboring occupancy $\tilde{O}(s)$. Tab. 13 shows that having at least one sparse submanifold convolution layer improves the downstream performance. Adding more layers performs on par with the single layer despite the extra computation cost.

7.1. Detailed Results

In this section, we report detailed results for nuScenes semantic segmentation on the test set, in comparison to the validation set. In particular, we examine the influence of the class *bicycle* on the overall performance and its impact on the test mIoU score.

In Tab. 9, we observe that NOMAE outperforms the previous baseline PTv3 [37] on almost all classes. The performance of the *bicycle* class is lower than for other classes for both PTv3 and NOMAE. We attribute this to *bicycle* being a rare class in the nuScenes dataset. On the test set, however, state-of-the-art approaches achieve relatively high scores for the *bicycle* class, thereby optimizing the mIoU score, which

Table 12. Results with varying k and m in the neighborhood decoders during NOMAE pretraining.

	m1k2 n=3	**m2k2 n=5**	**m3k2 n=7**	**m4k2 n=9**	**m1k3 n=5**	**m2k3 n=9**
mIoU	69.7	71.2	72.4	72.8	72.5	73.3

Table 13. Results with different neighborhood decoder head depth c during NOMAE pretraining.

	c=0	**c=1**	**c=2**
mIoU	72.7	73.3	73.2

averages class-wise results irrespectively of the object's frequency in the dataset. We did not take any special measures to address this for minority classes during fine-tuning NOMAE. As a consequence, the final mIoU score of NOMAE is on par but slightly lower than the best leaderboard submission, while achieving similar or better performance than PTv3 [37] on the other classes. As a result, NOMAE sets the new state-of-the-art regarding frequency-weighted IoU (fwIoU) on the test set.

8. Generalization to Different Architectures

In this section, we present more quantitative results on the Nuscenes Semseg task for pretraining and finetuning different architecutres (MinkUnet [9], OACNN [28] and OctFormer [35]). We choose another transformer based architecture and two different CNN based ones. For fair comparison with the older architectures we reproduce their results for training from scratch using the modern optimizers and schedulers which already improves their performance by whopping 5.3 points. For the modern ones, the results are consistent with the original works. In Tab. 14 we can see that NOMAE consistently improves the downstream performance by $\sim 1.4 - 2.2$ mIoU points compared to training from scratch. It's worth mentioning that such improvement comes without tuning any of the pretraining hyperparameters, demonstrating the superb generalization capabilities of NOMAE pretraining. We hypothesize that the generalization capabilities arises from the diverse feature learning promoted by our MSP (see Sec. 3.3)

Model	Scratch(paper)		Scratch(ours)		+ NOMAE	
	mIoU	**mACC**	**mIoU**	**mACC**	**mIoU**	**mACC**
MinkUnet [9]	73.3	-	78.6	83.9	80.1	86.2
OACNNs [28]	78.9	-	78.8	85.8	80.9	86.4
Octformer [35]	-	-	79.4	87.0	81.6	**88.8**
PTv3[37]	80.4	-	80.4	87.3	**81.8**	87.7

Table 14. Results of pretraining and finetuning different architectures on Nuscenes Semseg val set compared to training from scratch.

9. Multi-Scale SSL

In this section we compare between our formulation of multi-scale SSL (assigning different granularity reconstruction targets to different feature levels at different scales, and the formulation of multi-scale SSL in GeoMAE [33] (assigning different granularity reconstruction targets to a single feature level). Intuitively, our formulation encourages feature diversity between different feature levels. While GeoMAE's formulation improves the semantics of a single layer level, yet it doesn't promote feature diversity between different layers and can make said layer a bottleneck. In Tab 15 we quantify the improvement from each formulation using neighborhood occupancy as the pretext. The results show that the gain from MSP is $\sim 3\times$ GeoMAE's multi-scale (MS) on nuScenes Semseg val set. This demonstrates the importance of assigning different tasks to different feature levels for an effective SSL on large-scale point clouds.

	SS $2^s = 8$ NOMAE	+GEO-MAE's MS	+our MSP
NonLP mIoU	68.3	69.7	**72.6**

Table 15. Results on nuScenes val set for NonLP mIoU using Neighbourhood occupancy as pretext task and different Multi-Scale formulations

10. NOMAE Improves Sample Efficiency

The results of our baseline PTv3 in Tab. 1 are taken from the original paper [37], To demonstrate that the improvement from NOMAE pretraining is not merely due to faster convergence of the pretrained weights, we report additional results for training the model from scratch using longer schedules of 50, 100, and 200 epochs. The results in Tab. 16 show no significant improvement from longer training, confirming that just 50 pretraining epochs of NOMAE followed by 50 epochs of finetuning improves sample efficiency and reaches higher final performance. We would like to highlight to the reader that we did not experiment with longer pretraining schedules and we suspect that further improvement can be brought by it.

PTv3Sup,50 ep [37]	Sup, 100ep	Sup, 200ep	NOMAE+50ep
80.4	80.7	80.6	**81.8**

Table 16. Results on Nuscenes SemSeg val set for longer training from scratch compared to NOMAE pretraining

11. Additional Object Detection Results

In this section we report more results on the object detection downstream task. We use the same pretrained model as in Sec. 4.4, however we finetune the model using 100%

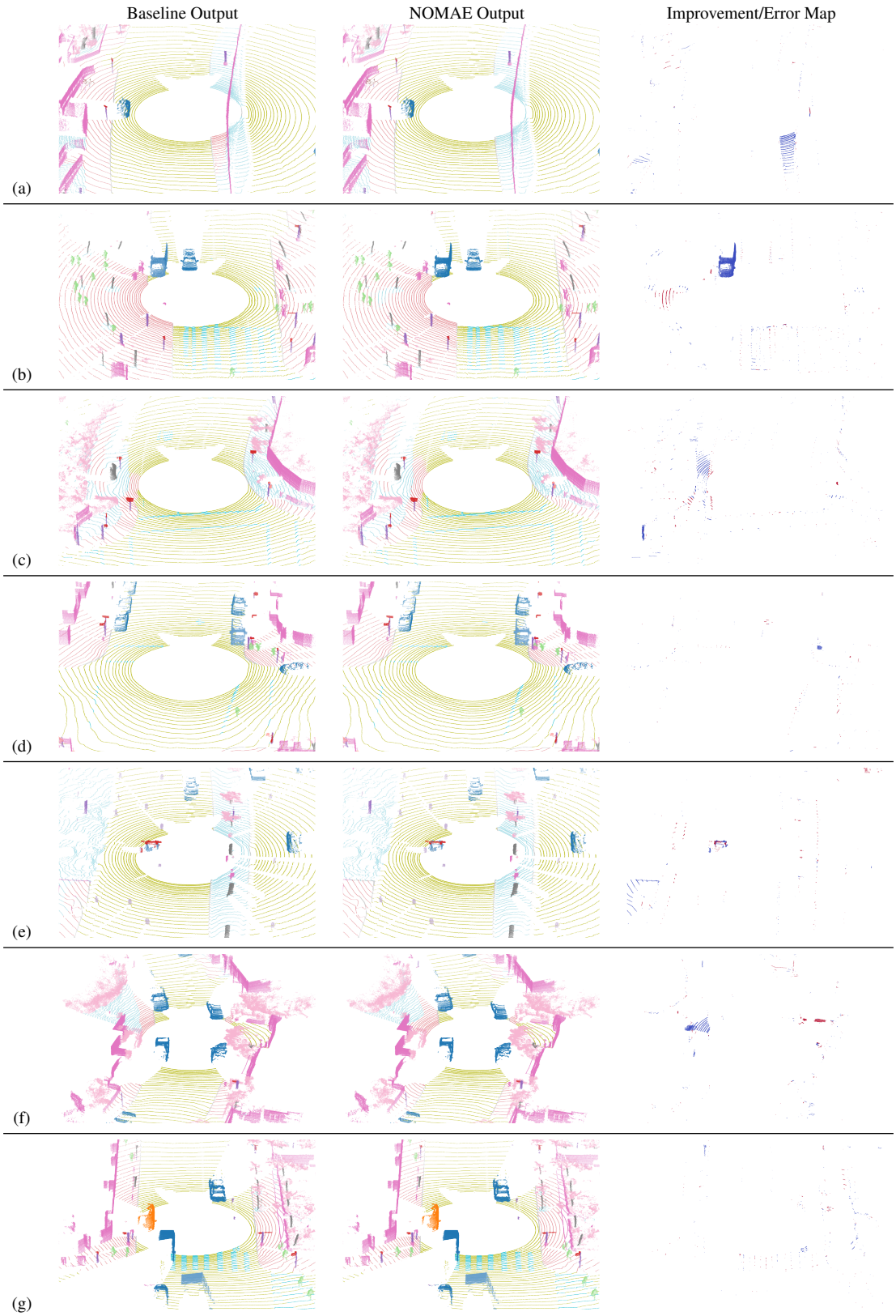

Figure 8. Qualitative comparison of semantic segmentation performance with our baseline Ptv3 [37] on the Waymo validation set.

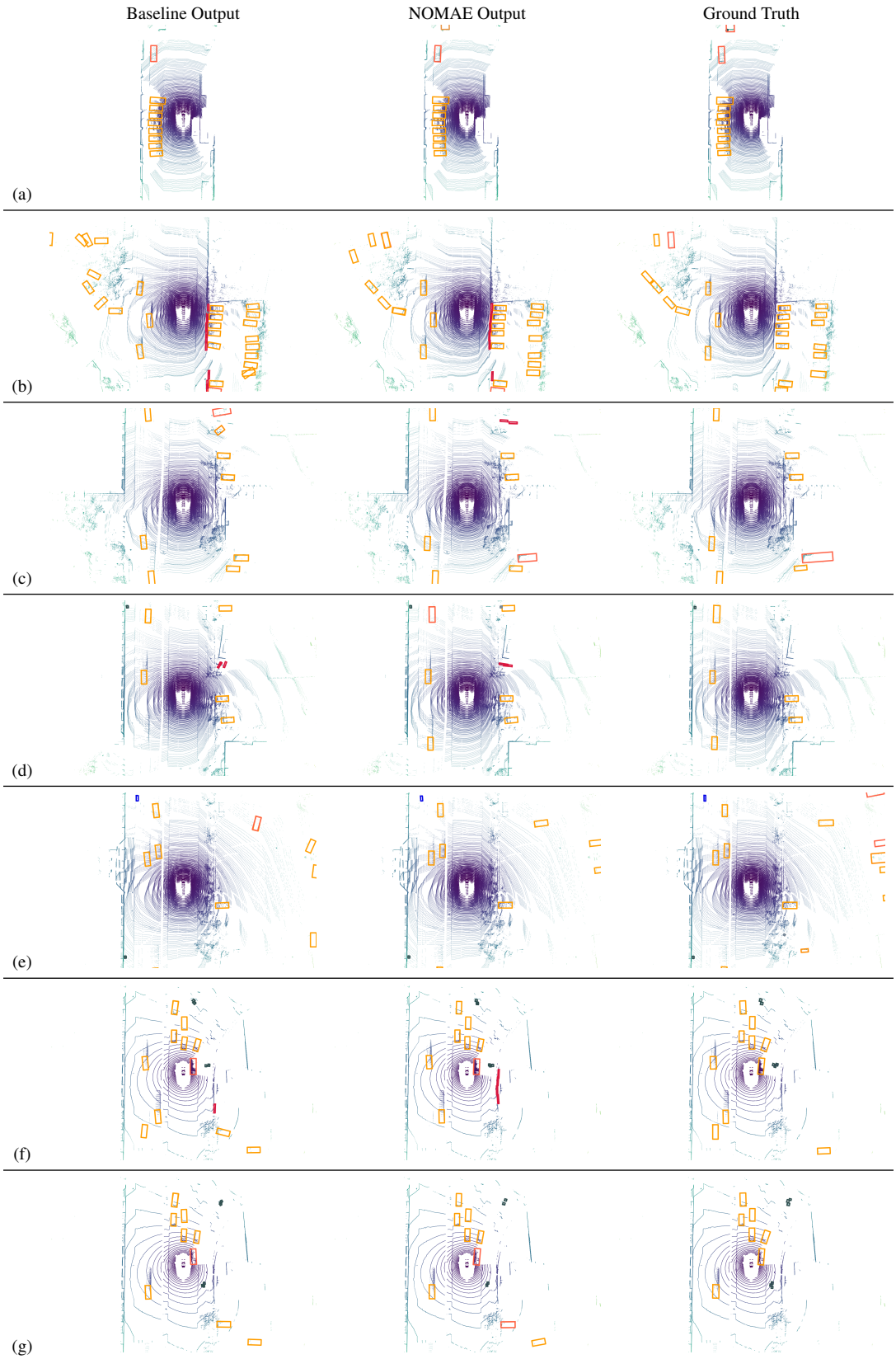

Figure 9. Qualitative comparison of object detection performance with the baseline UVTR [23] on nuScenes val set.

of the labelled data. Additionally we use all modern strategies commonly used for object detection such as CBGS [48], and Copy-Paste augmentation [42]. We compare between NOMAE pretraining followed by 5 epochs fineuning and traning from scratch for 20 epochs (any longer schedule does not provide improvement). We also perform the same comparison using the commonly used CNN based SparseEncoder backbone. The results in Tab.17 shows that NOMAE brings consistent improvement to both NDS and mAP irrespective of the backbone architecture. Using NOMAE pretraining UVTR+PTv3 backbone achieves new SOTA for object detection on nuScenes val set.

NDS/mAP	Scratch 20ep	NOMAE +5ep
UVTR+PTv3	69.3/64.1	**71.2/67.0**
UVTR+SparseEncoder [37]	67.7/60.9	70.4/65.3

Table 17. Results on nuScenes object detection val set for training from scratch compared to NOMAE pretraining.

12. Qualitative Analysis

In this section, we present more qualitative results. Fig. 8 shows results for semantic segmentation and Fig. 9 object detection. For semantic segmentation, we observe the general trend of NOMAE correctly classifying smaller objects that are misclassified by the baseline (examples (c), (e), and (g)). Additionally, the boundaries between different semantic classes are more pronounced in the case of NOMAE (examples (a), (d), and (f)), which shows the importance of self-supervision at finer resolutions. We can also observe that NOMAE helps with the mixing between similar classes, such as trucks and buses in example (b), as well as different ground types in other examples. NOMAE also more accurately estimates the orientation of the objects and has higher true positive detections in the case of object detection.