LinK: Linear Kernel for LiDAR-based 3D Perception Supplementary Material

Tao Lu [1] Xiang Ding [1] Haisong Liu [1] Gangshan Wu [1] Limin Wang [1,2*]
[1]State Key Laboratory for Novel Software Technology, Nanjing University [2]Shanghai AI Lab
{taolu,xding,liuhs}@smail.nju.edu.cn, {gswu,lmwang}@nju.edu.cn

In this document, we provide more implementation details, experimental results and analyses to clarify the properties of LinK.

A. More implementation details and results

A.1. Detection

Training Process Following common practice [2, 7], the reported validation results are obtained through training on the train split, and the results on test set are obtaining through training on the train+val split. The subset for training (train or train+val) are augmented using the CBGS strategy, which balances the sample distribution. Meanwhile, a gt-sampling strategy [5] is adopted to enhance object-level balance during training. Our network are trained with CBGS+gt-sampling for 15 epochs, and then finetuned by removing the gt-sampling for extra 5 epochs. Experiences in previous work indicate that such training policy benefits from the augmented dataset most while avoids overfitting the synthetic distribution.

Results The TTA process for nuScenes contains the flipping and rotation. For flipping, we apply 4 operations: [no flip, x-axis, y-axis, x-axis+y-axis]. For the rotation, we adopt 7 angles, i.e., [$0°$, $\pm 6.25°$, $\pm 12.5°$, $\pm 25°$]. Thus there are total 28 variants for each sample during inference. All the results for the same sample are reduced by a NMS process.

A.2. Segmentation

Data Augmentation The input for semantic segmentation is a 4-dimension tensor, consisting of the normalized coordinate of each point and the corresponding LiDAR reflection intensity. The coordinates are augmented with random flip along x-aixs or y-axis, random scaling within [0.95, 1.05], and random rotation within $[0, 2\pi)$. For the TTA process during inference, we apply the random augmentation for 12 times and average the results.

*Corresponding author

Table 1. Different kernel sizes for segmentation. Without TTA.

$r \times s$	mIoU(%)@SemKITTI val
3×2	66.9
3×3	67.3
3×5	67.5
3×7	67.2

Table 2. Validation on Waymo Detection. Trained for 6 epochs.

Methods	Vehicle	Pedestrian	Cyclist	mAPH(%)
CenterPoint	63.4	59.5	66.4	63.4
+LinK	(+1.6)65.0	(+0.9)60.4	(+2.0)68.4	(+1.2)64.6

B. Detailed layer architecture

Both of the segmentation and detection task share the same encoder design. The encoder starts with a Stem Block (Conv3$\times$3$\times$3+BN+ReLU+Conv3$\times$3$\times$3+BN+ReLU) and then appends with 4 downsample+parallel layers (Residual Branch $\|$ LinK Module). Each Residual Branch consists of two residual blocks. The detailed architecture of each parallel layer is depicted in Fig 1. For segmentation task, the hidden dimensions for all the encoder layers are 64. For the detection, the hidden dimension is $[16, 32, 64, 128]$.

C. Results on more datasets

To further explore the potential of LinK, we conduct experiments on other three benchmarks. For the 3D object detection, we first train the CenterPoint and LinK for 6 epochs on Waymo Detection [4], the results on validation split are shown in Table 2. We also train KITTI Detection [3] using CentePoint-KITTI [6] and LinK under the same settings. The results is provided in Table 3. The detection results on the two datasets further demonstrate the effectiveness of our large kernel design.

For semantic segmentation, we design one more segmentation experiment on nuScenes [1]. According to Table 4, LinK achieves consistent improvement over the baseline.

D. More ablations

Different kernel sizes in semantic segmentation task.

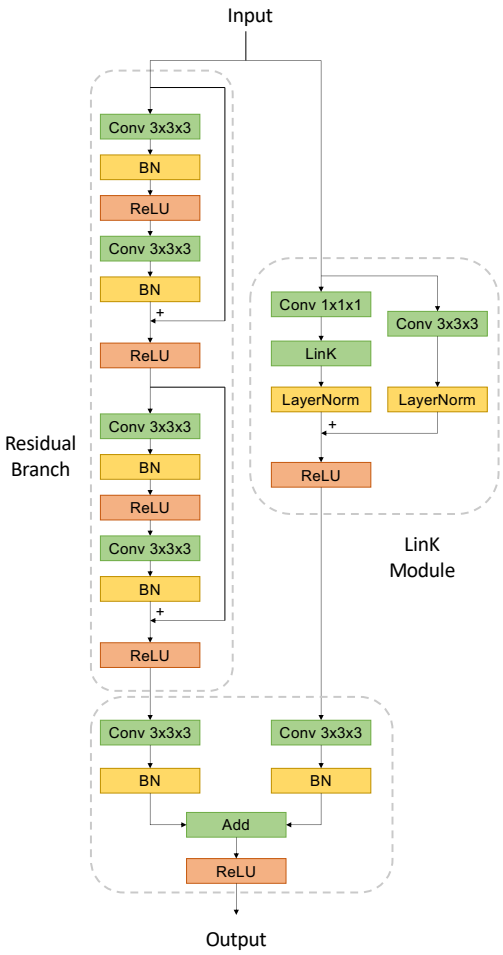


Figure 1. The detailed architecture of the encoder layer.

Table 3. Validation on KITTI Detection. (mAP)

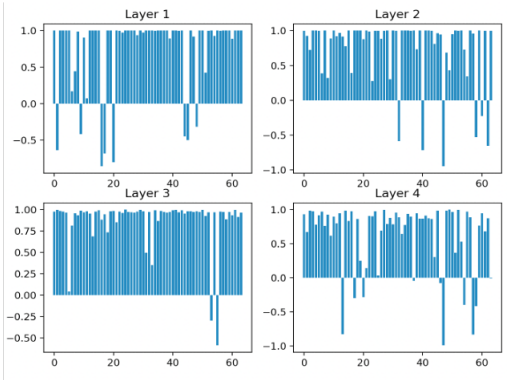
Methods	Easy	Moderate	Hard
CenterPoint	71.2	61.7	58.1
+LinK	(+1.6)72.8	(+2.1)63.8	(+2.1)60.2

Table 4. Validation on nuScenes Segmentation. (#channel=64, #epoch=80, bs=16, voxel size=10cm)

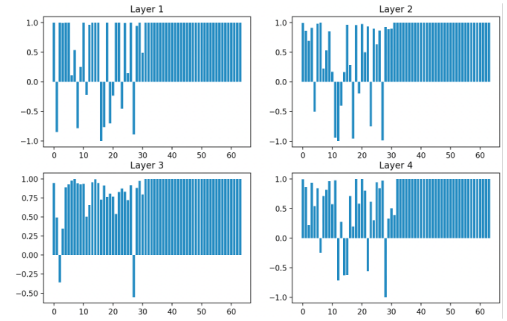
Methods	mIoU(%)	mAcc(%)	oAcc(%)
Mink	74.8	82.3	93.3
+LinK	(+1.6)76.4	(+0.8)83.1	(+0.4)93.7

We report the segmentation results of more kernel sizes in Table 1 and the performance saturates at 3×5.

Mirco-designs in LinK module. The default dilation in the bypass branch is 1, and we enlarge it to 2 in Table 5. The comparison implies that there is no need to enlarge the receptive field of bypass branch. Furthermore, the norm type in LinK does not have an impact on the segmentation results.



(a) w/o Group Sharing



(b) Group Sharing

Figure 2. Activations in different channels. For (b), the first 32 channels are repeated to serve as 64-channel weights. The latter 32 channels for the Group Sharing does not participate in any forward or backward process.

Table 5. Mirco-designs. Validate on val@SemanticKITTI.

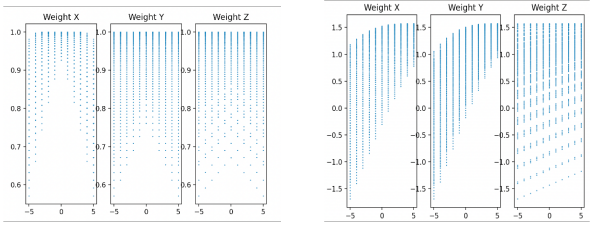
bypass dilation	mIoU	mAcc	norm type	mIoU	mAcc
1	67.5	74.7	LayerNorm	67.5	74.7
2	66.3	72.9	BatchNorm	67.8	74.6

E. Visualizations

E.1. Kernel weight Distributions

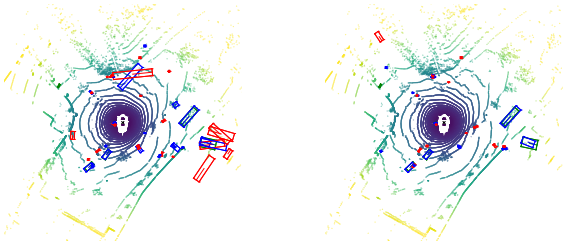
Channel Distribution To explore the effect of the group sharing strategy, we analyze the weight distribution in channel dimension. According to Fig 2, before adopting the Group Sharing strategy, the low-level channels are optimized insufficiently, since the variations among channels are very small. After adopting the Group Sharing policy, the low-level channel are activated significantly, which verifies the effectiveness of Group Sharing.

Spatial Distribution This part introduces the effect of the Learnable Frequency strategy. According to the Fig 3, the original kernel shows poor spatial inductive bias since it cannot distinguish the symmetric locations. And the Learnable Frequency enhances the spatial bias in the generated kernels.



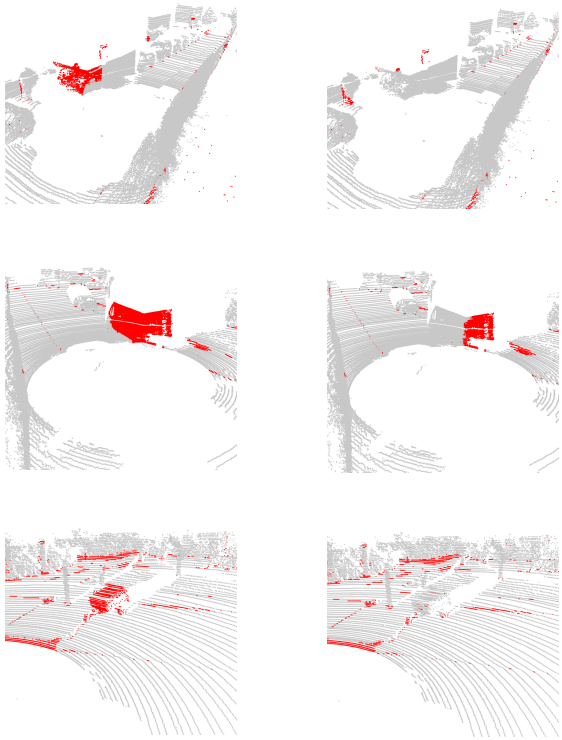
(a) $\Psi(x) = \cos(x)$ (b) $\Psi(x) = \cos(\alpha \cdot x) + x$

Figure 3. Spatial distribution of activations. We illustrate the activation from X-axis, Y-axis, and Z-axis.



(a) Baseline (b) LinK

Figure 4. Detection results on nuScenes. Blue and green box are predictions and ground truth, respectively. LinK improves those remote and sparse objects in scenes. Better viewed in color and using zoom.



(a) Baseline (b) LinK

Figure 5. Error map of the segmentation in SemanticKITTI. Red point denote false prediction.

E.2. More qualitative results

We provide more detection results in Fig 4 and more segmentation results in Fig 5 to show our improvement compared to the baseline. The sparse areas are sensitive to the noise, making the prediction unreliable. LinK improves this issue by introducing wider-range context to enhance the robustness. And the larger receptive field in LinK is more friendly to the large object.

References

[1] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, 2020. 1

[2] Yukang Chen, Jianhui Liu, Xiaojuan Qi, Xiangyu Zhang, Jian Sun, and Jiaya Jia. Scaling up kernels in 3d cnns. *arXiv preprint arXiv:2206.10555*, 2022. 1

[3] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361, 2012. 1

[4] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1

[5] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10):3337, 2018. 1

[6] Tianwei Yin. Centerpoint-kitti. https://github.com/tianweiy/CenterPoint-KITTI, 2022. 1

[7] Benjin Zhu, Zhengkai Jiang, Xiangxin Zhou, Zeming Li, and Gang Yu. Class-balanced grouping and sampling for point cloud 3d object detection. *arXiv preprint arXiv:1908.09492*, 2019. 1