Appendix

A. Details of the Fusion of 2D and 3D Features

The comparative performance of various fusion techniques is delineated in Table A1. This comparison includes methods such as concatenation, LiDAR-guided deformable-attention-based fusion [5] and LiDAR-camera self-attention-based fusion [34]. Analysis of the results in Table A1 reveals that while both deformable-attention-based and LiDAR-camera self-attention-based fusion techniques offer improvements in accuracy, they entail a significant increase in latency. Furthermore, these methods exhibit heightened sensitivity to QAT: the accuracy decline in both deformable-attention-based and LiDAR-camera self-attention-based fusions is more pronounced compared to the simpler concatenation approaches. Consequently, this study adopts feature concatenation as the primary method for multi-modality fusion.

B. Quantization of the 3D Branch

Algorithm 1 shows our quantization-aware training for LiDAR-based 3D Object Detection. It begins by initializing a pre-trained model and selecting a calibration subset from the LiDAR dataset. Then it defines a range for the quantization parameters. In the next step, each layer of the model undergoes calibration using a Max-min Calibrator, which determines the range of weights and activations. Based on these ranges, the quantization parameters for each layer are initialized.

The algorithm proceeds to optimize these parameters through a grid search, applying different sets of parameters to the model and evaluating performance on the calibration set to identify the best set. Once the optimal quantization parameters are determined, they are applied across all layers of the model. The model then undergoes training on the full dataset, involving standard procedures like forward passes, backpropagation, and parameter updates. The final output of the algorithm is a trained quantized model, tailored to maintain high performance despite the constraints imposed by quantization, and specifically optimized for the nuances of LiDAR data.

Table A1. Comparation of different multi-modality feature fusion approaches on nuScenes validation dataset.

Method	mAP	NDS	FPS
Deformable-attention-based Fusion (32-bit) [5]	71.5	73.9	2.9
Self-attention-based Fusion (32-bit) [34]	71.8	74.2	1.2
Concatenation (32-bit)	70.9	73.4	4.4
Deformable-attention-based Fusion (8-bit) [5]	69.7	72.2	8.5
Self-attention-based Fusion (8-bit) [34]	69.9	72.5	6.2
Concatenation (8-bit)	69.4	71.9	13.2

Algorithm 1 Quantization Aware Training for Q-TempFusion

```
Given 3D LiDAR Point Cloud Dataset $D$, Model $M$; Output: Quantized Model $M_{quantized}$;
// Step1: Initialize Model $M$, GridSearchSpace,
    CalibrationSet $\subseteq D$
foreach $layer$ in $M$ do
    Determine $WeightRange$ and $ActivationRange$ using Max-min Calibrator; Initialize quantization parameters;
end
// Step2: Optimize Quantization Parameters
foreach $param$ in $GridSearchSpace$ do
    Apply $param$, Evaluate $M$ on $CalibrationSet$; Update $param_{best}$ if performance improves;
end
// Step3: Apply Best Parameters
foreach $layer$ in $M$ do
    Apply $param_{best}$;
end
// Step4: Train Quantized Model
foreach $epoch$ do
    foreach $batch$ in $D$ do
        Forward pass, Backpropagate, Update $M$;
    end
end
$M_{quantized} \leftarrow M$; Return $M_{quantized}$;
```

C. Outlier-aware training for systematic channel-wise outliers quantization

The algorithm of outlier-aware training is divided into three steps, see Algorithm 2.

D. Baselines for the Experiments

In this paper, we conduct a comprehensive comparison of our method against a majority of the leading multi-modality baseline detection models. In our comparative analysis, camera-only methods and some of LiDAR-only methods are not included. This exclusion is based on the established understanding that such methods generally yield lower accuracy compared to multi-modality models. Also, it is pertinent to note that our comparative analysis excludes [5] due to the fairness issue in comparison. The methodology in [5] builds upon [36] and employs specific data augmentation techniques on the nuScenes dataset, which are not utilized in other detection baseline models. This discrepancy in data handling practices potentially skews a direct comparison. Therefore, for [36], we present its results prior to the application of data augmentation in Table 1. Additionally, it is noteworthy that the latency results reported

Algorithm 2 Outlier-Aware Training

```
Given the full-precision ViT Model, the test subdataset of target
    Dataset $D$, the number of encoder blocks $L$, Epoch$_s$ for
    searching, Epoch$_f$ for fine-tuning, and quantized low-bit $b$;
    // Step1: Initialize the PTC $r$ with the outlier
    estimated from the full-precision Model.
foreach $l \in [0, 1, \ldots, L-1]$ do
    $i_o$, $r_o$ = Check_Outliers$_{3\sigma}$(Model$_l$, D)  Quantize Model$_l$ by
      Eq. (2) with $i_o$, $r_o$, $b$ into the QModel$_l$;
end
// Step2: Search for the channel index of outliers and
    determine the PTR $r$ by the l2 regularization.
$r = r_o$, $i$=$i_o$;
 foreach $eps \in [0, 1, \ldots, Epoch_s - 1]$ do
    // These two operations are gradient-free.
    $i$, $r$ = Check_Outliers$_{3\sigma}$(Model$_l$); $r_i = \underset{r_i \in \{1,2,\ldots,R\}}{argmin} ||X_i -$
        $\lfloor \frac{X_i}{2^{r_i} s} \rceil \cdot 2^{r_i} s||_2$ ;
        // Following Eq. (4) and Eq. (2).
    task$_{loss}$, quantization$_{loss}$ = Quantize(QModel, $b$);
      Backward(task$_{loss}$, quantization$_{loss}$)
end
// Step3: Finetune the $b$-bit quantizaton.
Fix $r$ and $i$ for outliers and quantize QModel with fine-tune
  Epoch$_f$;
  Finalize the quantized ViT Model.
```

in [62] exhibit slight discrepancies when compared to our experimental findings. Consequently, the results presented in this paper are based on our own experimental evaluations.