Q-LiDAR: Efficient and Accurate Training-Free Quantization for Point Cloud 3D Object Detection Models

Anonymous CVPR submission

Paper ID *****

Abstract

3D object detection with point clouds is crucial for applications in autonomous driving, robotics, and augmented reality. As these applications advance towards real-time processing on edge devices, they demand models that enable efficient and flexible inference. While recent post-training quantization methods address some of these challenges by reducing model size and computational load without retraining, they are limited by calibration data biases and suffer from sub-optimal accuracy without precise calibration. Moreover, existing methods often overlook the distinct characteristics of various components in 3D LiDAR models, where high variance in value distributions poses additional quantization challenge. To overcome these issues, we propose Q-LiDAR, a novel quantization approach that incorporates techniques, including SmoothQConv, fine-grained quantization for sparse operators, and Hessian-guided bit-width allocation. Our approach achieves W4A8 mixed-precision quantization on state-of-the-art 3D LiDAR models while retaining XX% model accuracy, without requiring a calibration dataset or retraining.

1. Introduction

3D object detection with point clouds is a critical task in various applications such as autonomous driving, robotics, and augmented reality. These applications rely on accurate and efficient detection of objects in 3D space to navigate and interact with their environment safely and effectively. As they move toward real-time processing on edge devices, the demand for efficient models has grown even higher.

Model quantization has proven to be an effective compression method. By compressing high bit-width floating-point (FP) data into lower bit-width integers, the computational and memory costs of the model can be significantly reduced. Prior works have studied quantization methods for 2D object detection models and achieved promising results [20, 22, 34]. However, directly applying these quantization methods to 3D point cloud object detection models leads to sub-optimal accuracy [10]. Moreover, prior work often relies on quantization-aware training (QAT) [10, 37], which requires extensive fine-tuning, limiting its flexibility for rapid deployment in resource-constrained environments.

Recently, LiDAR-PTQ introduces a post-training quantization (PTQ) approach that reduces model size and computational demands for 3D object detection models without retraining [38]. While achieving promising results, LiDAR-PTQ faces two main challenges:

- **Calibration data bias**: Despite eliminating the need for retraining, LiDAR-PTQ relies on calibration data during the quantization process. The quality of the quantization can be negatively affected depending on the calibration data provided. For example, if the calibration data is not representative of the full dataset, the quantization might not generalize well.
- **Poor accuracy without calibration**: Empirical results show that quantizing LiDAR models for 3D object detection with point cloud is challenging due to the complex mix of diverse layers specifically designed for point cloud processing. We observe high variance across four different components in LiDAR models: (1) 2D/1D convolution (Conv2D/1D), (2) sparse convolution (SPConv3D), (3) submanifold convolution (SubMConv3D), and (4) multi-layer perceptron (MLP). The unique distributions of activations and weights across these components result in significant variations, making it very challenging to apply a uniform set of quantization parameters, such as, W8A8, to values with high variance.

Based on our observations of the data distribution, we draw ideas from the large language model (LLM) compression literature, where training-free quantization methods have been shown to be effective [8, 27, 33]. To avoid quantization errors from outliers in convolution operations, we extend a smoothing-based quantization technique [33] to transform the quantization difficulties in activations to convolution weights or vise versa. To compress SPConv3D and SubMConv3D layers, we adopt channel-wise quantization. However, when assigning proper bit-width for each component, we find that simple Mean Square Error with the uncompressed model is inadequate to distinguish quantization effects on various components. To overcome this, we propose to leverage Hessian information to estimate the quantization sensitivity of each component and guide mixed-precision bit-width allocation across components.

The contributions of the paper are four-fold:

1. **Comprehensive Component Analysis**: We conduct an in-depth analysis of the data distribution across different components in 3D LiDAR object detection models. This investigation reveals the unique quantization challenges associated with the diverse layers in these models.
2. **Development of Calibration-Free Quantization Method**: We propose a calibration-free quantization method, Q-LiDAR, that employs component-specific quantization strategies, including SmoothQConv for Conv2D/1D and MLP layers, and channel-wise quantization for sparse operators.
3. **Sensitivity-Based Mixed-Precision Quantization**: To address the bit-width allocation challenges in various components, such as SPConv3D, SubMConv3D, Conv2/1d, and MLP layers, Q-LiDAR incorporates a sensitivity-based bit-width allocation policy based on Hessian information, tailored to each component's unique characteristics to mitigate the accuracy loss.
4. **Extensive Experimental Validation**: We validate the effectiveness of Q-LiDAR across a range of state-of-the-art LiDAR models for 3D object detection. Experimental results demonstrate that Q-LiDAR achieves XX compression ratio while obtaining very comparable accuracy (XX%) as the uncompressed model including [Hongbo: will add the finalized models over here]. **[Explicit accuracy here]**. We empirically select recent advances that have been adopted in many industries to construct a unified baseline. The follow-up experiments show that we achieved **[n%]**, **[n%]** and **[n%]** respectively on KITTI, nuScences and Waymo datasets as compared to direct quantization and achieve mAP and NDS of **[n%]**, **[n%]** and **[n%]**. [Hongbo: evaluation results here]

2. Related Work

3D object detection. 3D object detection (3DOD) is a pivotal area of research for autonomous driving, robotics, and augmented reality. This process heavily relies on sophisticated sensor technologies such as LiDAR (Light Detection and Ranging), radar, and stereo vision cameras that capture detailed three-dimensional information about the environment. Among them, LiDAR has become one of the most widely used sensors for its real-time feedback and high accuracy, and since the data collected are separated points with different properties, they are also called point cloud.

Several notable methods are introduced to capture precise 3D spatial information, including PointNet [24] and its variants PointNet++ [25] and PointNeXt [26], which directly process point clouds, and voxel-based methods like VoxelNet [39], Voxel Transformer [21], and VoxelNeXt [4], which convert point clouds into structured grids for easier processing.

To efficiently manage sparse point cloud data, which is inherently memory-intensive, prior work introduce custom layers, such as SparseConv and SubMConv layers to handle sparse point cloud data [5]. These layers leverages the inherit sparsity of the input data by performing convolutions exclusively on non-zero elements, which drastically reduces both memory consumption and computational overhead, making it particularly suitable for large-scale 3D data processing.

Training-free quantization. While early model compression techniques focus on improving model accuracy through retraining or fine-tuning [6, 7, 15, 19, 23], they face challenges in flexibility, which hinders the widespread adoption of those methods across diverse deployment environments. Recent advancements in training-free compression have significantly improved the efficiency and deployment of vision transformers [13, 14, 17, 20, 36]. In NLP, techniques such as GPTQ [9], AWQ [16], SmoothQuant [33] have also demonstrated their success in quantizing large language models. These advancement highlight the ongoing efforts to compress DNN models. While demonstrating promising results, few studies have looked into training-free compression for 3D LiDAR object detection models.

3. Methodology

In this section, we first introduce Q-LiDAR, a novel training-free quantization method to compress 3D object detectors while retaining accuracy. And then we develop a Hessian-guided method for bit-width allocation to reduce quantization errors.

Giving the hybrid architecture of 3D LiDAR models, we investigate the impact of quantization on specific components within 3D object detection models, especially six components commonly used in 3DOD models: SparseConv3d, SubMConv3d, SparseConv2d, SubMConv2d, Conv2d/1d, MLP.

3.1. Improving 3D LiDAR Model Compression via SmoothQConv

We start by directly applying W8A8 post-training quantization to 3D LiDAR models. However, we find that W8A8 leads to large accuracy drop. Table 1 shows the results of various combination of quantization bit-width with both static and dynamic round-to-nearest (RTN) post-training quantization over CenterPoint-Voxel [35] and the autonomous driving dataset NuScenes *val*. The quantization operation is formulated as:

$$\boldsymbol{X}^{\mathrm{INT}} = \mathrm{clamp}\left(\left\lfloor \frac{x}{s} \right\rceil + z, q_{\min}, q_{\max}\right) \tag{1}$$

where $\lfloor \cdot \rceil$ is the rounding-to-nearest operator, s is the scaling factor, and z is the zero-point. As shown, while W8A8 quantization has been considered quite robust to leads to traditional 2D convolution tasks [], it causes around 20 mAP and NDS loss, which is quite significant. In contrast, W4A16 quantization leads to relatively lower accuracy loss.

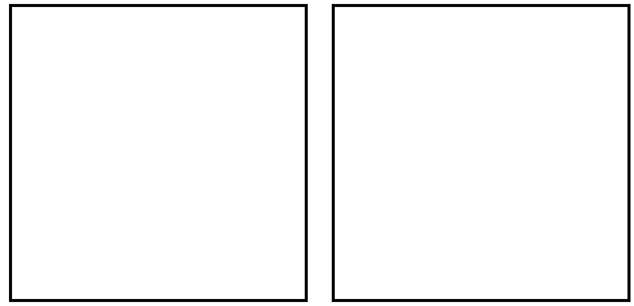
Table 1. Quantization Results with Performance Gaps

Method	Bits(W/A)	Metrics mAP	NDS
Full Prec.	32/32	59.22	66.48
Dynamic	8/8	39.56 **(-19.66)**	47.63 **(-18.85)**
Static	8/8	38.46 **(-20.76)**	46.13 **(-20.35)**
Dynamic	4/8	34.36 **(-24.86)**	44.87 **(-21.61)**
Static	4/8	33.80 **(-25.42)**	44.16 **(-22.32)**
Dynamic	4/16	51.24 **(-7.98)**	59.38 **(-7.10)**
Static	4/16	51.24 **(-7.98)**	59.38 **(-7.10)**
Dynamic	16/4	xx.xx **(-7.98)**	xx.xx **(-7.10)**
Static	16/4	xx.xx **(-7.98)**	xx.xx **(-7.10)**

To investigate why activation quantization leads to more significant accuracy drop, we further collect 1) the top-8 maximum weight values; and 2) the average of the top-8 activation values. As illustrated in Figure 1 and 2, the results show that the activations, especially those convolution layers, contain outliers, whereas the weight values contain a much smaller dynamic range. Notably, the majority of outliers are found in the activation values with the maximum value being up to 120, and the scaling of the activation is way greater than that of the weight(magnitude of 30 compared to 2). This confirms the imbalanced scaling and magnitude of the activation and weight value within the model.

[Minjia: TODO: Add the figure that show the dynamic range results of weights and activations across layers here. I remember asking Banghao to collect these results before, so we should have them. Also, it would be better to show the results of conv2D, e.g., the one that correspond to the 120.]

To mitigate the errors introduced by extreme outliers and the imbalanced quantization difficulty between activation and weight, a technique called SmoothQuant [33] can be implemented. SmoothQuant redistributes quantization complexity from one tensor to another (e.g., from activation to weight). This approach is especially effective to reduce outlier impact, particularly in the context of linear operators in large language models (LLMs), where outliers are often found in per-token areas. However, applying SmoothQuant to 3D LiDAR models presents a unique challenge, as there is no direct mathematical mechanism for shifting quantization complexity between activations and convolutional weights.



(a) Activation distribution. (b) Weight distribution.

Figure 1. The left figure shows the dynamic range of activations across different convolution layers. The right figure shows the dynamic range of weights across layers.

To overcome this, we introduce SmoothQConv, an extension of SmoothQuant tailored to convolutional operators. Our primary insight is that convolution can be reformulated as a matrix multiplication by transforming the input data through the *im2col* (image-to-column) operation [3]. The *im2col* operation rearranges the input activation (feature map) into a matrix where each column represents a local region (receptive field) of the input that the convolutional filter will slide over. This process effectively "unfolds" the input data into a 2D matrix. allowing the convolution to be treated as standard matrix multiplication. The resulting matrix from the multiplication is then reshaped back ("fold") into the original spatial dimensions of the output feature map.

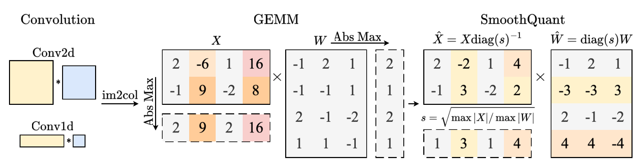


Figure 2. Overview of SmoothQConv operation when $\alpha = 0.5$. "*" and $\times$ indicate convolution and matrix multiplication operations respectively.

The original General Matrix Matrix Multiplication (GEMM) floating-point operation of Conv2d after unfolding is:

$$\boldsymbol{Y} = \boldsymbol{X}^{\mathrm{FP32}} \boldsymbol{W}^{\mathrm{FP32}} \tag{2}$$

where $\boldsymbol{X} \in \mathbb{R}^{bn \times ihw}$ and $\boldsymbol{W} \in \mathbb{R}^{ihw \times c}$.

To leverage INT8 GEMM acceleration on general hardware, we implement weight-per-channel and activation-per-tensor quantization. The output $\boldsymbol{Y}$ is approximated using

Table 2. Notations

in channels	i
out channels	c
kernel height	h
kernel width	w
kernel depth	d
batch size	b
number of sliding	n
number of active voxels	v

quantized INT8 operands as:

$$\boldsymbol{Y} \approx (\hat{\boldsymbol{X}}^{\text{INT8}} \odot \Delta_{\boldsymbol{X}}^{\text{FP32}})(\hat{\boldsymbol{W}}^{\text{INT8}} \text{diag}(\boldsymbol{\Delta}_{\boldsymbol{W}}^{\text{FP32}})) \tag{3}$$

$$\approx \text{diag}(\Delta_{\boldsymbol{X}}^{\text{FP32}})(\hat{\boldsymbol{X}}^{\text{INT8}} \hat{\boldsymbol{W}}^{\text{INT8}})\text{diag}(\boldsymbol{\Delta}_{\boldsymbol{W}}^{\text{FP32}}) \tag{4}$$

where $\hat{\boldsymbol{X}}^{\text{INT8}}$ and $\hat{\boldsymbol{W}}^{\text{INT8}}$ are the quantized activation and weight matrices, and $\Delta_{\boldsymbol{X}} \in \mathbb{R}$ and $\boldsymbol{\Delta}_{\boldsymbol{W}} \in \mathbb{R}^c$ denote the scaling factors for activation-per-tensor and weight-per-channel quantization respectively.

The quantization of activations and weights is defined as:

$$\hat{\boldsymbol{X}}^{\text{INT8}} = \left\lfloor \frac{\boldsymbol{X}^{\text{FP32}}}{\Delta_{\boldsymbol{X}}^{\text{FP32}}} \right\rceil \qquad \hat{\boldsymbol{W}}^{\text{INT8}} = \left\lfloor \frac{\boldsymbol{W}^{\text{FP32}}}{\boldsymbol{\Delta}_{\boldsymbol{W}}^{\text{FP32}}} \right\rceil \tag{5}$$

where $\lfloor \cdot \rceil$ denotes rounding to the nearest integer.

The scaling factors are computed to map the floating-point values to the INT8 quantization range:

$$\Delta_{\boldsymbol{X}} = \frac{\max(\boldsymbol{X}^{\text{fp32}}) - \min(\boldsymbol{X}^{\text{fp32}})}{2^b - 1} \tag{6}$$

and

$$\boldsymbol{\Delta}_{\boldsymbol{W},j} = \frac{\max\limits_{i=1,\dots,ihw}(\boldsymbol{W}_{ij}^{\text{fp32}}) - \min\limits_{i=1,\dots,ihw}(\boldsymbol{W}_{ij}^{\text{fp32}})}{2^b - 1} \tag{7}$$

To extend the SmoothQuant technique to convolution operators, we introduce a dedicated hyperparameter α, which controls the degree to which quantization difficulty is redistributed between tensors. The scaling value s_k is calculated as:

$$s_k = \max(|\boldsymbol{X}_k^{\text{FP32}}|)^{\alpha} / \max(|\boldsymbol{W}_k^{\text{FP32}}|)^{1-\alpha} \tag{8}$$

Utilizing s_k, we can transfer the quantization difficulty from one to the other by applying this scaling value to our activation and weight prior to the actual quantization stage. This yields the new quantized representations:

$$\hat{\boldsymbol{X}}^{\text{INT8}} = \left\lfloor \frac{\boldsymbol{X}^{\text{FP32}}}{\Delta_{\boldsymbol{X}}^{\text{FP32}}} \text{diag}(s_k)^{-1} \right\rceil \tag{9}$$

$$\hat{\boldsymbol{W}}^{\text{INT8}} = \left\lfloor \frac{\boldsymbol{W}^{\text{FP32}}}{\boldsymbol{\Delta}_{\boldsymbol{W}}^{\text{FP32}}} \text{diag}(s_k) \right\rceil \tag{10}$$

The final matrix multiplication is then approximated using these quantized INT8 operands, $\hat{\boldsymbol{X}}$ from equation 9 and $\hat{\boldsymbol{W}}$ from equation 10, as per the approximation in equation 4.

As seen in Figure 1 and 2, the scaling of activation is way greater than that of weight, meaning s_k is mostly greater than 1. Therefore, after scaling up the weight and scaling down the activation, we manage to reduce the rounding error by reducing the value of $\max(|\boldsymbol{X}|)$ in every tensor. Hence, we successfully reduce the quantization error and achieve more accurate results of the convolution operation even though the parameters of the convolution is INT8 quantized.

3.2. Fine-grained Quantization for Sparse Convolutions

3D LiDAR models incorporate sparse operators, such as submanifold convolution [12] and sparse convolution [18], to reduce the computation load. Specifically, these operators selectively process only the active voxels, bypassing non-active regions. Let $\boldsymbol{x}_u$ represent an input feature vector of an active voxel located at 3-dimensional coordinates $u \in \mathbb{R}^3$. The submanifold operator F_0 by a kernel for $\boldsymbol{X}_u$ is formulated as:

$$F_0(\boldsymbol{W}, \boldsymbol{X}_u) = \sum_{\boldsymbol{i} \in N(u)} \boldsymbol{W}_i \boldsymbol{X}_{u+i} \tag{11}$$

where $N(u)$ denotes the set of offsets in the 3-dimensional cube centered at origin relative to u. Each offset is associated with a specific kernel weight parameterized by $\boldsymbol{W_i}$.

Since the sparse operators perform convolution only in active regions of the feature map, we adpot a channel-wise quantization approach for both weights and activations in SPConv and SubMConv layers. The weights of these sparse convolutions, $\boldsymbol{W} \in \mathbb{R}^{c \times i \times h \times w \times d}$, extend Conv2d weights with an additional depth dimension. To quantize these weights, we first reshape $\boldsymbol{W}$ into a 2D matrix $\boldsymbol{W} \in \mathbb{R}^{c \times ihwd}$ and apply channel-wise quantization along the output channel dimension c.

Activations in sparse convolution layers, represented as $\boldsymbol{X} \in \mathbb{R}^{v \times 3}$, where v is the number of active voxels in the feature map and 3 indicates the coordinates (x, y, z) of each active voxel, are similarly quantized. We apply channel-wise quantization along each spatial axis (x, y, z) of the coordinates to ensure independent quantization for each spatial dimension.

3.3. Searching to Allocate Bit-Width/Hessian-Guided Bit-width Allocation

[Hongbo: start fixing here] To consider layer sensitivity, we choose to automatically search for an optimal bit-width allocation policy that minimizes the output difference (e.g., L1 loss) after the quantization for a certain layer.

[Minjia: TODO: Depending on the Hessian results, we may consider the Hessian-guide bit-width allocation or the original sensitivity analysis based quantization. @Banghao, please share the Hessian results as soon as you get them.]

Algorithm 1 Auto-Sensitive Analysis

Require: Pretrained FP model with N layers; Calibration dataset D^c; Standard quantization module replacement map M^q; **XXX** module replacement map M^{sq}; Calibration number T
Ensure: quantization method assigned to each type of layer using corresponding map M_q or M_sq.
Input T samples of D^c to FP network to get averaged FP output of each layer O_{fp};
for $L_i = \{L_i | i = 1, 2, ...N\}$ **do**
    Find quantized layer L_i^q with map M^q;
    Replace the original layer within the model L_i with quantized layer L_i^q;
end for
Input T samples of D^c to standard-quantized network to get averaged standard-quantized output of each layer O^{qint};
for $L_i = \{L_i | i = 1, 2, ...N\}$ **do**
    Find quantized layer L_i^{sq} with map M^{sq};
    Replace the original layer within the model L_i with quantized layer L_i^{sq};
end for
Input T samples of D^c to **XXX** network to get averaged **XXX** output of each layer O^{sqint};
Check standard-quantized network output O^{qint} and FP final output O_{fp} to calculate $L1_{\text{qint}}$;
Check **XXX** network output O^{sqint} and FP final output O_{fp} to calculate $L1_{\text{sqint}}$;
Check $L1_{\text{qint}}$ and $L1_{\text{sqint}}$ to get the list of layers that can be quantized under **XXX**, with others being standard-quantized;

4. Experiments

We conduct experiments to evaluate the effectiveness of Q-LiDAR in terms of accuracy preserving and compression ratio. Our evaluation aims to answer the following questions:

- Can Q-LiDAR enable high compression ratio for 3D LiDAR models without compromising accuracy?
- Does Q-LiDAR effectively generalize across diverse model architectures?
- How does Q-LiDAR compare to existing 3D LiDAR model compression methods in terms of the trade-off between compression ratio and accuracy?

4.1. Evaluation Methodology

Models Our experiments include both transformer-based and convolution-based state-of-the-art 3D LiDAR models. The transformer-based models, DSVT-Voxel [32] and TransFusion-L [1], feature a voxel transformer backbone, a 2D convolution backbone, and a dense convolutional head. In contrast, the convolution-based models, PV-RCNN++[30], PV-RCNN[28], Part-A^2-Anchor [29], and CenterPoint-Voxel [35], are equipped with a sparse 3D convolution backbone, a 2D convolution backbone, and a convolutional dense head.

Datasets We use Waymo Open Dataset (WOD) [31], nuScenes [2], and KITTI [11] for evaluation.

Baselines We evaluate the performance of Q-LiDAR by comparing it against two primary baselines: (1) the full-precision model (FP32) to establish an upper-bound reference, and (2) the standard Max-min quantized model (W8A8), commonly used in edge deployments. Since LiDAR-PTQ does not have their code released, we skip it for quantitative comparison.

4.2. Performance Comparison on Datasets

Waymo Dataset. To evaluate the performance of Q-LiDAR, several experiements are performed with DSVT-Voxel and PV-RCNN++ models on the Waymo dataset.

Table 3. Waymo Results for Different Detectors.

Models	Methods	Bits(W/A)	Vehicle	Pedestrian	Cyclist
	Full Prec.	32/32	x.x	x.x	
	Max-min	8/8	x.x	x.x	
DSVT-Voxel	QL-0.XX	8/8	x.x	x.x	
	Max-min	4/8	x.x	x.x	
	QL-X.XX	4/8	x.x	x.x	
	Full Prec.	32/32	67.68	60.17	72.55
	Max-min	8/8	x.x	x.x	
PV-RCNN++	QL-0.40	8/8	67.01	59.57	72.17
	Max-min	4/8	x.x	x.x	
	QL-0.xx	4/8	x.x	x.x	

nuScenes Dataset. To evaluate the performance of Q-LiDAR, several experiements are performed with BEVFusion and TransFusion-L models on the nuScenes dataset.

KITTI Dataset. To evaluate the performance of our method, we conduct experiments on 2 models, PV-RCNN and Part-A^2, on KITTI dataset.

As shown in Table 5, achieves superior performance compared to max-min quantization method. It manages to

Table 4. nuScenes Results for Different Detectors.

Models	Methods	Bits(W/A)	mAP	NDS
	Full Prec.	32/32	x.x	x.x
	Max-min	8/8	x.x	x.x
TransFusion-L	QL-X.XX	8/8	x.x	x.x
	Max-min	4/8	x.x	x.x
	QL-X.XX	4/8	x.x	x.x
	Full Prec.	32/32	59.22	66.48
	Max-min	8/8	x.x	x.x
CP-Voxel	QL-0.80	8/8	59.16	66.40
	Max-min	4/8	x.x	x.x
	QL-X.XX	4/8	x.x	x.x

Table 5. KITTI Results for Different Detectors

Models	Methods	Bits(W/A)	Car	Pedestrian	Cyclist
	Full Prec.	32/32	83.69	54.84	68.92
	Max-min	8/8	79.28	54.65	69.45
PV-RCNN	QL-0.30	8/8	82.98	54.83	69.65
	Max-min	4/8	78.01	56.54	62.88
	QL-0.40	4/8	78.74	54.74	67.41
	Full Prec.	32/32	79.40	60.11	69.92
	Max-min	8/8	79.40	60.90	70.67
Part-A^2-Anchor	QL-0.40	8/8	79.41	60.28	69.95
	Max-min	4/8	78.17	53.28	67.18
	QL-0.35	4/8	79.25	55.05	68.77

minimize the accuracy loss within less than 1% for both W8A8 and W4A8.

[Minjia: TODO: Add more in-depth description of the results. 1. Describe how to interpret the results in the table. 2. Main observations. 3. Explanation of why we see these results.]

4.3. Ablation Study

We conducted an ablation study to evaluate the effects of the three key components of our framework, using the XXX model on the XXX dataset. As illustrated in Table 6, the application of channel-wise quantization to the 3D backbone network yielded a modest improvement in performance. Building on this, the introduction of SmoothQuant to the model's 2D backbone resulted in a substantial performance leap from xx.x to xx.x. Finally, by employing the auto-sensitive-analysis algorithm (Algorithm 1) to identify and exclude layers particularly susceptible to quantization, we achieved a peak accuracy of xx.x.

[Minjia: TODO: Suggested ablation studies: 1. Q-LiDAR, 2. Q-LiDAR- layer sensitivity, 3. Q-LiDAR- layer sensitivity - channelwise quantization for sparse ops, 4. Q-LiDAR- layer sensitivity - channelwise quantization for sparse ops - SmoothQConv.]

5. Conclusion

In this work, we introduce a training-free approach for efficient and accurate 3D object detection. Our approach employs tailored optimizations against different components in 3D LiDAR models, including SmoothQConv, subchannel-wise grouped quantization for SPConv and SubMConv. Additionally, we introduce a Hessian-guided method for bit-width allocation. Together, Q-LiDAR achieves state-of-the-art compression ratio for LiDAR models over 3D object detection.

Acknowledgments

Use unnumbered third level headings for the acknowledgments. All acknowledgments, including those to funding agencies, go at the end of the paper.

References

[1] Xuyang Bai, Zeyu Hu, Xinge Zhu, Qingqiu Huang, Yilun Chen, Hongbo Fu, and Chiew-Lan Tai. Transfusion: Robust lidar-camera fusion for 3d object detection with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1090–1099, 2022. 5

[2] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020. 5

[3] Kumar Chellapilla, Sidd Puri, and Patrice Simard. High performance convolutional neural networks for document processing. In *Tenth international workshop on frontiers in handwriting recognition*. Suvisoft, 2006. 3

[4] Yukang Chen, Jianhui Liu, Xiangyu Zhang, Xiaojuan Qi, and Jiaya Jia. Voxelnext: Fully sparse voxelnet for 3d object detection and tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21674–21683, 2023. 2

[5] Spconv Contributors. Spconv: Spatially sparse convolution library. `https://github.com/traveller59/spconv`, 2022. 2

[6] Peiyan Dong, LEI LU, Chao Wu, Cheng Lyu, Geng Yuan, Hao Tang, and Yanzhi Wang. Packqvit: Faster sub-8-bit vision transformers via full and packed quantization on the mobile. In *Advances in Neural Information Processing Systems*, pages 9015–9028. Curran Associates, Inc., 2023. 2

[7] Zhen Dong, Zhewei Yao, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. Hawq: Hessian aware quantization of neural networks with mixed-precision. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019. 2

[8] Zhen Dong, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. Hawq: Hessian aware quantization

Table 6. Ablation study of different components of XXX on XXX dataset.

Models	Methods	Bits(W/A)	mAP	NDS
	Full Prec.	32/32	x.x	x.x
DSVT-Voxel	+ASA	8/8	x.x	x.x
	+CWQ	8/8	x.x	x.x
	+SQConv	8/8	x.x	x.x

of neural networks with mixed-precision. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 293–302, 2019. 1

[9] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022. 2

[10] Huan-ang Gao, Beiwen Tian, Pengfei Li, Hao Zhao, and Guyue Zhou. Dqs3d: Densely-matched quantization-aware semi-supervised 3d detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 21905–21915, 2023. 1

[11] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013. 5

[12] Benjamin Graham and Laurens Van der Maaten. Submanifold sparse convolutional networks. *arXiv preprint arXiv:1706.01307*, 2017. 4

[13] Jung Hwan Heo, Arash Fayyazi, Mahdi Nazemi, and Massoud Pedram. A fast training-free compression framework for vision transformers. *CoRR*, abs/2303.02331, 2023. 2

[14] Woosuk Kwon, Sehoon Kim, Michael W. Mahoney, Joseph Hassoun, Kurt Keutzer, and Amir Gholami. A fast post-training pruning framework for transformers. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022*, 2022. 2

[15] Yanjing Li, Sheng Xu, Baochang Zhang, Xianbin Cao, Peng Gao, and Guodong Guo. Q-vit: Accurate and fully quantized low-bit vision transformer. In *Advances in Neural Information Processing Systems*, pages 34451–34463. Curran Associates, Inc., 2022. 2

[16] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. Awq: Activation-aware weight quantization for on-device llm compression and acceleration. *Proceedings of Machine Learning and Systems*, 6:87–100, 2024. 2

[17] Yang Lin, Tianyu Zhang, Peiqin Sun, Zheng Li, and Shuchang Zhou. Fq-vit: Post-training quantization for fully quantized vision transformer. *arXiv preprint arXiv:2111.13824*, 2021. 2

[18] Baoyuan Liu, Min Wang, Hassan Foroosh, Marshall Tappen, and Marianna Pensky. Sparse convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 806–814, 2015. 4

[19] Shih-Yang Liu, Zechun Liu, and Kwang-Ting Cheng. Oscillation-free quantization for low-bit vision transformers. In *Proceedings of the 40th International Conference on Machine Learning*, pages 21813–21824. PMLR, 2023. 2

[20] Zhenhua Liu, Yunhe Wang, Kai Han, Wei Zhang, Siwei Ma, and Wen Gao. Post-training quantization for vision transformer. *Advances in Neural Information Processing Systems*, 34:28092–28103, 2021. 1, 2

[21] Jiageng Mao, Yujing Xue, Minzhe Niu, Haoyue Bai, Jiashi Feng, Xiaodan Liang, Hang Xu, and Chunjing Xu. Voxel transformer for 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3164–3173, 2021. 2

[22] Markus Nagel, Rana Ali Amjad, Mart Van Baalen, Christos Louizos, and Tijmen Blankevoort. Up or down? adaptive rounding for post-training quantization. In *International Conference on Machine Learning*, pages 7197–7206. PMLR, 2020. 1

[23] Eunhyeok Park, Sungjoo Yoo, and Peter Vajda. Value-aware quantization for training and inference of neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 580–595, 2018. 2

[24] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[25] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017. 2

[26] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. In *Advances in Neural Information Processing Systems*, pages 23192–23204. Curran Associates, Inc., 2022. 2

[27] Sheng Shen, Zhen Dong, Jiayu Ye, Linjian Ma, Zhewei Yao, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. Q-bert: Hessian based ultra low precision quantization of bert. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8815–8821, 2020. 1

[28] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10529–10538, 2020. 5

[29] Shaoshuai Shi, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. From points to parts: 3d object detection from point cloud with part-aware and part-aggregation network. *IEEE transactions on pattern analysis and machine intelligence*, 43(8):2647–2664, 2020. 5

[30] Shaoshuai Shi, Li Jiang, Jiajun Deng, Zhe Wang, Chaoxu Guo, Jianping Shi, Xiaogang Wang, and Hongsheng Li. Pv-rcnn++: Point-voxel feature set abstraction with local vector representation for 3d object detection. *International Journal of Computer Vision*, 131(2):531–551, 2023. 5

[31] 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)*, 2020. 5

[32] Haiyang Wang, Chen Shi, Shaoshuai Shi, Meng Lei, Sen Wang, Di He, Bernt Schiele, and Liwei Wang. Dsvt: Dynamic sparse voxel transformer with rotated sets. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13520–13529, 2023. 5

[33] Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pages 38087–38099. PMLR, 2023. 1, 2, 3

[34] Hongyi Yao, Pu Li, Jian Cao, Xiangcheng Liu, Chenying Xie, and Bingzhang Wang. Rapq: Rescuing accuracy for power-of-two low-bit post-training quantization. *arXiv preprint arXiv:2204.12322*, 2022. 1

[35] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. Center-based 3d object detection and tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11784–11793, 2021. 2, 5

[36] Zhihang Yuan, Chenhao Xue, Yiqi Chen, Qiang Wu, and Guangyu Sun. Ptq4vit: Post-training quantization for vision transformers with twin uniform quantization. In *European conference on computer vision*, pages 191–207. Springer, 2022. 2

[37] Yifan Zhang, Zhen Dong, Huanrui Yang, Ming Lu, Cheng-Ching Tseng, Yuan Du, Kurt Keutzer, Li Du, and Shanghang Zhang. Qd-bev : Quantization-aware view-guided distillation for multi-view 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3825–3835, 2023. 1

[38] Sifan Zhou, Liang Li, Xinyu Zhang, Bo Zhang, Shipeng Bai, Miao Sun, Ziyu Zhao, Xiaobo Lu, and Xiangxiang Chu. Lidar-ptq: Post-training quantization for point cloud 3d object detection. *arXiv preprint arXiv:2401.15865*, 2024. 1

[39] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4490–4499, 2018. 2

A. Appendix

You may include other additional sections here.

Table 7. nuScenes Results for Different Detectors

Models	Methods	Bits(W/A)	mAP	NDS	Car	Truck	CV	Bus	Trail
	Full Prec.	32/32	59.22	66.48	84.86	57.38	16.85	70.75	38.1
CP-Voxel	SQ-0.80	8/8	59.16	66.40	84.69	57.31	16.89	70.70	38.1
	SQ-X.XX	4/8	x.x	x.x	x.x	x.x	x.x	x.x	x.x

Table 8. KITTI Results for Different Detectors

Models	Methods	Bits(W/A)	Car	Pedestrian	Cyclist
	Full Prec.	32/32	78.62	52.97	67.14
SECOND	Max-min	8/8	78.23	52.96	62.01
	SQ-0.60	8/8	78.69	52.97	67.03
	Max-min	4/8	69.41	42.81	52.99
	SQ-0.65	4/8	78.29	54.72	64.03
	Full Prec.	32/32	77.28	52.30	62.71
PointPillar	Max-min	8/8	74.68	50.83	60.44
	SQ-0.70	8/8	76.79	51.96	62.84
	Max-min	4/8	63.74	44.15	55.50
	SQ-0.35	4/8	75.11	49.79	60.02