MVAT: Multi-View Aware Teacher for Weakly Supervised 3D Object Detection

Supplementary Material

A. Implementation Details

Data Pre-processing For 2D mask generation from box prompts, we use SAM 2 and filter out masks with a confidence score below 0.6. When identifying static objects, we use a centroid variance threshold of 0.3m across frames. For the subsequent cleaning of aggregated point clouds (P^s_{agg}) before coarse box estimation, we apply DBSCAN with an ϵ of 0.5m and a minimum of 10 points per cluster.

Motion Classification To reliably separate static from moving instances for temporal aggregation, we implement a straightforward yet robust motion analysis procedure based on an object's displacement in a global reference frame.

First, leveraging the provided ego-pose data from the dataset, all object-centric point clouds within a given sequence are transformed into a common global coordinate frame. This step ensures that any observed motion is from the object itself, not due to the ego-vehicle's movement.

For a given object instance j, which is tracked across a set of frames $\mathcal{F}_j$, we compute its point cloud centroid $c_{t,j}$ in this global frame for each timestamp $t \in \mathcal{F}_j$. An object is then classified as **static** if the maximum Euclidean distance between any pair of its observed centroids throughout its entire track is below a predefined displacement threshold, τ_{static}.

Formally, an object j is considered static if:

$$\max_{i,k \in \mathcal{F}_j} \|c_{i,j} - c_{k,j}\|_2 < \tau_{\text{static}} \tag{5}$$

In our experiments, we empirically set $\tau_{\text{static}} = 0.5$meter, a value that robustly accounts for potential minor centroid jitter caused by partial occlusions, point cloud sparsity, or slight inaccuracies in the provided ego-pose estimation. This criterion, which considers an object's full trajectory rather than just consecutive frames, effectively identifies globally stationary objects whose point clouds can be confidently aggregated.

3D Box Coarse Estimation with Geometric Verification The goal of this step is to derive a coarse-yet-robust 3D pseudo-box, B^{coarse}_{3D}, from each aggregated static point cloud cluster, C^*. We acknowledge that naively fitting a bounding box using methods like Principal Component Analysis (PCA) can be fragile. This can lead to significant errors, particularly for sparse point clouds or those with irregular distributions arising from occlusions (e.g., 'L'-shaped views). To mitigate this, our coarse estimation pipeline incorporates a geometric verification step to filter out unreliable instances before they are used to train the Teacher network.

The process is twofold:

1. **Initial Box Hypothesis via PCA**: We first generate an initial box hypothesis. By applying PCA to the bird's-eye view (BEV) projection of the points in C^*, we determine the primary orientation and initial extent of the object. This provides an estimate for the BEV center (c_x, c_y), size (l, w), and heading θ. The vertical parameters (c_z, h) are subsequently derived from the min/max z-coordinates of the points.
2. **Geometric Consistency Verification**: To validate the quality of this PCA-derived box, we perform a consistency check in the BEV plane. We compute the **2D convex hull** of the point cluster, which provides a tight geometric footprint of the object's observed shape. We then calculate the **Intersection-over-Union (IoU)** between the area of the PCA-estimated BEV box and the area of this convex hull. An instance is deemed geometrically reliable and retained for the Teacher's burn-in phase only if this shape-consistency IoU exceeds a predefined threshold, τ_{IoU}:

$$\text{IoU}(\text{Box}^{\text{BEV}}_{\text{PCA}}, \text{ConvexHull}(C^*_{\text{BEV}})) > \tau_{\text{IoU}} \tag{6}$$

Instances failing this check are discarded from the initial training set for the Teacher. In our experiments, we set $\tau_{\text{IoU}} = 0.6$. This verification step is crucial as it prunes cases where PCA fails, ensuring that the Teacher is bootstrapped using only pseudo-labels with a high degree of geometric fidelity to the underlying point cloud data.

Pseudo-Label Filtering for Student Training A critical step before the knowledge distillation phase is to generate a high-quality set of pseudo-labels, B^{Teacher}_{3D}, to supervise the Student network. A naive transfer of all the Teacher's predictions would propagate errors and degrade the Student's performance. To prevent this, and to address any ambiguity in our filtering methodology, we employ a rigorous two-stage process on the Teacher's outputs for each object instance.

1. **Class-Consistency Check**: First, we ensure semantic correctness. Our Teacher network outputs both a 3D box and a class prediction for each object. We perform an initial filtering pass by comparing the Teacher's predicted class with the ground-truth class label that is provided with the original 2D bounding box annotation. Any instance where the predicted class does not match the ground-truth class is immediately discarded. This

step eliminates major classification failures and ensures the Student learns the correct class associations.

2. **Confidence-Based Filtering**: Second, among the class-consistent predictions, we filter based on the model's certainty. The Teacher network also outputs a confidence score for each prediction, which reflects its certainty in both the classification and the localization quality of the 3D box. We only retain an instance for the Student's training set if its associated confidence score exceeds a predefined, class-specific threshold, $\tau_{\text{conf}}^{(c)}$, where c denotes the object class.

This two-stage verification ensures that the Student is trained on a "clean" dataset of pseudo-labels, where objects are correctly classified and their 3D boxes are predicted with high confidence by the Teacher. These thresholds are set based on the validation set to balance the trade-off between the quantity and quality of pseudo-labels. For example, in our experiments, we used values such as $\tau_{\text{conf}}^{(\text{Car})} = 0.5$ and $\tau_{\text{conf}}^{(\text{Pedestrian})} = 0.4$. This transparent and reproducible filtering mechanism is essential for effective knowledge distillation and prevents the propagation of low-quality predictions.

Training Details Our framework is implemented on top of the CenterPoint detector architecture. For the *Teacher Burn-in* phase, we train on the filtered set of static objects. The loss weight for the training of the Teacher (Eq. 3) is set to $\lambda = 0.5$ for the 2D multi-view loss. For the *Student Distillation* phase, the teacher's weights are frozen and the student network is trained from scratch using the full dataset. The loss weight for the student's training (Eq. 4) is set to $\gamma = 0.5$ for the 2D multi-view loss.

PV-RCNN settings The PV-RCNN model parameters were primarily configured based on the KITTI dataset's optimal settings, which were then consistently applied across Waymo and nuScenes datasets. Training involved 80 epochs with a batch size of 24 and a learning rate of 0.01. The 3D voxel CNN backbone had four levels with feature dimensions 16, 32, 64, and 64 respectively. The Voxel Set Abstraction (VSA) module utilized two neighboring radii for each level, specifically (0.4m, 0.8m), (0.8m, 1.2m), (1.2m, 2.4m), and (2.4m, 4.8m), with raw points also using (0.4m, 0.8m). Keypoint sampling was set at 2,048. The Rol-grid pooling operation uniformly sampled $6 \times 6 \times 6$ grid points per 3D proposal, with two neighboring radii of (0.8m, 1.6m). Voxel sizes were (0.05m, 0.05m, 0.1m), and the detection range was $[0, 70.4]m$ (X), $[-40, 40]m$ (Y), and $[-3, 1]m$ (Z). The model was trained end-to-end using the ADAM optimizer, incorporating random flipping along the X axis, global scaling with a random factor from $[0.95, 1.05]$, global rotation around the Z axis with a random angle from $[-\frac{\pi}{4}, \frac{\pi}{4}]$, and ground-truth sampling for data augmentation. During inference, the top-100 proposals generated by the 3D voxel CNN were kept with a 3D IoU threshold of 0.7 for non-maximum-suppression (NMS), and a final NMS threshold of 0.01 was applied.

The overall 3D training loss ($\mathcal{L}_{\text{3D}}$) was a sum of the region proposal loss, 3D segmentation loss, and proposal refinement loss, all with equal weights. The region proposal loss includes focal loss for classification and smooth-L1 for anchor box regression. The 3D segmentation loss also uses focal loss for keypoint segmentation. The proposal refinement loss comprises an IoU-guided confidence prediction loss, which uses cross-entropy on a normalized IoU target, and a box refinement loss based on smooth-L1.

CenterPoint settings We used the exact same architecture as in the original paper for the CenterPoint model. The architecture consists of a standard 3D backbone that extracts map-view feature representation from Lidar point-clouds. Subsequently, a 2D CNN architecture detection head identifies object centers and regresses to full 3D bounding boxes using center features. This box prediction is then utilized to extract point features at the 3D centers of each face of the estimated 3D bounding box, which are passed into an MLP to predict an IoU-guided confidence score and box regression refinement. All first-stage outputs share a 3×3 convolutional layer, Batch Normalization, and ReLU, with each output then branching into two 3×3 convolutions separated by a batch norm and ReLU. The second stage employs a shared two-layer MLP, including a batch norm, ReLU, and Dropout with a drop rate of 0.3, followed by two branches of three fully-connected layers for confidence score and box regression prediction.

The CenterPoint model utilizes specific hyperparameters for optimal performance across different datasets, namely nuScenes and Waymo Open Dataset. For the nuScenes dataset, the model is optimized using the AdamW optimizer with a one-cycle learning rate policy, setting the maximum learning rate at 10^{-3}, a weight decay of 0.01, and momentum ranging from 0.85 to 0.95. Training is conducted with a batch size of 16 over 20 epochs on 4 V100 GPUs. The detection range for nuScenes is defined as [-51.2m, 51.2m] for the X and Y axes and [-5m, 3m] for the Z axis. Depending on the encoder, the voxel size for CenterPoint-Voxel is (0.1m, 0.1m, 0.2m) and the grid size for CenterPoint-Pillars is (0.2m, 0.2m). Data augmentation techniques include random flipping along both X and Y axes , global scaling with a random factor between [0.95, 1.05], random global rotation within $[-\pi/8, \pi/8]$, and ground-truth sampling to address class distribution imbalances. For the second stage refinement during training, 128 boxes are randomly sampled with a 1:1 positive-negative ratio, where a positive proposal overlaps with a ground truth annotation by at least

0.55 IoU. During inference, the second stage processes the top 500 predictions after Non-Maxima Suppression (NMS). For the nuScenes test set submission, a finer input grid size of $0.075m \times 0.075m$ was used, incorporating two separate deformable convolution layers in the detection head.

Conversely, for the Waymo Open Dataset, the CenterPoint model employs a learning rate of 3×10^{-3} and is trained for 30 epochs. The detection range is set to [-75.2m, 75.2m) for the X and Y axes, and [-2m, 4m] for the Z axis. CenterPoint-Voxel utilizes a voxel size of (0.1m, 0.1m, 0.15m), while CenterPoint-Pillar uses a grid size of (0.32m, 0.32m). Data augmentation for Waymo includes random flipping along both X and Y axes, global scaling between [0.95, 1.05], and a random global rotation of $[-\pi/4, \pi/4]$. For various ablation studies, the model was finetuned for 6 epochs with the second stage refinement modules.

Dataset Considerations Our experiments are conducted on sequential datasets like nuScenes and Waymo. While the KITTI dataset is prominent in 3D detection, it does not provide the sequential, multi-view data that is essential for our method's temporal aggregation strategy.

B. Number of view frames per object for each class.

To better understand the characteristics of our dataset and the potential benefits of temporal aggregation, we analyze the distribution of view frames per object across different classes. Figure 5 illustrates this distribution, revealing significant variations among object categories. The median number of frames in which an object appears varies substantially across classes. Traffic cones exhibit the lowest median visibility at only 5 frames per instance, which can be attributed to their small size and frequent occlusion in urban environments. In contrast, buses maintain the highest visibility with a median of 18 frames per instance, likely due to their large size and tendency to remain in the field of view for extended periods. This class-dependent distribution of temporal visibility has important implications for our method. Classes with higher frame counts (buses, trucks, and cars) benefit more from our temporal aggregation approach, as more views contribute to a more complete 3D representation. Conversely, objects with fewer frames (traffic cones, pedestrians) present a greater challenge, as the temporal aggregation provides more limited improvements over single-frame detection.

C. Qualitative results of the PCA BEV box estimation.

We present visualizations of our coarse 3D box estimation, where PCA is applied to the BEV-projected aggregated point clouds. As shown in Figure 9, when objects

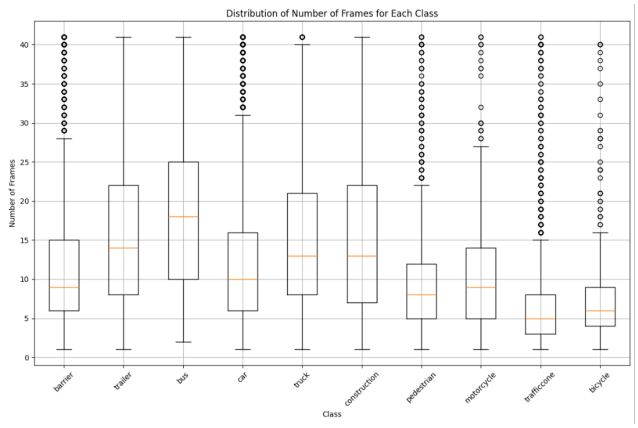

Figure 5. Distribution of the number of frames for objects of each class

are clearly visible, our method accurately estimates the 3D center, dimensions, and orientation. For example, the Barrier, Car, and Truck examples illustrate that the estimated boxes closely align with the object boundaries and correctly capture their orientations. These results demonstrate that our PCA-based estimation reliably produces high-quality coarse 3D annotations under favorable conditions, providing robust supervisory signals for subsequent 3D detection stages.

D. Detailed quantitative results of the coarse 3D box estimation.

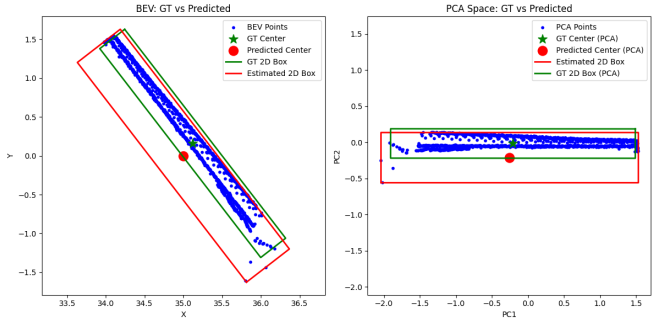

Figure 6. Barrier Example

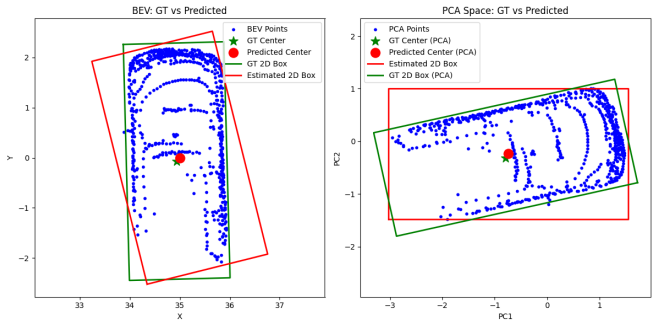

Figure 7. Car Example 1

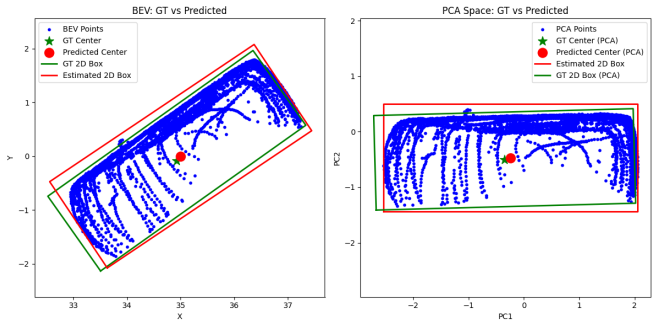

Figure 8. Car Example 2

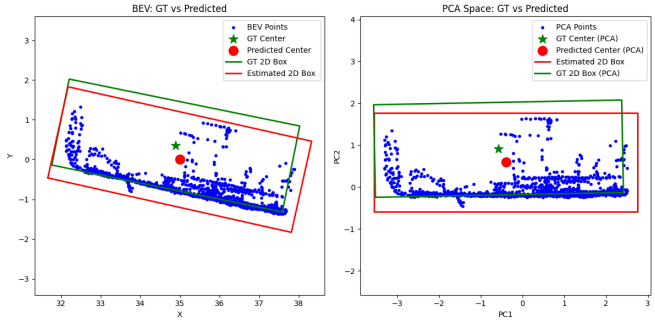

Figure 9. Truck Example

Category	Barrier	Car	Truck	Motorcycle	Traffic Cone	Pedestrian	Construction	Bicycle	Bus	Trailer	Average
3D IoU	44.7	53.3	49.5	50.8	48.6	53.6	49.4	46.2	42.7	53.2	**49.2**

Table 5. Evaluation in 3D IoU between the 3D coarse estimations B_{3D}^{coarse} and ground truth 3D boxes for the samples used to train the teacher network.

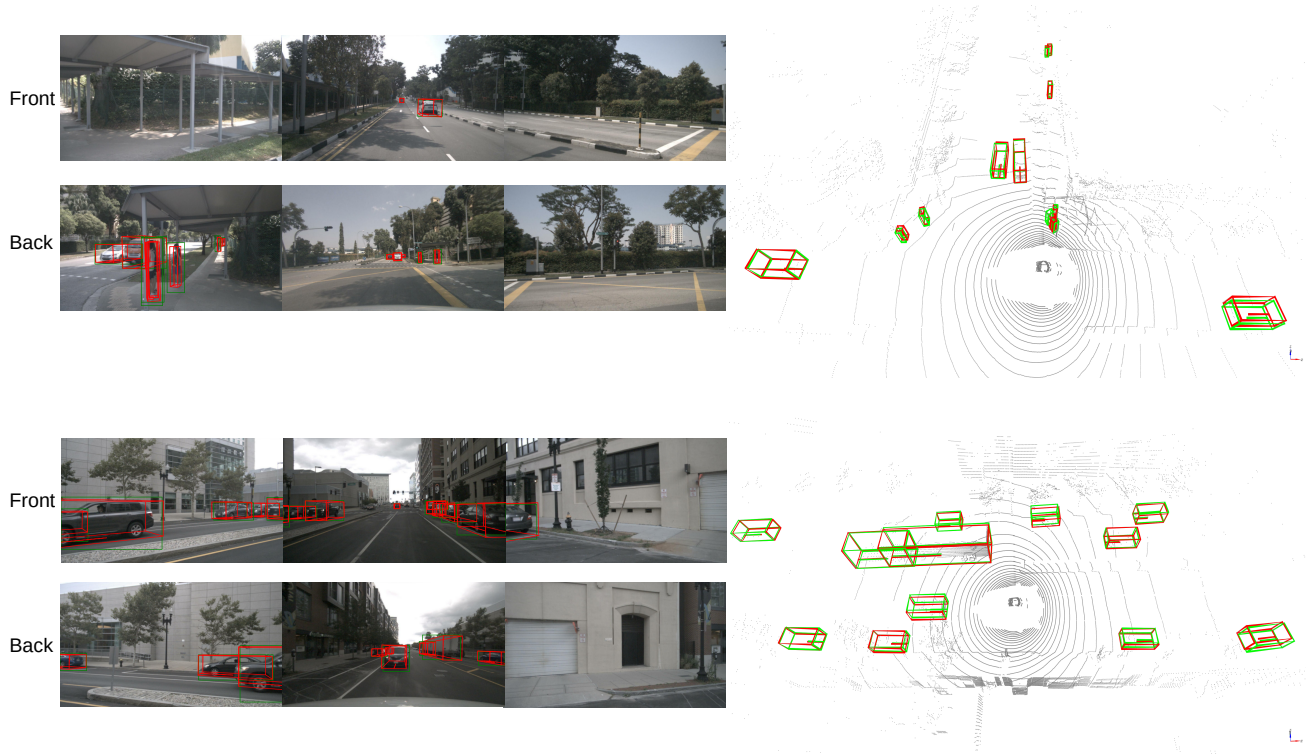

Figure 10. Qualitative results of CenterPoint [20] trained using 3D pseudo-labeled by MVAT on the nuScenes validation set. We show the ground truth boxes in green and the predictions in red.