SUPPLEMENTARY MATERIAL OF LIDAR-PTQ

APPENDIX A: LIDAR-PTQ FOR DIFFERENT DETECTORS

CenterPoint (Yin et al., 2021) integrates two milestone works in LiDAR-based BEV detection, VoxelNet (Zhou & Tuzel, 2018) and PointPillars (Lang et al., 2019) as CP-Pillar and CP-Voxel. In particular, CP-Pillar and CP-Voxel have different network design.

The CP-Pillar model is a fully dense convolutional network, while the CP-Voxel model includes SP-Conv and dense convolution. Our results on CenterPoint (-pillar and -voxel) demonstrate that: i) Lidar-PTQ is applicable to pillar-based and voxel-

| Method | representation | backbone | neck | head |
|-----------|----------------|----------|--------|--------|
| CP-Pillar | Pillar | dense | dense | dense |
| CP-Voxel | Voxel | sparse | dense | dense |
| FSD | Point+Voxel | sparse | sparse | sparse |

based detectors. ii) Lidar-PTQ is applicable to SPConv and dense convolution operations.

Table 1: Performance comparison on nuScene *val* set. We show the NDS, and mAP for each class. Abbreviations: construction vehicle (CV), pedestrian (Ped), motorcycle (Motor), bicycle (BC) and traffic cone (TC).

| Models | Methods | Bits(W/A) | NDS | mAP Car | Truck | Bus | Trailer | CV | Ped | Motor | BC | TC | Barrier |
|---------------|----------------------------|-------------------|----------------------|--------------------------|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|----------------------|
| | Full Prec. | 32/32 | 60.3 | 50.0 83.8 | 50.6 | 61.8 | 31.2 | 9.2 | 79.4 | 44.1 | 20.2 | 57.7 | 61.3 |
| CP- Pillar | Brecq QDrop PD-Quant | 8/8 8/8 8/8 | 56.9 57.8 59.6 | 43.675.945.978.848.381.8 | 41.4 44.2 47.6 | 54.3 57.0 59.4 | 21.6 23.8 28.2 | 3.8 5.2 7.8 | 78.1 78.4 78.4 | 37.4 40.1 41.6 | 15.7 17.6 19.8 | 55.0 56.7 57.6 | 53.3 56.8 61.0 |
| | LiDAR-PTQ | 8/8 | 60.2 | 49.8 83.7 | 50.8 | 61.8 | 30.6 | 9.0 | 79.0 | 43.6 | 20.4 | 57.8 | 61.0 |
| | Full Prec. | 32/32 | 64.8 | 56.6 84.6 | 54.5 | 66.7 | 36.4 | 16.9 | 83.1 | 56.1 | 39.6 | 64.0 | 64.3 |
| CP- Voxel | Brecq QDrop PD-Quant | 8/8 8/8 8/8 | 62.0 63.2 63.7 | 51.276.554.082.155.283.7 | 46.8 48.5 51.1 | 60.5 64.9 66.6 | 28.9 32.9 34.1 | 12.5 15.1 16.6 | 80.4 81.1 82.8 | 53.7 55.1 55.1 | 34.8 36.9 36.4 | 58.8 60.9 62.6 | 59.1 63.7 63.2 |
| | LiDAR-PTQ | 8/8 | 64.7 | 56.5 84.6 | 54.2 | 66.7 | 36.4 | 16.6 | 83.3 | 56.0 | 39.4 | 63.6 | 64.4 |

APPENDIX B: PERFORMANCE COMPARISON ON NUSCENES DATASET

To further evaluate the effectiveness of LiDAR-PTQ, we also conducted experiments on nuScenes (Caesar et al., 2020) dataset. Our performance evaluation involves two metrics, average precision (mAP) and nuScenes detection score (NDS). NDS is a weighted average of mAP and other attributes metrics, including translation, scale, orientation, velocity, and other box attributes. As shown in Tab 1, LiDAR-PTQ achieves state-of-the-art performance and outperforms BRECQ and QDrop by a large margin of 6.2 mAP and 3.9 mAP on CenterPoint-Pillar model and 5.3 mAP and 2.5 mAP on CenterPoint-Voxel model. Consistent with the accuracy on the Waymo dataset, our LiDAR-PTQ also achieves almost the same performance as the full precision model on nuScenes dataset.

APPENDIX C: LIDAR-PTQ FOR POINT CLOUD SEGMENTATION

Additionally, we conducted experiments on SemanticKITTI (Behley et al., 2019) dataset for point cloud segmentation to further evaluate the generalization of LiDAR-PTQ. Specifically, we utilize SPVNAS (Tang et al., 2020) as our baseline, which is a representative work in point cloud segmentation task. As shown in Tab 2, adopting entropy calibration leads to a significant accuracy drop of **18.09 mIOU**. As for a vanilla max-min calibration, there is still a performance drop **2.64 mIOU** for quantized SPVNAS. However, LiDAR-PTQ can further achieve comparable accuracy to its float counterpart. This demonstrates the effectiveness of LiDAR-PTQ on point cloud segmentation tasks as well.

| Method | mIoU | car | bicycle | motorcycle | truck | other-vehicle | person | bicyclist | motorcyclist | road | parking | sidewalk | other-ground | building | fence | vegetation | trunk | terrain | pole | traffic sign |
|------------|------|------|---------|------------|-------|---------------|--------|-----------|--------------|------|---------|----------|--------------|----------|-------|------------|-------|---------|------|--------------|
| Full Prec. | 65.0 | 96.3 | 49.0 | 77.6 | 74.4 | 51.8 | 75.2 | 88.2 | 5.7 | 93.4 | 44.6 | 81.0 | 3.5 | 89.5 | 56.5 | 87.8 | 68.4 | 75.1 | 67.1 | 49.6 |
| Entropy | 46.9 | 92.9 | 34.7 | 72.1 | 20.4 | 37.2 | 48.5 | 80.9 | 5.1 | 47.8 | 16.9 | 28.7 | 0.2 | 79.9 | 47.5 | 82.9 | 57.0 | 44.0 | 55.9 | 38.8 |
| Max-min | 62.4 | 94.5 | 46.2 | 75.3 | 73.0 | 50.2 | 73.6 | 86.4 | 5.7 | 92.3 | 41.5 | 78.9 | 2.1 | 87.3 | 53.4 | 85.1 | 65.3 | 71.8 | 63.0 | 48.6 |
| LiDAR-PTO | 64.9 | 96.3 | 48.7 | 78.0 | 74.3 | 52.2 | 74.5 | 87.9 | 5.9 | 93.3 | 44.0 | 80.9 | 3.5 | 89.4 | 56.4 | 87.6 | 68.3 | 74.5 | 67.2 | 49.5 |

Table 2: The PTQ performance of SPVNAS on SemanticKITTI val set.

APPENDIX D: EXPERIEMNTS DETAILS

Dataset. NuScenes dataset (Caesar et al., 2020) uses a LiDAR with 32 lines to collect data, containing 1000 scenes with 700, 150, and 150 scenes for training, validation, and testing, respectively. The metrics of the 3D detection task are mean Average Precision (mAP) and the nuScenes detection score (NDS). Waymo Open Dataset (Sun et al., 2020) uses a LiDAR with 64 beams to collect data, containing 1150 sequences in total, 798 for training, 202 for validation, and 150 for testing. The metrics of the 3D detection task are mAP and mAPH (mAP weighted by heading). In Waymo, LEVEL1 and LEVEL2 are two difficulty levels corresponding to boxes with more than five LiDAR points and boxes with at least one LiDAR point. The detection range in nuScenes and WOD is 50 meters (cover area of 100m × 100m) and 75 meters (cover area of 150m × 150m).

Implementation Details. All the FP models in our paper use CenterPoint(Yin et al., 2021) official open-source codes based on Det3D (Zhu et al., 2019) framework. In WOD dataset, we randomly sample 256 frames point cloud data from the training set as the calibration data. The calibration set proportions is **0.16%** (256/158,081) for WOD. In nuScenes dataset, the calibration set proportions are **0.91%** (256/28,130). We set the first and the last layer of the network to keep full precision. We execute block reconstruction for the backbone and layer reconstruction for the neck and the head with a batch size of 4, respectively. Note that we do not consider using Int8 quantization for the PFN in CenterPoint-Pillar, since the input is 3D coordinates, with approximate range $\pm 10^2$ m and accuracy 0.01 m, so that Int8 quantization in FPN would result in a significant loss of information. The learning rate for the activation quantization scaling factor is 5e-5, and for weight quantization rounding, the learning rate is 5e-3. In TGPL loss, we set γ as 0.1, and K as 500. We execute all experiments on a single Nvidia Tesla V100 GPU. For the speed test, the inference time of all comparison methods is measured on an NVIDIA Jeston AGX Orin, a resource-constrained edge GPU platform widely used in real-world autonomous driving.

APPENDIX E: ENTROPY CALIBRATION METHOD

Given the original and quantized data distribution p(i) and q(i) as follows:

$$D_{KL}(p(i), q(i)) = \sum_{i} p(i) \log p(i) - p(i) \log q(i)$$
(1)

The entropy calibration method in Algorithm2

APPENDIX F: GIRD SEARCH

For a weight or activation tensor X, we can get their initial quantization scale factor using the following equation:

$$\hat{x} = (clamp(\lfloor \frac{x}{s} \rceil + z, q_{min}, q_{max}) - z) \cdot s$$
⁽²⁾

$$s = (x_{max} - x_{min}) / (2^{b} - 1)$$
(3)

Algorithm 1 Entropy calibration method **Input**: FP32 histogram H with N bins, and bit-width b. **Output**: threshold with $min(D_{KL}(p(i), q(i)))$. **Require:** len(p) = len(q)1: for *i* in range $(2^{b-1}, N)$ do $ref_dist_p(i) = [bin[0], ..., bin[i - 1]]$ 2: outliers_count = $sum(bin[i], bin[i+1], \dots, bin[N-1])$ 3: 4: $ref_dist_p(i)[i-1] + = outliers_count$ $p(i) = \text{ref_dist_p}(i) / sum(\text{ref_dist_p}(i))$ 5: quantize candidate_dist_q(i) from [bin[0], ..., bin[i-1]] into 2^{b-1} levels 6: candidate_dist_q(i)=interp1d((bin[0],...,bin[127]), (bin[0],...,bin[i-1]),method='linear') 7: 8: $q(i) = \text{candidate_dist_q}(i)/sum(\text{candidate_dist_q}(i))$ 9: divergence $[i] = D_{KL}(p(i), q(i))$ using Eq 1 10: end for 11: $m = \operatorname{argmin} \left(D = \left[\operatorname{divergence} \left[2^{b-1} - 1 \right], ..., \operatorname{divergence} \left[N - 1 \right] \right] \right)$ 12: threshold = $(m + 0.5) * (width_{bin})$ 13: return threshold

$$\arg\min \|(X - \hat{X}(s_l))\|_F^2 \tag{4}$$

 $\|\cdot\|_{F}^{2}$ is the Frobenius norm (MSE Loss). Refer to appendix for more details about grid search. Then linearly divide the interval $[\alpha s_{0}, \beta s_{0}]$ into T candidate bins, denoted as $\{s_{t}\}_{t=1}^{T}$. α, β and T are designed to control the search range and granularity. Finally, search $\{s_{t}\}_{t=1}^{T}$ to find the optimal s_{opt} that minimizes the quantization error, The entropy calibration method in Algorithm2

 s_t

Algorithm 2 Grid search

Input: the input of full precision tensor X, bit-width b and T bins.

Output: scale factor s_{opt} with $min(||(X - \hat{X}(s_l))||_F^2)$.

- 1: using $x_{max} = max(|x|)$ get max value of tensor X
- 2: set $range = x_{max}, c_{best} = 100$
- 3: set $v_{min} = x_{min}$ and $v_{max} = x_{max}$
- 4: for i in range(1, T) do
- 5: threshold = range/T/i
- 6: $x_{min} = -threshold, x_{max} = threshold$
- 7: get scale s_t with x_{min} and x_{max} using Eq 3
- 8: input the quantized value \hat{x} and FP value x using Eq 2 to get score c
- 9: update v_{min} and v_{max} when $c < c_{best}$ and update $c_{best} = c$
- 10: end for
- 11: get v_{min} and v_{max} with the minimal score c
- 12: get final scale s_{opt} with v_{min} and v_{max} using Eq 3
- 13: return scale s_{opt}

REFERENCES

- Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9297–9307, 2019. 1
- Holger Caesar, Varun Bankiti, Alex Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. Nuscenes: A multimodal dataset for autonomous driving. pp. 11621–11631, 2020. 1, 2

- Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In CVPR, pp. 12697–12705, 2019. 1
- 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. pp. 2446–2454, 2020. 2
- Haotian Tang, Zhijian Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching efficient 3d architectures with sparse point-voxel convolution. In *European conference on computer vision*, pp. 685–702. Springer, 2020. 1
- Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. pp. 11784–11793, 2021. 1, 2
- Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. pp. 4490–4499, 2018. 1
- 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. 2