Learning to Detect Mobile Objects from LiDAR Scans Without Labels

Yurong You*†,1 Katie Luo*,1 Cheng Perng Phoo[1] Wei-Lun Chao[2]
Wen Sun[1] Bharath Hariharan[1] Mark Campbell[1] Kilian Q. Weinberger[1]
[1]Cornell University [2]The Ohio State University

Abstract

Current 3D object detectors for autonomous driving are almost entirely trained on human-annotated data. Although of high quality, the generation of such data is laborious and costly, restricting them to a few specific locations and object types. This paper proposes an alternative approach entirely based on unlabeled data, which can be collected cheaply and in abundance almost everywhere on earth. Our approach leverages several simple common sense heuristics to create an initial set of approximate seed labels. For example, relevant traffic participants are generally not persistent across multiple traversals of the same route, do not fly, and are never under ground. We demonstrate that these seed labels are highly effective to bootstrap a surprisingly accurate detector through repeated self-training without a single human annotated label. Code is available at https://github.com/YurongYou/MODEST.

1. Introduction

Autonomous driving promises to revolutionize how we transport goods, travel, and interact with our environment. To safely plan a route, a self-driving vehicle must first perceive and localize mobile traffic participants such as other vehicles and pedestrians in 3D. Current state-of-the-art 3D object detectors are all based on deep neural networks [45, 48, 49, 61] and can yield up to 80 average precision on benchmark datasets [21, 22].

However, as with all deep learning approaches, these techniques have an insatiable need for labeled-data. Specifically, to train a 3D object detector that takes LiDAR scans as input, one typically needs to first come up with a list of objects of interest and annotate each of them with tight bounding boxes in the 3D point cloud space. Such a data annotation process is laborious and costly, but worst of all, the resulting detectors only achieve high accuracy when the training and test data distributions match [58]. In other words, their accuracy deteriorates over time and space, as looks and shapes of cars, vegetation, and background objects change. To guarantee good performance, one has to collect labeled training data for specific geo-fenced areas and re-label data constantly, greatly limiting the applicability and development of self-driving vehicles.

These problems motivate the question: *Can we learn a 3D object detector from unlabeled LiDAR data?* Here, we focus on "mobile" objects, i.e., objects that might move, which cover a wide range of traffic participants. At first glance, this seems insurmountably challenging. After all, how could a detector know just from the LiDAR point cloud that a pedestrian is a traffic participant and a tree is not? We tackle this problem with the help of two important insights: 1) we can use simple heuristics that, even without labeling, can occasionally distinguish traffic participants from background objects more or less reliably; 2) if data is noisy but diverse, neural networks excel at identifying the common patterns, allowing us to repeatedly self-label the remaining objects, starting from a small set of seed labels.

Weak labels through heuristics. We build upon a simple yet highly generalizable concept to discover mobile objects — *mobile objects are unlikely to stay persistent at the same location over time.* While this requires unlabeled data at multiple timestamps for the same locations, collecting them is arguably cheaper than annotating them. After all, many of us drive through the same routes every day (e.g., to and from work or school). Even when going to new places, the new routes for us are likely frequent for the local residents.

Concretely, whenever we discover multiple traversals of one route, we calculate a simple ephemerality statistic [3] for each LiDAR point, which characterizes the change of its local neighborhood across traversals. We cluster LiDAR points according to their coordinates and ephemerality statistics. Resulting clusters with high ephemerality statistics, and located on the ground, are considered as mobile objects and are further fitted with upright bounding boxes.

Self-training (ST). While this initial *seed* set of mobile objects is not exhaustive (e.g., parked cars may be missed) and somewhat noisy in shape, we demonstrate that an object detector trained upon them can already learn the underlying object patterns and is able to output more and higher-

*Denotes equal contribution.
†Correspondences could be directed to `yy785@cornell.edu`

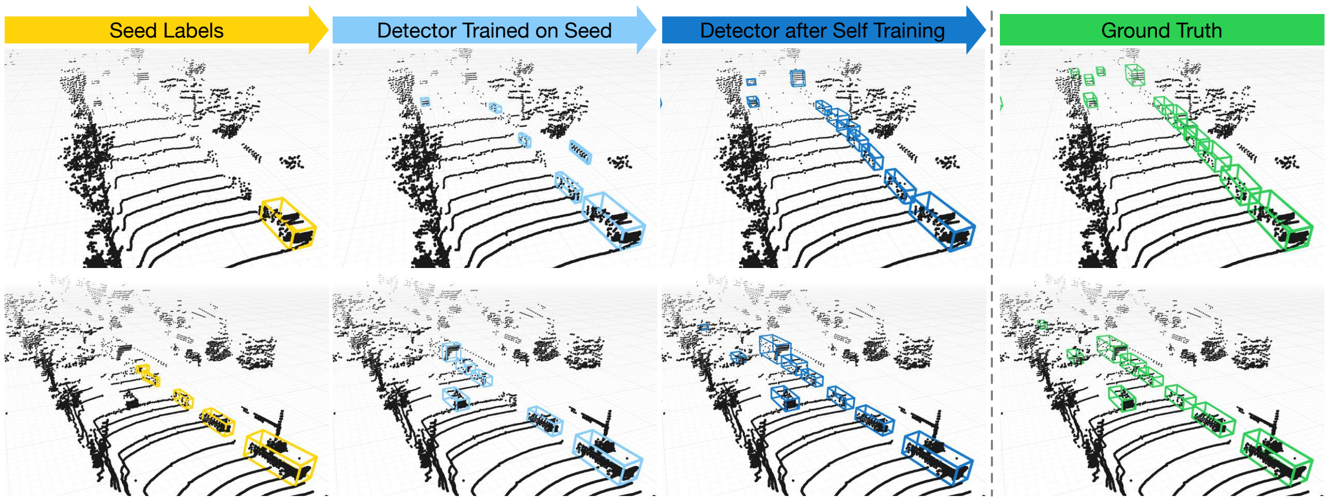


Figure 1. **Visualizations of MODEST outputs.** We show LiDAR scans from two scenes in the Lyft dataset in two rows. From *zero labels*, our method is able to bootstrap a detector that achieves results close to the ground truth. The key insight is to utilize noisy "seed" labels produced from an ephemerality score and filtered with common-sense properties, and self-train upon them to obtain high quality results.

quality bounding boxes than the seed set. This intriguing observation further opens up the possibility of using the detected object boxes as "better" pseudo-ground truths to train a new object detector. We show that such a self-training cycle [32, 59] enables the detector to improve itself over time; notably, it can even benefit from additional, unlabeled data that do not have multiple past traversals associated to them.

We validate our approach, MODEST (**M**obile **O**bject **D**etection with **E**phemerality and **S**elf-**T**raining) on the Lyft Level 5 Perception Dataset [29] and the nuScenes Dataset [6] with various types of detectors [31, 49, 60, 70]. We demonstrate that MODEST yields remarkably accurate mobile object detectors, comparable to their supervised counterparts. Concretely, our contributions are three-fold:

1. We propose a simple, yet effective approach to identifying "seed" mobile objects from multiple traversals of LiDAR scans using *zero labels*.
2. We show that using these seed objects, we can bootstrap accurate mobile object detectors via self-training.
3. We evaluate our method exhaustively under various setting and demonstrate consistent performance across multiple real-world datasets.

2. Related Works

We seek to build object detectors without any human supervision. We briefly discuss several related research areas.

3D object detection and existing datasets. Most existing 3D object detectors take 3D point clouds generated by LiDAR as input. They either consist of specialized neural architectures that can operate on point clouds directly [45–47, 49, 63] or voxelize the point clouds to leverage 2D or 3D convolutional neural architectures [10, 31, 34, 48, 60, 61, 69, 70]. Regardless of the architectures, they are trained using supervision and their performances hinges directly on the training dataset. However, the limited variety of objects and driving conditions in existing autonomous driving datasets [6, 7, 21, 22, 29] impedes the generalizability of the resulting detectors [58].

Generalizing these to new domains requires a fresh labeling effort. In contrast, our unsupervised approach automatically discovers all the traffic participants, and can be used to train detectors in any new condition without any labeling.

Unsupervised Object Discovery in 2D/3D. Our work follows prior work on discovering objects both from 2D images as well as from 3D data. A first step in object discovery is to identify candidate objects, or "proposals" from a single scene/image. For 2D images, this is typically done by segmenting the image using appearance cues [11, 20, 35, 56], but color variations and perspective effects make this difficult. Tian *et al.* [53] exploits the correspondence between images and 3D point clouds to detect objects in 2D. In 3D scenes, one can use 3D information such as surface normals [14, 20, 26–28, 30, 50, 55]. One can also use temporal changes such as motion [13, 17, 30, 35, 36]. Our work combines effective 3D information with cues from changes in the scene over time to detect mobile objects [26, 27, 37]. In particular, similar to our approach, Herbst et al. [26, 27] reconstruct the same scene at various times and carve out dissimilar regions as mobile objects. We use the analogous idea of ephemerality as proposed by Barnes et al. [3]. We show in our work that this idea yields a surprisingly accurate set of initial objects. In addition, we also leverage other common-sense rules such as locations of the objects (e.g. objects should stay on the ground) [9, 15, 16, 38] or shapes of an object (e.g. objects should be compact) [15, 16, 28]. However, crucially, we do not just stop at this proposal stage. Instead, we use these seed labels to train an object detector through multiple rounds of self-training. This effectively identifies objects consistent across multiple scenes. While previous work has attempted to use this consistency cue [1, 11, 50, 55–57, 66] (including co-segmentation [19, 33, 64]), prior work typically uses clustering to accomplish this. In contrast, we demonstrate that neural network training and self-training provides a very strong signal and substantially improves the quality of the proposals or seed labels.

Self-training, semi-supervised and self-supervised learning. When training our detector, we use self-training, which has been shown to be highly effective for semi-supervised learning [32, 59], domain adaptation [8, 39, 62, 67, 71, 72] and few-shot/transfer learning [23,42,43,54]. Interestingly, we show that self-training can not only discover more objects, but also correct the initially noisy box labels. This result that neural networks can denoise noisy labels has been observed before [2,25,41,44]. Self-training also bears resemblance to other semi-supervised learning techniques [4, 5, 24, 51] but is simpler and more broadly applicable.

3. Method

Problem setup. We want a detector that detects *mobile* objects, i.e., objects that might move, in LiDAR point clouds. We wish to train this detector only from *unlabeled* data obtained simply by driving around town, using a car equipped with synchronized sensors (in particular, LiDAR which provides 3D point clouds and GPS/INS which provides accurate estimates of vehicle position and orientation). Such a data collection scheme is practical and requires no annotators; indeed, it can be easily collected as people go about their daily lives. We assume that this unlabeled data include at least a few locations that have been visited *multiple times*; as we shall see, this provides us with a very potent learning signal for identifying mobile objects.

Overview. We propose simple, high-level common-sense properties that can easily identify a few *seed* objects in the unlabeled data. These discovered objects then serve as labels to train an off-the-shelf object detector. Specifically, building upon the neural network's ability to learn consistent patterns from initial seed labels, we bootstrap the detector by self-training [32, 59] using the same unlabeled data. The self-training process serves to correct and expand the initial pool of seed objects, gradually discovering more and more objects to further help train the detector. The whole process is summarized in Algorithm 1.

3.1. Discovering objects through common-sense

What properties define mobile objects or traffic participants? Clearly, the most important characteristic is that they are *mobile*, i.e., they move around. If such an object is spotted at a particular location (e.g., a car at an intersection), it is unlikely that the object will still be there when one visits the intersection again a few days hence. In other words, mobile objects are *ephemeral* members of a scene [3]. Of course, occasionally mobile objects like cars might be parked on the road for extended periods of time. However, for the most part ephemeral objects are likely to be mobile objects.

What other properties do mobile objects have? It is clear that they must be on the ground, not under the ground or above in the sky. They are also likely to be smaller than buildings. One can come up with more, but we find that these intuitive, common-sense properties serve as sufficient constraints for mining objects from unlabeled data.

Building upon these two sets of properties, we propose a bottom-up approach, which begins with identifying points that are ephemeral, followed by clustering them into seed objects that obey these common-sense properties. In the following sections, we discuss the implementations and visualize an example seed label generation in Figure 2.

3.1.1 Identifying ephemeral points

We assume that our unlabeled data include a set of locations L which are traversed multiple times in separate driving sessions (or *traversals*). For every traversal t through location $c \in L$, we aggregate point clouds captured within a range of $[-H_s, H_e]$ of c to produce a dense 3D point cloud $\boldsymbol{S}_c^t$ for location c in traversal t[1]. We then use these dense point clouds $\boldsymbol{S}_c^t$ to define ephemerality as described by Barnes et al. [3]. Concretely, to check if a 3D point $\boldsymbol{q}$ in a scene is ephemeral, for each traversal t we can count the number $N_t(\boldsymbol{q})$ of LiDAR points that fall within a distance r to $\boldsymbol{q}$,

$$N_t(\boldsymbol{q}) = \left|\left\{\boldsymbol{p}_i \mid \|\boldsymbol{p}_i - \boldsymbol{q}\|_2 < r, \boldsymbol{p}_i \in \boldsymbol{S}_c^t\right\}\right|. \quad (1)$$

If $\boldsymbol{q}$ is part of the static background, then its local neighborhood will look the same in all traversals, and so the counts $N_t(\boldsymbol{q})$ will be all similar. Thus, we can check if $\boldsymbol{q}$ is ephemeral by checking if $N_t(\boldsymbol{q})$ is approximately uniform across traversals. To this end, we compute

$$P(t; \boldsymbol{q}) = \frac{N_t(\boldsymbol{q})}{\sum_{t'=1}^{T} N_{t'}(\boldsymbol{q})}. \quad (2)$$

and define the *persistence point score* (PP score) as:

$$\tau(\boldsymbol{q}) = \begin{cases} 0 & \text{if } N_t(\boldsymbol{q}) = 0 \;\; \forall t; \\ \frac{H(P(t;\boldsymbol{q}))}{\log(T)} & \text{otherwise.} \end{cases} \quad (3)$$

Here $H(\cdot)$ is the information entropy, T is the number of traversals through location c, and $\log(T)$ a normalizer to guarantee that $\tau(\mathbf{q}) \in [0, 1]$.[2] A high PP score implies a

[1] We can easily transform captured point clouds to a shared coordinate frame via precise localization information through GPS/INS.

[2] The KL divergence between P and the uniform distribution $f(t) = 1/T$ is $KL(P(t;\boldsymbol{q})||f(t)) = \log(T) - H(P(t;\boldsymbol{q}))$ and high entropy implies large similarity with the uniform distribution.

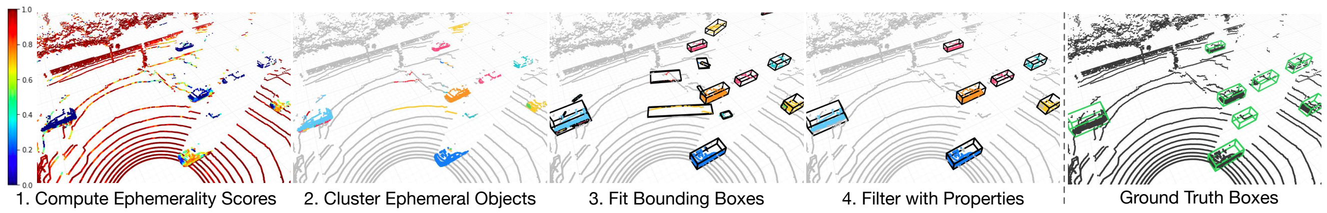


Figure 2. **Generation of seed labels.** Seed labels from object discovery are used to train downstream detectors. We begin by computing the PP score for each point. Then we segment out clusters that are non-persistent and apply a box-fitting algorithm to each cluster. We filter out superfluous bounding boxes using our common-sense assumptions. This entire process is supervision-free.

high entropy of the distribution $P(\cdot;\boldsymbol{q})$, which means that the counts $N_t(\boldsymbol{q})$ over t are all similar, indicating that the point $\boldsymbol{q}$ is part of the static background (see 1. in Figure 2).

3.1.2 From ephemeral points to ephemeral objects

We compute the PP score for each point in the LiDAR point clouds collected at a location c using multiple traversals. This automatically surfaces non-persistent (and thus mobile) objects as blobs of points with *low PP scores*. We segment out these blobs automatically using the following straightforward clustering approach. First, we construct a graph whose vertices are points in the point cloud. The edges in the graph connect each point only to its mutual K-nearest neighbors in 3D within a distance r'. Each edge between points $\boldsymbol{p}$ and $\boldsymbol{q}$ is assigned a weight equal to the difference between their PP scores, i.e.,

$$w(e_{\boldsymbol{q},\boldsymbol{p}}) = |\tau(\boldsymbol{q}) - \tau(\boldsymbol{p})|. \tag{4}$$

The graph structure together with the edge weights define a new metric that quantifies the similarity between two points. In this graph, two points that are connected by a path are considered to be close if the path has low total edge weight, namely, the points along the path share similar PP scores, indicating these points are likely from the same object. In contrast, a path in the graph that has high total edge weight likely goes across the boundary of two different objects (e.g., a mobile object and the background). Many graph-based clustering algorithms can fit the bill. We deploy the widely used DBSCAN [18] algorithm for the clustering due to its simplicity and its ability to cluster without the need of setting the number of clusters beforehand. DBSCAN returns a list of clusters, from which we remove clusters of static (and thus persistent) background points by applying a threshold γ on the α percentile of PP scores in a cluster (i.e., remove the cluster if the α percentile of the PP scores in this cluster is larger than γ). We then apply a straightforward bounding box fitting algorithm [68] to fit an up-right bounding box to each cluster.

3.1.3 Filtering using other common sense properties

Finally, following our common-sense assumptions, we remove bounding boxes that are under the ground plane, floating in the air, or having exceptional large volume (see the supplementary). This produces the final set of *seed* pseudo-ground-truth bounding boxes for mobile objects.

Algorithm 1 Mobile Object Detection with Ephemerality and Self-Training (MODEST)

```
Input: $\{\boldsymbol{P}_i\}$ LiDARs with accurate localization,
       $I_{\max}$ maximum self-training (ST) iterations
Output: $D_{I_{\max}}$ the mobile detector after $I_{\max}$ rounds ST
  $\{\{S^t\}_i\} \leftarrow$ aggregate LiDAR from other traversals
  $\{\tau(\boldsymbol{P}_i)\} \leftarrow \{$compute_PP$(\{S^t\}_i, \boldsymbol{P}_i)\}$
  $\mathcal{B}_0 \leftarrow \{$cluster_fit_boxes_filter$(\tau(\boldsymbol{P}_i), \boldsymbol{P}_i)\}$
  $D_0 \leftarrow$ train_detector$(\{\boldsymbol{P}_i\}, \mathcal{B}_0)$      ▷ 0-th round training
  for $j \leftarrow 1$ to $I_{\max}$ do                 ▷ $j$-th round self-training
      $\mathcal{B}_j \leftarrow$ get_detection$(D_{j-1}, \{\boldsymbol{P}_i\})$     ▷ pseudo-labels
      $\mathcal{B}_j \leftarrow$ filter_by_PP$(\mathcal{B}_j, \{\boldsymbol{P}_i, \tau(\boldsymbol{P}_i)\})$
      $D_j \leftarrow$ train_detector$(\{\boldsymbol{P}_i\}, \mathcal{B}_j)$
  end for
```

Discussion. Our proposed procedure for object discovery, while intuitive and fully unsupervised, has several limitations. First, the bounding boxes are produced only for locations that were traversed multiple times. Second, owing to many filtering steps that we apply, these bounding boxes are not exhaustive. For example, parked cars along the side of the road might be marked as persistent and therefore may not be identified as mobile objects. Finally, our heuristic box-fitting approach might fit inaccurate bounding boxes to noisy clusters that contain background points or miss foreground points. Thus in general, this initial set of seed bounding boxes might (a) miss many objects, and (b) produce incorrect box shapes and poses. Nevertheless, we find that this seed set has enough signal for training a high quality mobile object detector, as we discuss below.

3.2. Bootstrapping a mobile object detector

Concretely, we simply take off-the-shelf 3D object detectors [31, 49, 60, 70] and directly train them *from scratch* on these initial seed labels via minimizing the corresponding detection loss from the detection algorithms.

Intriguingly, the object detector trained in this way *outperforms* the original seed bounding boxes themselves — *the "detected" boxes have higher recall and are more ac-*

curate than the "discovered" boxes on the same training point clouds. See Figure 1 for an illustration. This phenomenon of a neural network improving on the provided noisy labels themselves is superficially surprising, but it has been observed before in other contexts [41]. The key reason is that the noise is not consistent: the initial labels are generated scene-by-scene and object-by-object. In some scenes a car may be missed because it was parked throughout all traversals, while in many others it will be discovered as a mobile object. Even among discovered boxes of similar objects, some may miss a few foreground points but others wrongly include background points. The neural network, equipped with limited capacity[3], thus cannot reproduce this inconsistency and instead instead identifies the underlying consistent object patterns.

In particular, we find that the detector substantially improves recall: it is able to identify objects (like parked cars) that were missed in the ephemerality computation, because these seemingly static objects are nevertheless similar in shape to other moving objects identified as ephemeral. We also find many cases where the detector automatically corrects the box shape, based on the average box shape of similar objects it has encountered in the training data. Finally, because the detector no longer needs multiple traversals, it can also find new mobile objects in scenes that were only visited once. In summary, this initial detector already discovers far more objects than the initial seed set, and localizes them more accurately.

Automatic improvement through self-training. Given that the trained detector has discovered many more objects, we can use the detector itself to produce an improved set of ground-truth labels, and re-train a new detector from scratch with these better ground truths. Furthermore, we can *iterate* this process: the new retrained detector has more positives and more accurate boxes for training, so it will likely have higher recall and better localization than the initial detector. As such we can use this second detector to produce a new set of pseudo-ground-truth boxes which can be used to train a third detector and so on. This iterative self-training [32, 59] process will eventually converge when the pseudo-ground truth labeling is consistent with itself and the detector can no longer improve upon it.

While it is possible for this iterative self-training to cause concept drift (e.g., the detector reinforces from its error), we find empirically that a simple thresholding step similar to that in subsubsection 3.1.2 — remove the pseudo-ground-truth box if the α percentile of the PP scores within the box is larger than γ — is highly effective in removing false positives (hence improve precision) in the self-training process and prevents performance degradation (Figure 5).

[3]We note that for detection problems, a neural network can hardly overfit the training data to achieve 100% accuracy [40], in contrast to classification problems [65].

4. Experiments

Datasets. We validate our approach on two datasets: Lyft Level 5 Perception [29] and nuScenes [6]. To the best of our knowledge, these are the only two publicly available autonomous driving datasets that have both bounding box annotations and multiple traversals with accurate localization. To ensure fair assessment of generalizability, we re-split the dataset so that the training set and test set are *geographically disjoint*; we also discard locations with less than 2 examples in the training set. This results a train/test split of 11,873/4,901 point clouds for Lyft and 3,985/2,324 for nuScenes. To construct ground truth labels, we group all the traffic participants types in the original datasets into a single "mobile" object. Note that *the ground-truth labels are only used for evaluation, not training.*

In addition, we convert the raw Lyft and nuScenes data into the KITTI format to leverage off-the-shelf 3D object detectors that is predominantly built for KITTI [21]. We use the roof LiDAR (40 or 60 beam in Lyft; 32 beam in nuScenes), and the global 6-DoF localization along with the calibration matrices directly from the raw data.

On localization. With current localization technology, we can reliably achieve accurate localization (e.g., *1-2 cm-level* accuracy with RTK[4], *10 cm-level* with Monte Carlo Localization scheme [12] as adopted in the nuScenes dataset [6]). We assume good localization in the training set.

Evaluation metric. We follow KITTI [22] to evaluate object detection in the bird's-eye view (BEV) and in 3D for the mobile objects. We report average precision (AP) with the intersection over union (IoU) thresholds at 0.5/0.25, which are used to evaluate cyclists and pedestrians objects in KITTI. We further follow [58] to evaluate the AP at various depth ranges. Due to space constraints, we only present evaluation results with IoU=0.25 in Tables 1 to 6. Please refer to the supplementary for results with IoU=0.5.

Implementation. We present results on PointRCNN [49] (the conclusions hold for other detectors such as PointPillars [31], and VoxelNet (SECOND) [60, 70]. See more details in the supplementary materials). For reproducibility, we use the publicly available code from OpenPCDet [52] for all models. We use the default hyperparameters tuned for KITTI except on the Lyft dataset in which we enlarge the perception range from 70m to 90m (since Lyft provides labels beyond 70m) and reduce the number of training epochs by 1/4 (since the training set is about three times of the size of KITTI). We default to 10 rounds of self-training (chosen arbitrarily due to compute constraints) and trained the model from scratch for each round of self-training. We also include results on PointRCNN trained up to 40 rounds, where we empirically observe that the performance con-

[4]https://en.wikipedia.org/wiki/Real-time_kinematic_positioning

Table 1. **Detection performance with different methods on the Lyft dataset.** We report AP_{BEV}/ AP_{3D} with IoU=0.25 for mobile objects under various ranges. Ri stands for i-th round self-training (R0 is training from seed labels). We also report the performance of detectors trained with ground-truth labels on the KITTI and the Lyft datasets at the last two rows.

Method	AP_{BEV}/ AP_{3D} @ IoU = 0.25			
	0-30	30-50	50-80	0-80
MODEST-PP (R0)	46.4 / 45.4	16.5 / 10.8	0.9 / 0.4	21.8 / 18.0
MODEST-PP (R10)	49.9 / 49.3	32.3 / 27.0	3.5 / 1.4	30.9 / 27.3
MODEST (R0)	65.7 / 63.0	41.4 / 36.0	8.9 / 5.7	42.5 / 37.9
MODEST (R10)	73.8 / 71.3	62.8 / 60.3	27.0 / 24.8	57.3 / 55.1
MODEST (R40)	76.4 / 74.1	64.2 / 62.9	47.1 / 45.5	64.4 / 62.7
Sup. (KITTI)	79.3 / 78.9	57.2 / 56.6	30.8 / 29.8	58.6 / 57.3
Sup. (Lyft)	82.8 / 82.6	70.8 / 70.3	50.2 / 49.6	69.5 / 69.1

Table 2. Detection results on the nuScenes Dataset. We report AP_{BEV}/ AP_{3D} at IoU=0.25 for mobile objects under various ranges. Please refer to Table 1 for naming.

Method	AP_{BEV}/ AP_{3D} @IoU = 0.25			
	0-30	30-50	50-80	0-80
MODEST-PP(R0)	0.7 / 0.1	0.0 / 0.0	0.0 / 0.0	0.2 / 0.1
MODEST-PP(R10)	-	-	-	-
MODEST (R0)	16.5 / 12.5	1.3 / 0.8	0.3 / 0.1	7.0 / 5.0
MODEST (R10)	24.8 / 17.1	5.5 / 1.4	1.5 / 0.3	11.8 / 6.6
Sup. (nuScenes)	39.8 / 34.5	12.9 / 10.0	4.4 / 2.9	22.2 / 18.2

verges (Figure 5). All models are trained with 4 NVIDIA 3090 GPUs. Please refer to the supplementary materials for full hyperparameters.

Baselines and ablations. We are the first work to train object detectors without any labels at all, and as such no previously published baselines exist. We create baselines by ablating the two key components of our model: seed labels generation via multiple traversals and repeated self-training:

- MODEST-PP (R0): This is a detector trained with seed labels generated *without* leveraging the multiple traversals. The seed labels are constructed by the exact same process as described in section 3, except we replace the edge weights in Equation 4 by spatial proximity: $w(e_{\boldsymbol{q},\boldsymbol{p}}) = \|\boldsymbol{q} - \boldsymbol{p}\|_2$ and change ϵ to 1.0 in DBSCAN, and do not perform any PP-score-based filtering on the clusters generated by DBSCAN. No repeated self-training is performed for this baseline.
- MODEST-PP (Ri): This detector is trained similarly to the previous baseline except we repeat i rounds of self-training witout using PP-score-based filtering.
- MODEST (R0): This detector is trained with the seed labels without repeated self-training.

4.1. Detecting mobile objects without annotations

We present results on Lyft in Table 1 and observe that:

1. **Object detectors can be trained using unlabeled data**: We observe that our approach yields accurate detectors, especially for the 0-50m range. For this range, MODEST is competitive with the fully supervised model trained on Lyft, and in fact *outperforms* a model trained with ground-truth supervision on KITTI. This suggests that MODEST is especially useful for bootstrapping recognition models in new domains.
2. **Our initial seed labels suffices to train a detector**: Detectors learned from our initial seed (MODEST (R0)) achieve more than 50% of supervised performance for nearby objects, suggesting that our common sense cues do produce a good initial training set. We investigate the quality of the seed labels in Table 4.
3. **Repeated self-training significantly improves performance.** We observe that if the seed labels can provide enough signals for training a decent detector, self-training can further drastically boost the performance, for example, by more than 500% from 8.9 to 47.1 on AP_{BEV} IoU=0.25 on 50-80m range.
4. **Ephemerality is a strong training signal.** We observe MODEST unanimously outperform MODEST-PP by a significant margin.

Table 3. Detection performance on the KITTI validation set with models trained on the Lyft dataset. We report AP_{BEV} / AP_{3D} with IoU=0.25 for mobile objects under various ranges. Please refer to Table 1 for naming.

Method	AP_{BEV} / AP_{3D} @ IoU = 0.25			
	0-30	30-50	50-80	0-80
MODEST-PP (R10)	56.6 / 55.0	22.7 / 18.1	0.9 / 0.7	42.7 / 40.7
MODEST (R10)	73.5 / 71.6	50.3 / 48.3	9.6 / 8.1	61.7 / 59.7
MODEST (R40)	73.6 / 73.2	49.9 / 48.2	15.1 / 13.9	63.0 / 61.1
Sup. (Lyft)	82.0 / 81.9	53.4 / 51.8	24.9 / 22.2	71.3 / 69.6
Sup. (KITTI)	88.0 / 87.9	73.6 / 72.0	46.7 / 45.4	81.1 / 81.0

We note that this performance of our detectors is especially good considering that we are evaluating it on predicting the *full amodal extent of the bounding box*, even though it has only seen the visible extent of the objects in the data. We notice that because of this discrepancy, our model produces smaller boxes; a size adjustment might well substantially improve accuracy for higher overlap thresholds.

We apply our approach in nuScenes dataset without changing the hyperparameters and report the results in Table 2. The above conclusions still hold. Notice that when we remove ephemerality (MODEST-PP(R10)), we are not able to extract enough signals to train a decent detector on nuScenes where LiDAR is much sparser than Lyft.

Cross-domain evaluation. It is possible that our automatic labeling process and multiple rounds of self-training overfit to biases in the training domain. To see if this is the case we test whether our models trained on unlabeled data from Lyft generalize to KITTI (Table 3). We observe that our detectors are still just as competitive with supervised detectors especially on close to middle ranges.

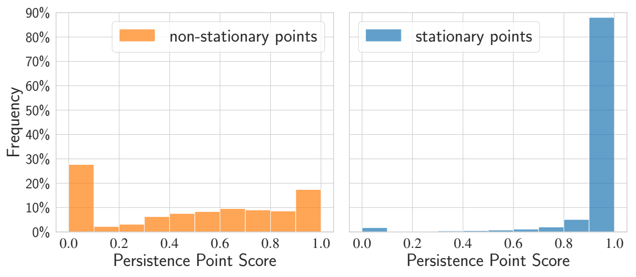
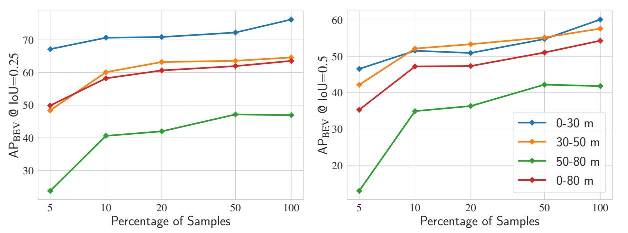


Figure 3. **Histogram of PP score for non-stationary and stationary points.** We separate the non-stationary and stationary points by the ground-truth object labels, and compute corresponding PP score frequency in the Lyft train set.



(a) IoU=0.25 (b) IoU=0.5

Figure 4. **Number of “train” samples *vs*. the performance.** We report AP_{BEV} with IoU=0.5 and IoU=0.25 for mobile objects in various ranges on Lyft test set from models trained with different mount of unlabeled data with 40 rounds of self-training.

4.2. Analysis

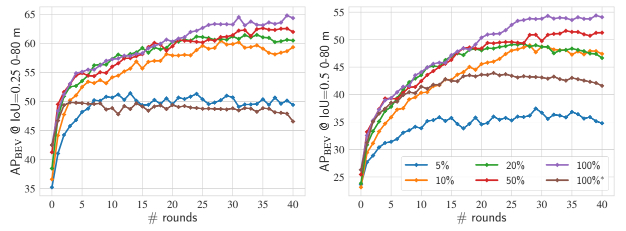
Given the good performance of our detectors, we dig deeper into the individual components to identify the key contributors to success.

Analysis on the PP score. In MODEST, PP score plays a critical role of distinguishing stationary points from stationary, background points. As such, in Figure 3, we plot the histogram of PP score for non-stationary and stationary points in the train split of the Lyft dataset. Non-stationary points are defined as the points within labeled bounding boxes for mobile objects, while background points are the points outside of these bounding boxes. The histogram clearly shows that for background points the PP score highly concentrates around 1, while for non-stationary points the score is much lower.

Effect of different amount of unlabeled data. Customary to any unsupervised learning algorithm, we investigate how different amount of unlabeled data affects our algorithm. We randomly subsample our training set and report the performance of MODEST in Figure 4. Across all ranges, we observe a general upward trend with more unlabeled data available. These results suggest that MODEST only needs a small amount of data to identify the rough location of mobile objects (evaluation at lower IoU is more lenient towards localization errors) and can significantly improve localization with more data.

Table 4. **The precision and recall of the “labels” on the Lyft dataset “train” split.** We report the ***precision / recall*** rate with BEV IoU=0.25 for mobile objects under various ranges. Please refer to Table 1 for naming.

Method	Precision / Recall @ IoU = 0.25			
	0-30	30-50	50-80	0-80
MODEST-PP (seed)	56.3 / 63.8	21.7 / 41.1	8.6 / 9.8	27.8 / 38.6
MODEST (seed)	73.9 / 57.7	55.8 / 37.3	41.3 / 11.5	62.7 / 35.7
MODEST (R0)	91.5 / 64.5	77.0 / 51.0	55.1 / 17.6	80.2 / 44.7
MODEST (R10)	92.4 / 71.2	83.7 / 69.1	58.4 / 42.8	79.5 / 61.7
MODEST (R40)	91.4 / 72.9	84.8 / 73.2	81.4 / 65.2	86.7 / 71.1



(a) IoU=0.25 (b) IoU=0.5

Figure 5. **Number of self-training rounds *vs*. the performance.** We report AP_{BEV} with IoU=0.5 and IoU=0.25 for mobile objects in 0-80 m on Lyft test set from models trained with different rounds of self-training. We report results from models trained with different amount of unlabeled data. Note that the $100\%^*$ line is self-training without PP score filtering.

Quality of training labels. The quality of the detector is determined by the quality of the automatically generated training labels. We evaluate the generated pseudo-ground truth boxes in Table 4 by computing their *precision* and *recall* compared to the ground truth. For seed labels, compared with MODEST-PP, PP score yields a set of labels with much higher precision but lower recall due to the filtering process. This is in line with our intuition that these seed boxes are conservative but high quality. After one round of training, the generated boxes have higher recall and precision. Subsequent rounds of self-training substantially improves recall especially on the far range, affirming our intuition that the neural network slowly identifies missed objects that are consistent with the conservative training sets. Put together, the whole process boosts the overall precision of seed labels by almost 40% and nearly doubles overall recall. This improved training data is reflected in the improvement in detector performance with self-training.

Effect of different rounds of self-training. Given that self-training substantially improves the quality of the training labels, we next look at how this impacts the detector. In Figure 5, we show how AP_{BEV} changes with different rounds of self-training. We observe that performance can improve for up to *40 rounds* of self-training with larger amount of data ($\geq 50\%$). Although more rounds can improve the performance, we emphasize that the improvement brought by PP score filtering cannot be compensated by additional rounds of self-training (the $100\%^*$ line).

Table 5. **Max recall with different methods on the Lyft dataset.** We report the max recall rate with BEV IoU= 0.25 for mobile objects within 0-80m. Please refer to Table 1 for naming. Please refer to the supplementary for results on different depth ranges.

Method	Car	Truck	Ped.	Cyc.
MODEST-PP (R0)	43.7	19.5	4.9	28.1
MODEST-PP (R10)	52.4	35.1	0.5	13.2
MODEST (R0)	54.6	31.8	10.2	50.5
MODEST (R10)	72.6	46.8	42.5	60.7
MODEST (R40)	81.4	46.4	42.1	62.9
Sup. (KITTI)	73.6	49.4	33.5	56.7
Sup. (Lyft)	83.6	65.4	53.8	67.5

Maximum achievable recall by object types. In Table 5, we further evaluate the maximum achievable recall of the different object types in the Lyft test set for various methods. We combine the *other_vehicle, truck, bus, emergency_vehicle* in the raw Lyft dataset into the *Truck* type and the *motorcycle, bicycle* into the *Cyclist* type. It can be seen that MODEST detects not only dominant, large objects in the dataset (Car), but also less common, smaller objects (Pedestrian and Cyclist).

Common sense *vs*. self-training. Clearly both our common sense-based seed boxes and our self-training approach are crucial for detector accuracy. But how do they stack against each other? In Table 6, we attempt to trade-off the seed labels *vs*. the self-training by varying the number of scenes available to each step. We also experiment with switching off common sense-based (i.e., PP score-based) filtering during self-training. We observe that increasing the number of scenes for self-training has a bigger impact than increasing the size of the seed set (row 1 *vs*. rows 3 and 6). Interestingly, without PP score-based filtering, using 100% data for seed labels performs worse than using only 5% of the scenes. This may be because if all scenes are used for seed label computation and then used for training the detector, the detector may over-fit to the quirks of these labels and may not be able to correct them during self-training. Having a "held-out" set of scenes for self-training thus seems beneficial. We also observe that PP score-based filtering does have a big impact on self-training and improves performance significantly (row 1 *vs*. 2, 4 *vs*. 5, 6 *vs*. 7). Thus common sense-based filtering is crucial even within the self-training pipeline, suggesting a synergy between common sense and neural net training.

Qualitative Results. We show qualitative results of "seed" label generation in Figure 2, and visualization of self training on two scenes in Figure 1 on the "train" split of the Lyft dataset. Observe that the seed label generation filters out many of the superfluous clusters, but occasionally misses some objects or produces incorrectly sized objects. Via bootstrapping an object detector, our method can gradually recover the shape of mobile objects, as well as obtain higher recall than the initial seed label set (Figure 1). Please refer to supplementary material for more.

Table 6. **Common sense vs self-training.** We report AP_{BEV} / AP_{3D} with IoU=0.25 for mobile objects under various ranges. Seed and ST column mean how much data are used as seed data and self-training data respectively; FT stands for filtering by PP score during self-training. All are with 10 rounds of self-training.

Combinations			AP_{BEV} / AP_{3D} @ IoU = 0.25			
Seed	ST	FT	0-30	30-50	50-80	0-80
5%	5%		54.8 / 53.2	40.4 / 39.4	17.3 / 16.3	39.4 / 37.6
5%	5%	✓	70.3 / 68.0	51.5 / 49.1	22.4 / 16.7	50.7 / 46.9
5%	100%		68.8 / 67.3	55.6 / 54.8	19.6 / 17.4	51.0 / 49.4
100%	5%		55.4 / 54.1	41.8 / 41.4	23.3 / 22.3	41.0 / 40.4
100%	5%	✓	68.0 / 65.6	49.0 / 47.4	28.7 / 25.3	51.1 / 47.9
100%	100%		68.5 / 68.1	52.9 / 51.9	17.3 / 16.4	48.9 / 47.8
100%	100%	✓	73.8 / 71.3	62.8 / 60.3	27.0 / 24.8	57.3 / 55.1

5. Discussion

Limitation. Our approach focuses on learning to detect mobile objects and is evaluated by IoU between detections and ground-truth bounding boxes with single object type. We do not take the heading of objects into account, nor do we classify different object types. Also as mentioned above, our model tends to produce smaller boxes rather than amodal boxes. We leave these as future work.

Conclusion. In this work, we explore a novel problem of learning a 3D mobile object detector from LiDAR scans without any labels. Though this seems impossible at the first glance, we propose MODEST and show that with simple heuristics about mobile objects, e.g., they are not persistent over time, we can generate weak labels and bootstrap a surprisingly accurate detector from them. We evaluate MODEST exhaustively on two large-scale, real-world datasets and draw consistent conclusions. We consider our work a first step towards a larger research effort to make entirely unsupervised object detectors a highly competitive reality. The potential impact of such an achievement would be monumental, allowing cars to be re-trained while in use, adapting to their local environments, enabling reliable driver assist and self-driving vehicles in developing countries, and avoiding privacy concerns by training vehicles on their own locally gathered data. We hope our work will inspire more research into this new and highly relevant problem.

Acknowledgements This research is supported by grants from the National Science Foundation NSF (III-1618134, III-1526012, IIS-1149882, IIS-1724282, TRIPODS-1740822, IIS-2107077, OAC-2118240, OAC-2112606 and IIS-2107161), the Office of Naval Research DOD (N00014-17-1-2175), the DARPA Learning with Less Labels program (HR001118S0044), the Bill and Melinda Gates Foundation, the Cornell Center for Materials Research with funding from the NSF MRSEC program (DMR-1719875), and SAP America.

References

[1] Wim Abbeloos, Esra Ataer-Cansizoglu, Sergio Caccamo, Yuichi Taguchi, and Yukiyasu Domae. 3d object discovery and modeling using single rgb-d images containing multiple object instances. In *2017 International Conference on 3D Vision (3DV)*, pages 431–439. IEEE, 2017. 3

[2] Devansh Arpit, Stanisław Jastrzębski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. In *ICML*, 2017. 3

[3] Dan Barnes, Will Maddern, Geoffrey Pascoe, and Ingmar Posner. Driven to distraction: Self-supervised distractor learning for robust monocular visual odometry in urban environments. In *ICRA*, pages 1894–1900. IEEE, 2018. 1, 2, 3

[4] David Berthelot, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. *Proceedings of the International Conference on Learning Representations*, 2020. 3

[5] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. *Advances in Neural Information Processing Systems*, 32, 2019. 3

[6] 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*, pages 11621–11631, 2020. 2, 5

[7] Ming-Fang Chang, John W Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, and James Hays. Argoverse: 3d tracking and forecasting with rich maps. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[8] Chaoqi Chen, Weiping Xie, Wenbing Huang, Yu Rong, Xinghao Ding, Yue Huang, Tingyang Xu, and Junzhou Huang. Progressive feature alignment for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 627–636, 2019. 3

[9] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3d object proposals using stereo imagery for accurate object class detection. *IEEE transactions on pattern analysis and machine intelligence*, 40(5):1259–1272, 2017. 2

[10] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3D object detection network for autonomous driving. In *CVPR*, 2017. 2

[11] Minsu Cho, Suha Kwak, Cordelia Schmid, and Jean Ponce. Unsupervised object discovery and localization in the wild: Part-based matching with bottom-up region proposals. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 2, 3

[12] Zhuang Jie Chong, Baoxing Qin, Tirthankar Bandyopadhyay, Marcelo H Ang, Emilio Frazzoli, and Daniela Rus. Synthetic 2d lidar for precise vehicle localization in 3d urban environment. In *ICRA*, pages 1554–1559. IEEE, 2013. 5

[13] Siddharth Choudhary, Alexander JB Trevor, Henrik I Christensen, and Frank Dellaert. Slam with object discovery, modeling and mapping. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1018–1025. IEEE, 2014. 2

[14] Alvaro Collet, Siddhartha S Srinivasay, and Martial Hebert. Structure discovery in multi-modal data: a region-based approach. In *2011 IEEE International Conference on Robotics and Automation*, pages 5695–5702. IEEE, 2011. 2

[15] Alvaro Collet, Bo Xiong, Corina Gurau, Martial Hebert, and Siddhartha S Srinivasa. Exploiting domain knowledge for object discovery. In *2013 IEEE International Conference on Robotics and Automation*, pages 2118–2125. IEEE, 2013. 2

[16] Alvaro Collet, Bo Xiong, Corina Gurau, Martial Hebert, and Siddhartha S Srinivasa. Herbdisc: Towards lifelong robotic object discovery. *The International Journal of Robotics Research*, 34(1):3–25, 2015. 2

[17] Yilun Du, Kevin Smith, Tomer Ulman, Joshua Tenenbaum, and Jiajun Wu. Unsupervised discovery of 3d physical objects from video. *arXiv preprint arXiv:2007.12348*, 2020. 2

[18] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996. 4

[19] Alon Faktor and Michal Irani. Co-segmentation by composition. In *Proceedings of the IEEE international conference on computer vision*, pages 1297–1304, 2013. 3

[20] Germán M García, Ekaterina Potapova, Thomas Werner, Michael Zillich, Markus Vincze, and Simone Frintrop. Saliency-based object discovery on rgb-d data with a late-fusion approach. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1866–1873. IEEE, 2015. 2

[21] 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. 1, 2, 5

[22] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012. 1, 2, 5

[23] Golnaz Ghiasi, Barret Zoph, Ekin D. Cubuk, Quoc V. Le, and Tsung-Yi Lin. Multi-task self-training for learning general representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 8856–8865, October 2021. 3

[24] Yves Grandvalet, Yoshua Bengio, et al. Semi-supervised learning by entropy minimization. *CAP*, 367:281–296, 2005. 3

[25] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *NeurIPS*, 2018. 3

[26] Evan Herbst, Peter Henry, Xiaofeng Ren, and Dieter Fox. Toward object discovery and modeling via 3-d scene comparison. In *2011 IEEE International Conference on Robotics and Automation*, pages 2623–2629. IEEE, 2011. 2

[27] Evan Herbst, Xiaofeng Ren, and Dieter Fox. Rgb-d object discovery via multi-scene analysis. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4850–4856. IEEE, 2011. 2

[28] Andrej Karpathy, Stephen Miller, and Li Fei-Fei. Object discovery in 3d scenes via shape analysis. In *2013 IEEE International Conference on Robotics and Automation*, pages 2088–2095. IEEE, 2013. 2

[29] R. Kesten, M. Usman, J. Houston, T. Pandya, K. Nadhamuni, A. Ferreira, M. Yuan, B. Low, A. Jain, P. Ondruska, S. Omari, S. Shah, A. Kulkarni, A. Kazakova, C. Tao, L. Platinsky, W. Jiang, and V. Shet. Lyft level 5 av dataset 2019. https://level5.lyft.com/dataset/, 2019. 2, 5

[30] Deyvid Kochanov, Aljoša Ošep, Jörg Stückler, and Bastian Leibe. Scene flow propagation for semantic mapping and object discovery in dynamic street scenes. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1785–1792. IEEE, 2016. 2

[31] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. Pointpillars: Fast encoders for object detection clouds. In *ICCV*, pages 12697–12705, 2019. 2, 4, 5, 1

[32] Dong-Hyun Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 896, 2013. 2, 3, 5

[33] Weihao Li, Omid Hosseini Jafari, and Carsten Rother. Deep object co-segmentation. In *Asian Conference on Computer Vision*, pages 638–653. Springer, 2018. 3

[34] Ming Liang, Bin Yang, Rui Hu, Yun Chen, Renjie Liao, Song Feng, and Raquel Urtasun. Learning lane graph representations for motion forecasting, 2020. 2

[35] Lu Ma, Mahsa Ghafarianzadeh, David Coleman, Nikolaus Correll, and Gabe Sibley. Simultaneous localization, mapping, and manipulation for unsupervised object discovery. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1344–1351. IEEE, 2015. 2

[36] Lu Ma and Gabe Sibley. Unsupervised dense object discovery, detection, tracking and reconstruction. In *European Conference on Computer Vision*, pages 80–95. Springer, 2014. 2

[37] Julian Mason, Bhaskara Marthi, and Ronald Parr. Object disappearance for object discovery. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2836–2843. IEEE, 2012. 2

[38] Julian Mason, Bhaskara Marthi, and Ronald Parr. Unsupervised discovery of object classes with a mobile robot. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3074–3081. IEEE, 2014. 2

[39] Ke Mei, Chuang Zhu, Jiaqi Zou, and Shanghang Zhang. Instance adaptive self-training for unsupervised domain adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 3

[40] Tai-Yu Pan, Cheng Zhang, Yandong Li, Hexiang Hu, Dong Xuan, Soravit Changpinyo, Boqing Gong, and Wei-Lun Chao. On model calibration for long-tailed object detection and instance segmentation. In *NeurIPS*, 2021. 5

[41] Deepak Pathak, Ross Girshick, Piotr Dollár, Trevor Darrell, and Bharath Hariharan. Learning features by watching objects move. In *CVPR*, 2017. 3, 5

[42] Cheng Perng Phoo and Bharath Hariharan. Coarsely-labeled data for better few-shot transfer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9052–9061, October 2021. 3

[43] Cheng Perng Phoo and Bharath Hariharan. Self-training for few-shot transfer across extreme task differences. In *Proceedings of the International Conference on Learning Representations*, 2021. 3

[44] Geoff Pleiss, Tianyi Zhang, Ethan R Elenberg, and Kilian Q Weinberger. Identifying mislabeled data using the area under the margin ranking. In *NeurIPS*, 2020. 3

[45] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *CVPR*, 2018. 1, 2

[46] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017. 2

[47] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, 2017. 2

[48] 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 *CVPR*, pages 10529–10538, 2020. 1, 2

[49] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *CVPR*, 2019. 1, 2, 4, 5

[50] Jiwon Shin, Rudolph Triebel, and Roland Siegwart. Unsupervised discovery of repetitive objects. In *2010 IEEE International Conference on Robotics and Automation*, pages 5041–5046. IEEE, 2010. 2, 3

[51] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in Neural Information Processing Systems*, 2020. 3

[52] OpenPCDet Development Team. Openpcdet: An open-source toolbox for 3d object detection clouds. https://github.com/open-mmlab/OpenPCDet, 2020. 5

[53] Hao Tian, Yuntao Chen, Jifeng Dai, Zhaoxiang Zhang, and Xizhou Zhu. Unsupervised object detection with lidar clues. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5962–5972, 2021. 2

[54] Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B Tenenbaum, and Phillip Isola. Rethinking few-shot image classification: a good embedding is all you need? In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*, pages 266–282. Springer, 2020. 3

[55] Rudolph Triebel, Jiwon Shin, and Roland Siegwart. Segmentation and unsupervised part-based discovery of repetitive objects. *Robotics: Science and Systems VI*, pages 1–8, 2010. 2, 3

[56] Huy V Vo, Francis Bach, Minsu Cho, Kai Han, Yann LeCun, Patrick Pérez, and Jean Ponce. Unsupervised image matching and object discovery as optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8287–8296, 2019. 2, 3

[57] Huy V Vo, Patrick Pérez, and Jean Ponce. Toward unsupervised, multi-object discovery in large-scale image collections. In *European Conference on Computer Vision*, pages 779–795. Springer, 2020. 3

[58] Yan Wang, Xiangyu Chen, Yurong You, Li Erran Li, Bharath Hariharan, Mark Campbell, Kilian Q. Weinberger, and Wei-Lun Chao. Train in germany, test in the usa: Making 3d object detectors generalize. In *CVPR*, pages 11713–11723, June 2020. 1, 2, 5

[59] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10687–10698, 2020. 2, 3, 5

[60] Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10):3337, 2018. 2, 4, 5, 1

[61] Bin Yang, Wenjie Luo, and Raquel Urtasun. Pixor: Real-time 3d object detection from point clouds. In *CVPR*, 2018. 1, 2

[62] Jihan Yang, Shaoshuai Shi, Zhe Wang, Hongsheng Li, and Xiaojuan Qi. St3d: Self-training for unsupervised domain adaptation on 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021. 3

[63] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. 3dssd: Point-based 3d single stage object detector. In *CVPR*, pages 11040–11048, 2020. 2

[64] Ze-Huan Yuan, Tong Lu, Yirui Wu, et al. Deep-dense conditional random fields for object co-segmentation. In *IJCAI*, pages 3371–3377, 2017. 3

[65] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3):107–115, 2021. 5

[66] Quanshi Zhang, Xuan Song, Xiaowei Shao, Huijing Zhao, and Ryosuke Shibasaki. Unsupervised 3d category discovery and point labeling from a large urban environment. In *2013 IEEE International Conference on Robotics and Automation*, pages 2685–2692. IEEE, 2013. 3

[67] Qiming Zhang, Jing Zhang, Wei Liu, and Dacheng Tao. Category anchor-guided unsupervised domain adaptation for semantic segmentation. In *Advances in neural information processing systems*, 2019. 3

[68] Xiao Zhang, Wenda Xu, Chiyu Dong, and John M Dolan. Efficient l-shape fitting for vehicle detection using laser scanners. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 54–59. IEEE, 2017. 4

[69] Yin Zhou, Pei Sun, Yu Zhang, Dragomir Anguelov, Jiyang Gao, Tom Ouyang, James Guo, Jiquan Ngiam, and Vijay Vasudevan. End-to-end multi-view fusion for 3D object detection in lidar point clouds. In *CoRL*, 2020. 2

[70] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *CVPR*, 2018. 2, 4, 5, 1

[71] Yang Zou, Zhiding Yu, BVK Kumar, and Jinsong Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *Proceedings of the European conference on computer vision (ECCV)*, pages 289–305, 2018. 3

[72] Yang Zou, Zhiding Yu, Xiaofeng Liu, BVK Kumar, and Jinsong Wang. Confidence regularized self-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5982–5991, 2019. 3

Supplementary Material

S1. Implementation details

We set $[-H_s, H_e]$ to $[0, 70]$ m since we experiment with frontal-view detection only. We combine only one scan into the dense point cloud $\boldsymbol{S}_c^t$ every 2 m within this range. In calculating PP score, we use as many traversals as possible (≥ 2) and set $r = 0.3$m. For clustering, we use $K = 70$ and $r' = 2.0$m in the graph, and $\epsilon = 0.1$, min_samples $= 10$ for DBSCAN. For filtering, we use a loose threshold of $\alpha = 20$ percentile and $\gamma = 0.7$. Other common sense properties are simply implemented as follows:

- $\#$ points in the cluster $>= 10$;
- Volume of fitted bounding boxes $\in [0.5, 120]m^3$;
- The height (upright distance against the ground plane) of points $\text{Height}_{\text{max}} > 0.5m$ and $\text{Height}_{\text{min}} < 1.0m$ to ensure clusters not floating in the air or beneath the ground due to errors in LiDAR.

We did not tune these parameters except qualitatively checked the fitted bounding boxes in few scenes in the Lyft "train" set. For detection models, we use the default hyper-parameters tuned on KITTI[5] with few exceptions listed in the paper. We will open-source the code upon acceptance.

S2. Experiments with other detectors

Besides the PointRCNN detector [49], We experiment with two other detectors PointPillars [31] and VoxelNet (SECOND) [60, 70], and show their results in Table S2 and Table S1. We apply the default hyper-parameters of these two models tuned on KITTI, and apply the same procedure as that on PointRCNN models. Note that PointPillars and VoxelNet model need a pre-defined anchor size for different types of objects, which we picked (length, width, height) as $(2.0, 1.0, 1.7)$ m without tuning. We observe that generally the PointPillars and VoxelNet yield worse results than PointRCNN models (possibly due to the fixed anchor size for all mobile objects), but we still observe significant gains from self-training.

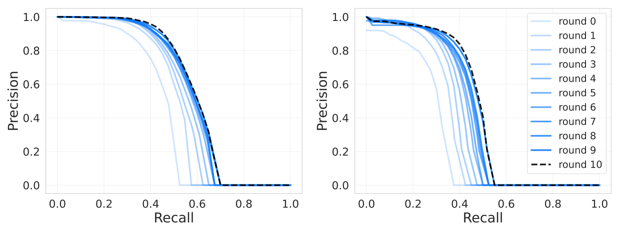
S3. Detailed evaluation by object types

In Table S3, we include detailed evaluations (BEV IoU$=$ 0.5, BEV IoU$=$ 0.25 and by different depth ranges) of the recall of different object types in the Lyft test set. This corresponds to Table 5 in the main paper.

S4. Corresponding IoU=0.5 results

We list the IoU=0.5 counterparts of Tables 1 to 6 in Tables S4 to S8.

[5]https://github.com/open-mmlab/OpenPCDet/tree/master/tools/cfgs/kitti_models



(a) IoU=0.25 (b) IoU=0.5

Figure S1. **Number of self-training rounds *vs.* precision-recall curves.** We show the precision-recall curves under AP_{BEV} with IoU=0.5 and IoU=0.25 for mobile objects in 0-80 m on Lyft test set from models trained with different rounds of self-training.

S5. Precision-recall evaluation

In Figure S1, we show how PR curve changes with different rounds of self-training: the max recall improves gradually while keeping high precision. This aligns with the expanded recall of the training set described above, and with what we observe qualitatively in Figure 1.

S6. More qualitative results

We show visualizations for additional qualitative results in Figure S2 for 5 additional LiDAR scenes. Visualizations show the progression of MODEST from seed generation, to detector trained on seed label set, to detector after 10 rounds of self training, and finally the ground truth bounding boxes. Observe that the detections obtain higher recall and learns a correct prior over object shapes as the method progresses.

Table S1. **Detection performance with PointPillars [31] on the Lyft dataset.** Please refer to Table 1 for naming.

Method	AP_{BEV}/ AP_{3D} @ IoU = 0.25				AP_{BEV}/ AP_{3D} @ IoU = 0.5			
	0-30	30-50	50-80	0-80	0-30	30-50	50-80	0-80
MODEST (R0)	56.3 / 51.3	26.6 / 19.5	5.4 / 3.0	30.4 / 24.6	32.1 / 25.2	10.0 / 4.2	1.2 / 0.2	13.8 / 8.5
MODEST (R10)	55.7 / 49.1	43.1 / 38.4	8.8 / 7.5	33.9 / 29.9	37.4 / 27.7	28.8 / 10.0	5.0 / 1.1	22.1 / 10.7
Sup. (Lyft)	78.7 / 77.9	64.6 / 63.7	45.4 / 44.1	64.7 / 63.6	72.9 / 68.9	55.5 / 50.3	41.5 / 35.4	58.0 / 52.8

Table S2. **Detection performance with VoxelNet (SECOND) [60, 70] on the Lyft dataset.** Please refer to Table 1 for naming.

Method	AP_{BEV}/ AP_{3D} @ IoU = 0.25				AP_{BEV}/ AP_{3D} @ IoU = 0.5			
	0-30	30-50	50-80	0-80	0-30	30-50	50-80	0-80
MODEST (R0)	54.3 / 49.7	27.8 / 21.4	4.9 / 2.8	30.2 / 24.8	30.3 / 24.9	11.7 / 5.1	1.1 / 0.2	14.0 / 8.8
MODEST (R10)	54.9 / 44.8	38.7 / 31.5	8.3 / 6.2	32.5 / 26.0	32.1 / 24.3	20.0 / 7.7	3.4 / 0.8	17.0 / 8.9
Sup. (Lyft)	81.6 / 81.1	67.8 / 66.3	45.5 / 44.6	65.9 / 64.9	76.7 / 73.9	59.7 / 55.3	41.8 / 36.3	60.1 / 55.4

Table S3. **Max recall with different methods on the Lyft dataset.** Please refer to Table 1 for naming. This corresponds to the counterpart Table 5 in the main paper.

(a) Recall @ IoU=0.5

Method	Car				Truck				Pedestrian				Cyclist			
	0-30	30-50	50-80	0-80	0-30	30-50	50-80	0-80	0-30	30-50	50-80	0-80	0-30	30-50	50-80	0-80
MODEST-PP (R0)	57.6	27.3	3.0	30.1	36.1	5.0	0.2	9.1	0.9	1.1	0.8	1.0	14.4	10.7	2.1	10.7
MODEST-PP (R10)	63.2	49.1	8.0	40.9	39.0	21.4	5.6	20.1	0.0	0.0	0.2	0.1	8.1	11.5	2.9	8.2
MODEST (R0)	63.7	45.5	11.2	41.0	29.8	18.3	2.4	17.2	5.2	0.6	0.0	1.7	34.1	11.7	1.2	20.2
MODEST (R10)	67.5	70.7	40.9	60.6	35.1	28.2	13.3	25.3	35.1	35.0	7.6	27.5	62.9	41.1	7.1	44.7
MODEST (R40)	69.9	78.8	68.9	72.9	25.4	27.4	20.1	28.0	39.4	38.8	6.2	28.6	70.6	32.5	1.5	44.4
Sup. (KITTI)	82.1	76.3	53.3	71.2	45.9	23.8	23.3	30.0	56.9	32.1	2.5	29.7	59.5	16.1	1.2	33.8
Sup. (Lyft)	85.7	82.5	75.2	81.5	64.9	52.0	50.3	54.7	60.8	55.4	18.9	45.2	71.0	43.2	6.8	49.2

(b) Recall @ IoU=0.25

Method	Car				Truck				Pedestrian				Cyclist			
	0-30	30-50	50-80	0-80	0-30	30-50	50-80	0-80	0-30	30-50	50-80	0-80	0-30	30-50	50-80	0-80
MODEST-PP (R0)	65.5	53.0	8.8	43.7	52.2	19.1	1.9	19.5	2.8	6.1	4.9	4.9	33.9	31.7	8.3	28.1
MODEST-PP (R10)	73.0	66.2	14.0	52.4	62.9	36.0	9.2	35.1	0.9	0.5	0.2	0.5	15.5	15.3	4.1	13.2
MODEST (R0)	73.9	63.6	22.9	54.6	46.8	30.5	12.2	31.8	22.5	8.6	1.5	10.2	75.3	42.3	4.4	50.5
MODEST (R10)	79.0	80.8	55.1	72.6	59.5	43.1	33.6	46.8	61.7	51.4	14.6	42.5	71.8	66.0	26.0	60.7
MODEST (R40)	81.2	83.1	78.1	81.4	53.2	43.3	31.7	46.4	63.9	51.8	11.0	42.1	81.6	62.0	19.8	62.9
Sup. (KITTI)	82.8	78.3	57.5	73.6	71.7	42.3	33.4	49.4	66.0	35.0	2.8	33.5	84.1	45.3	9.7	56.7
Sup. (Lyft)	86.9	84.1	78.5	83.6	73.7	66.3	58.0	65.4	72.3	63.7	25.2	53.8	83.1	68.1	29.2	67.5

Table S4. **Detection performance with different methods on the Lyft dataset.** We report AP_{BEV}/ AP_{3D} with IoU=0.5 for mobile objects under various ranges. Please refer to Table 1 for naming. This corresponds to the counterpart Table 1 in the main paper.

Method	AP_{BEV}/ AP_{3D} @ IoU = 0.25			
	0-30	30-50	50-80	0-80
MODEST-PP (R0)	34.1 / 31.3	5.1 / 3.0	0.0 / 0.0	12.0 / 9.7
MODEST-PP (R10)	42.1 / 38.3	21.9 / 19.2	1.0 / 0.9	22.8 / 20.6
MODEST (R0)	44.9 / 40.4	24.5 / 14.8	2.7 / 0.7	26.3 / 19.8
MODEST (R10)	56.8 / 51.3	51.4 / 40.5	19.2 / 9.0	44.1 / 35.5
MODEST (R40)	61.1 / 56.2	57.5 / 53.4	41.2 / 29.8	54.1 / 47.6
Sup. (KITTI)	72.3 / 69.5	53.2 / 48.1	27.9 / 20.5	53.1 / 48.1
Sup. (Lyft)	79.6 / 77.5	66.4 / 64.4	47.8 / 43.8	65.5 / 63.2

Table S5. **Detection results on the nuScenes Dataset.** We report AP_{BEV} / AP_{3D} at IoU=0.5 for mobile objects under various ranges. Please refer to Table 1 for naming. This corresponds to the counterpart Table 2 in the main paper.

Method	AP_{BEV} / AP_{3D} @IoU = 0.5			
	0-30	30-50	50-80	0-80
MODEST-PP(R0)	0.0 / 0.0	0.0 / 0.0	0.0 / 0.0	0.0 / 0.0
MODEST-PP(R10)	-	-	-	-
MODEST (R0)	8.4 / 2.9	0.3 / 0.1	0.1 / 0.0	3.0 / 0.9
MODEST (R10)	11.0 / 7.6	0.4 / 0.0	0.0 / 0.0	3.9 / 2.2
Sup. (nuScenes)	29.5 / 26.3	8.4 / 6.1	2.4 / 1.1	15.5 / 13.3

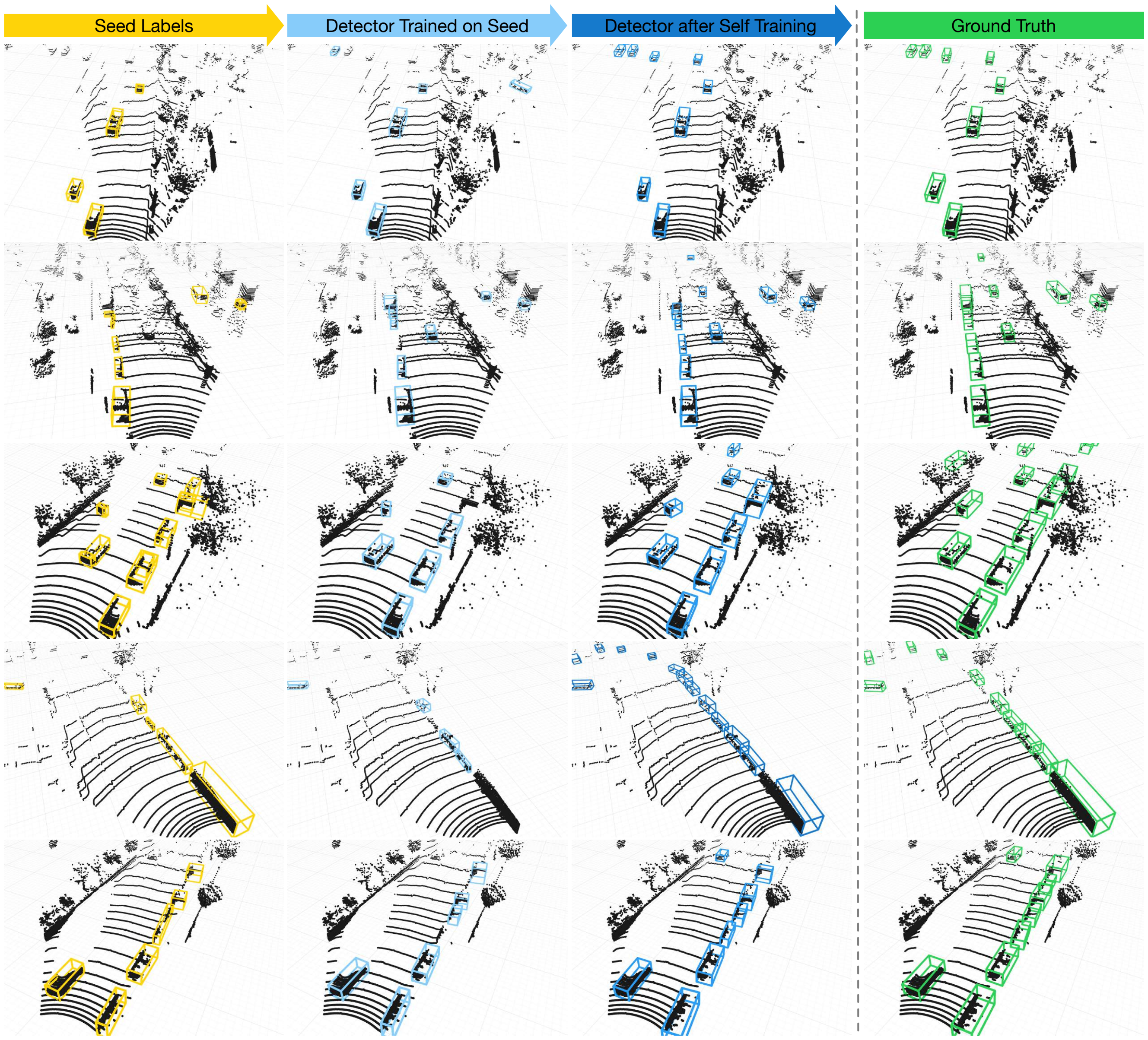


Figure S2. **Visualizations of MODEST outputs.** We show additional visualizations of LiDAR scans from three scenes in the Lyft dataset. From left to right: seed labels, detections trained on seed, detections after self training, and ground truth bounding boxes.

Table S6. **Detection performance on the KITTI validation set with models trained on the Lyft dataset.** We report AP$_{\text{BEV}}$ / AP$_{\text{3D}}$ with IoU=0.5 for mobile objects under various ranges. Please refer to Table 1 for naming. This corresponds to the counterpart Table 3 in the main paper.

Method	AP$_{\text{BEV}}$ / AP$_{\text{3D}}$ @ IoU = 0.5			
	0-30	30-50	50-80	0-80
MODEST-PP (R10)	50.5 / 48.5	10.3 / 8.7	0.2 / 0.1	35.6 / 33.8
MODEST (R10)	62.1 / 57.6	41.7 / 32.3	5.3 / 2.0	51.1 / 46.0
MODEST (R40)	57.7 / 52.7	44.1 / 40.1	11.6 / 7.2	49.3 / 44.6
Sup. (Lyft)	79.0 / 76.7	47.7 / 42.8	19.0 / 12.4	65.3 / 62.5
Sup. (KITTI)	85.4 / 83.3	69.5 / 66.9	41.2 / 35.0	78.3 / 76.0

Table S7. **The precision and recall of the "labels" on the Lyft dataset "train" split.** We report the ***precision / recall*** rate with BEV IoU=0.5 for mobile objects under various ranges. Please refer to Table 1 for naming. This corresponds to the counterpart Table 4 in the main paper.

Method	Precision / Recall @ IoU = 0.5			
	0-30	30-50	50-80	0-80
MODEST-PP (seed)	44.4 / 50.3	8.5 / 16.1	2.3 / 2.6	16.3 / 22.7
MODEST (seed)	55.8 / 43.5	28.8 / 19.3	15.6 / 4.4	38.9 / 22.2
MODEST (R0)	79.4 / 55.9	51.3 / 34.0	30.0 / 9.6	59.3 / 33.0
MODEST (R10)	82.2 / 63.4	65.9 / 55.0	38.7 / 28.9	62.9 / 49.3
MODEST (R40)	83.1 / 66.2	77.6 / 67.0	69.0 / 55.3	77.2 / 63.3

Table S8. **Common sense vs self-training.** We report AP_{BEV} / AP_{3D} with IoU=0.5 for mobile objects under various ranges. Seed and ST column mean how much data are used as seed data and self-training data respectively; FT stands for filtering by PP score during self-training. This corresponds to the counterpart Table 6 in the main paper.

Combinations			AP_{BEV} / AP_{3D} @ IoU = 0.5			
Seed	ST	FT	0-30	30-50	50-80	0-80
5%	5%		43.9 / 40.3	35.6 / 30.8	7.8 / 5.1	30.2 / 26.2
5%	5%	✓	47.7 / 40.0	40.8 / 37.7	10.1 / 8.6	33.9 / 29.8
5%	100%		55.2 / 51.1	46.4 / 38.0	13.8 / 8.4	40.2 / 34.2
100%	5%		44.6 / 40.4	38.7 / 33.0	16.3 / 7.2	34.4 / 28.0
100%	5%	✓	48.5 / 43.0	43.4 / 35.6	18.2 / 8.3	38.2 / 30.2
100%	100%		57.8 / 52.2	46.4 / 38.4	14.4 / 8.6	41.4 / 34.3
100%	100%	✓	56.8 / 51.3	51.4 / 40.5	19.2 / 9.0	44.1 / 35.5