Appendix of
LEOD: Label-Efficient Object Detection for Event Cameras

Ziyi Wu[1,2], Mathias Gehrig[3], Qing Lyu[1], Xudong Liu[1], Igor Gilitschenski[1,2]
[1]University of Toronto, [2]Vector Institute, [3] University of Zurich

Overview

We provide additional details and results to complement the main paper. This document includes the following materials:

- More implementation details of our method (Appendix A);
- Analysis of pseudo label quality under different settings (Appendix B);
- Visualizations of pseudo labels (Appendix C);
- Discussions on the naming of two experimental settings: WSOD vs. SSOD (Appendix D);
- Discussions on potential negative societal impacts (Appendix E).

A. More Implementation Details

A.1. Tracking-based Post-Processing

Given the detection outputs from TTA, we first aggregate them via Non-Maximum Suppression (NMS). Now, for each event frame I at timestep t, we have a set of 2D bounding boxes $\mathcal{B}^t = \{b_j^t = (x_j, y_j, w_j, h_j, l_j, t)\}$. We follow the tracking-by-detection paradigm [1] to build tracks by linking detection boxes between frames, which is also inspired by [13]. Each track $s_k = \{(b, v_x, v_y)_t, k, n, q\}$ maintains the following attributes: (v_x, v_y) is the estimated velocity in the pixel space, k is the track's unique ID, n is its length so far, and $q \in [0, 1]$ is its current score, which is decayed over time and determines whether to delete the track. In the first frame, we initialize each box in $\mathcal{B}^0$ as a track, where $(v_x, v_y) = (0, 0)$, $n = 1$, and $q = 0.9$. For every coming frame I_t, we need to link its bounding boxes $\mathcal{B}^t$ to existing tracks $\{s_k\}$. We first predict the new box parameter of each track using its coordinate in the last frame (x, y) and (v_x, v_y) with a linear motion assumption, while keeping its size in the last frame (w, h) unchanged. Then, we compute pairwise IoUs between the predicted boxes and $\mathcal{B}^t$ to apply greedy matching. Only boxes in the same category and with an IoU larger than τ_{iou} can be matched. For unmatched boxes, we initialize tracks for them as done in the first frame. For unmatched tracks, we decay its score as $q_t = 0.9 * q_{t-1}$, which allows for object re-identification in future frames. For matched boxes and tracks, we update the box parameters and velocity, and reset the score as $q = 1$. Finally, we go over each track and delete those with a lower score $q < \tau_{\text{del}}$. After tracking, each box is associated with a track, and thus a length n (note that n represents the number of successful matches instead of the time between creation and deletion, i.e., unmatched timesteps do not count). We identify boxes with $n < T_{trk}$ as temporally inconsistent.

Similar to TTA, we apply tracking in forward and backward event sequences, and will only remove a box if it has a short track length in both directions. For those long tracks, we inpaint boxes at their unmatched timesteps using the synthesized ones with linear motion. This builds upon the prior of object permanence and can further stabilize the training in our experiments. Overall, the detection losses related to removed and inpainted boxes will be ignored during model training. For hyper-parameters, we choose $\tau_{\text{iou}} = 0.45$ which is the same as the IoU threshold used in NMS, $\tau_{\text{del}} = 0.55$ which is slightly higher than $0.9^6 \approx 0.53$, and $T_{trk} = 6$. We do not tune these hyper-parameters and simply use the first set of values that works.

A.2. RVT Training

We build upon the open-source codebase of RVT[1] [4] and copy most of their training settings. Events in each $50ms$ time window are converted to a frame-like 10-channel event histogram representation. We use RVT-S in most of the experiments due to limited computation resources, but also scale up LEOD to RVT-B in Sec. 4.4. Following [4], we down-sample the labeling frequency of 1Mpx [8] to 10 Hz.

Pre-training on Sparse Labels. The same optimizer, batch size, data augmentation, and data sampling methods are used. In order to apply the time-flip TTA during pseudo-labeling, we add a temporal flipping augmentation. We train for 200k steps on 1% labels, 300k steps on 2% labels, and 400k steps on 5%, 10%, and 100% labels. On 1Mpx [8], we use an increased sequence length $L = 10$ for training, as we observed clearly better results compared to $L = 5$.

Pseudo Label Filtering. We filter out low-confidence bounding boxes to obtain high-quality pseudo labels. As introduced in Sec. 3.1, RVT predicts an objectness score $p_{obj} \in [0, 1]$ and a class-wise IoU score $p_{iou} \in \mathbb{R}^C, p_{iou}^i \in [0, 1]$ for each bounding box. We only keep boxes with $p_{obj} \geq \tau_{\text{hard}}$ and $\max(p_{iou}) \geq \tau_{\text{hard}}$, and further ignore losses on those with $p_{obj} < \tau_{\text{soft}}$ and $\max(p_{iou}) < \tau_{\text{soft}}$.

Self-training on Pseudo Labels. We still use the same batch size, data augmentation, and data sampling methods. Since pseudo labels have a much higher labeling frequency than the original ground-truth labels, the effective training batch size under the same event sequence length is larger. Following the square root scaling law [5], we use a higher learning rate of 5×10^{-4} on Gen1 [2] and 8×10^{-4} on 1Mpx. We train for 150k and 200k steps in round 1 and round 2 self-training, respectively. At each training step, we first conduct the normal anchor assignment process [3] to compute training losses, and then set the losses on anchors associated with uncertain boxes (boxes with a detection score lower than τ_{soft} and the ignored and inpainted boxes from tracking-based post-processing) as 0.

Training Objective of RVT. RVT adopts the anchor-free YOLOX [3] detection head. Let $o^i \in \{0, 1\}$ denote whether an anchor point is matched to a ground-truth box kept after label filtering, and $r^i \in \{0, 1\}$ denote whether it is matched to a box removed in tracking or soft anchor assignment (thus ignored in loss computation), the training loss of RVT is:

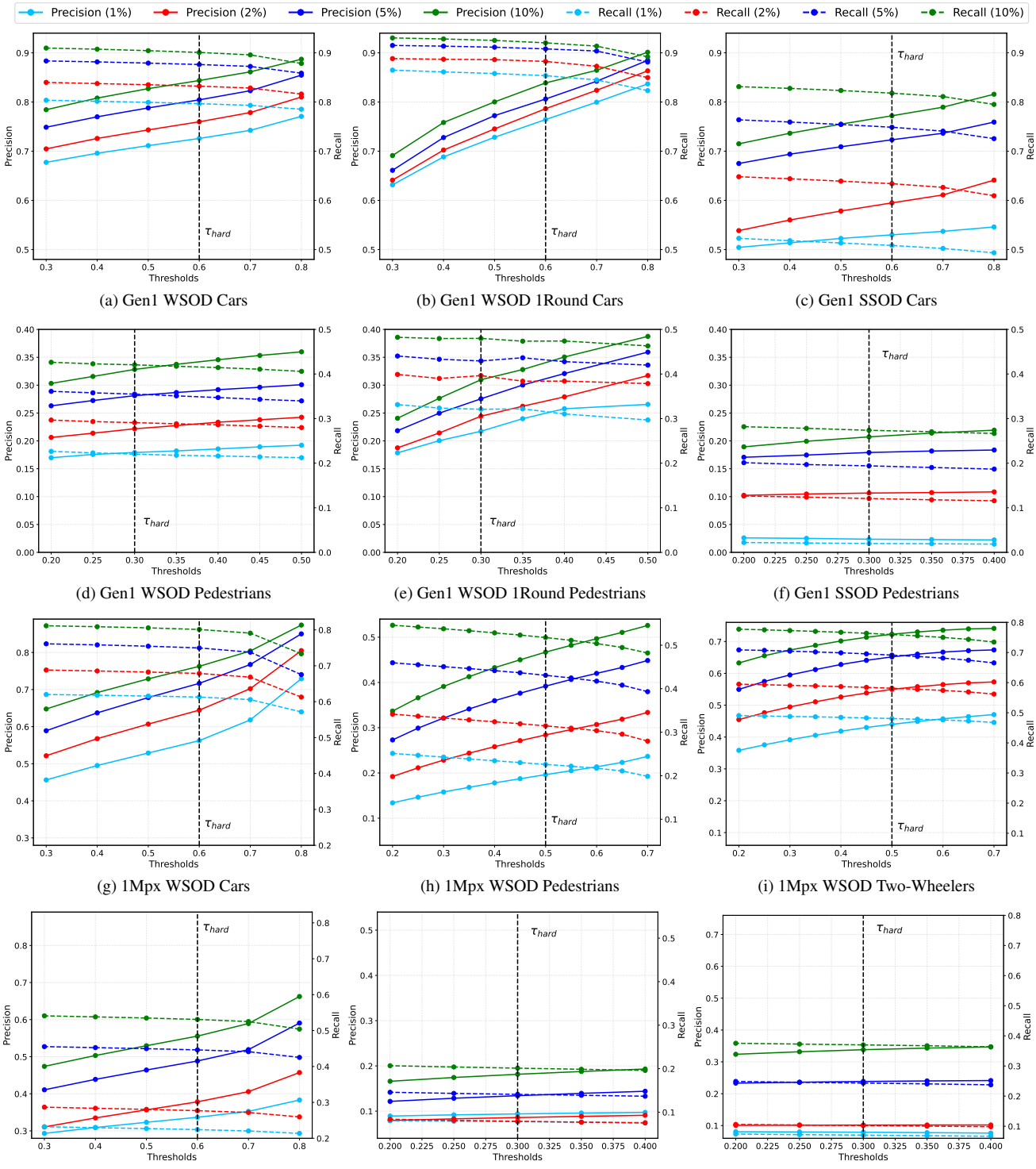
$$\begin{aligned} L = &\mathbb{1}_{\{r^i=0\}} L_{\text{BCE}}(p_{\text{obj}}^i, o^i) \; + \; \mathbb{1}_{\{o^i=1\}} L_{\text{CE}}(p_{\text{iou}}^i, l^i) \\ &+ \; \mathbb{1}_{\{o^i=1\}} L_{\text{IoU}}(\Delta b^i, b^i) \end{aligned} \tag{1}$$

Our proposed components only bring negligible overheads to model training. Therefore, we can train our model on 2 NVIDIA A40 GPUs. The pre-training stage takes 60 hours, while self-training takes around 40 hours.

B. Detailed Analysis of Pseudo Label Quality

Fig. 1 shows the precision and recall of pseudo labels under different settings and thresholds. They are computed by evaluating pseudo labels against the ground-truth labels at

[1] https://github.com/uzh-rpg/RVT



(a) Gen1 WSOD Cars (b) Gen1 WSOD 1Round Cars (c) Gen1 SSOD Cars

(d) Gen1 WSOD Pedestrians (e) Gen1 WSOD 1Round Pedestrians (f) Gen1 SSOD Pedestrians

(g) 1Mpx WSOD Cars (h) 1Mpx WSOD Pedestrians (i) 1Mpx WSOD Two-Wheelers

(j) 1Mpx SSOD Cars (k) 1Mpx SSOD Pedestrians (l) 1Mpx SSOD Two-Wheelers

Figure 1. We plot the precision and recall of pseudo labels generated under different settings. In each figure, solid lines represent precision and dotted lines represent recall. Four labeling ratios $1\%, 2\%, 5\%, 10\%$ are selected. The black dotted line is the threshold for label filtering. We fix the Y-axis value range within each ground {(a), (b), (c)}, {(d), (e), (f)}, {(g), (j)}, {(h), (k)}, {(i), (l)} for easy comparisons.

annotated but skipped frames. If a predicted box has an IoU higher than 0.75 with a ground-truth box, we treat it as a positive detection. We make the following observations:

More pre-training labels lead to better quality. In all settings, models pre-trained with more labels produce pseudo labels with clearly higher precision and recall.

Cars are much easier to detect than other categories. Comparing cars, pedestrians, and two-wheelers, it is clear that cars have a much better label quality in all settings. This is because cars are larger and there are more bounding box annotations of cars than other objects. On 1Mpx, two-wheelers are slightly easier to detect than pedestrians. Future work can study how to address the class-imbalance issue and improve detections on hard examples.

Self-training improves pseudo label quality, but may degrade precision. Comparing Fig. 1 (a) and (b), (d) and (e), we can see that one round of self-training greatly improves the recall (dotted lines). However, the precision (solid lines) drops if we use a small τ_{hard}. This is because the model learns to discover more objects after self-training, but is also over-confident in its predictions. Therefore, fewer false positives are removed in the filtering process. One solution is to increase the threshold τ_{hard} over the number of self-training rounds, as done in [10]. We tried this in our preliminary experiments but did not observe a clear improvement.

Weakly-supervised learning (WSOD) leads to better results than semi-supervised learning (SSOD). Comparing the WSOD and SSOD results in Fig. 1, we can see that models trained in WSOD produce much higher quality pseudo labels than their SSOD counterparts. Together with the detection mAP results presented in Sec. 4.2, this proves that sparsely labeling as many event streams as possible is better than densely labeling a few event sequences.

Gen1 vs. 1Mpx. Comparing Fig. 1 (a) and (g), (c) and (j), it is clear that models on Gen1 detect cars much better than on 1Mpx. This is because 1Mpx has a higher resolution and the number of cars per frame is also larger (1Mpx: 3.8 vs Gen1: 1.9). Interestingly, as can be seen from Fig. 1 (d) and (h), the label quality of pedestrians on Gen1 is worse than on 1Mpx. After visualizing some results, we realize that this is because Gen1 does not provide annotations for two-wheelers, but the model detects lots of two-wheelers as pedestrians, which are regarded as false positives. In contrast, 1Mpx does not have this issue as two-wheelers are also labeled which disambiguates model learning. Indeed, the gap in precision is much higher than recall, as precision penalizes false positives. Future work can study how to learn more discriminative features to separate object categories, e.g., with class-centric contrastive loss [6].

C. Visualization of Pseudo Labels

We visualize some pseudo labels on Gen1 in Fig. 2.

Failure case analysis. Tracking-based post-processing is able to eliminate temporally inconsistent boxes. However, since we use a fixed threshold $T_{trk} = 6$ for all tracks, some objects may be incorrectly removed. In Fig. 2 (a), the car highlighted by the purple arrow is a hard example as it only triggers a few events. The model only detects it in one frame while missing it in later frames, leading to a short track length. As a result, the correct detection at $t = 16$ is mistakenly removed. In Fig. 2 (b), the cars coming from the other direction move very fast, and only stay visible for less than T_{trk} timesteps. Thus, they are also wrongly removed. Nevertheless, since we ignore these boxes during model training instead of suppressing them as background, such errors are less harmful. Fig. 2 (c) shows another failure case where a two-wheeler is recognized as a pedestrian as discussed in Appendix B.

Successful examples. In Fig. 2 (c), we visualize the tracking trajectory of a pedestrian (the green curve). Although the pedestrian is occluded and thus not detected at $t = 16$, our tracker is able to re-identify it at $t = 21$, thus keeping it in the pseudo labels. Fig. 2 (d) shows an example where a car is not annotated in the ground-truth labels. Our model successfully discovers it and corrects the annotation error.

D. Discussion on Experimental Setting Naming

In this paper, we propose two settings under the label-efficient event-based detection task: **(i)** weakly-supervised object detection (WSOD) where all event sequences are sparsely annotated, and **(ii)** semi-supervised object detection (SSOD) where some event sequences are densely annotated, and others are fully unlabeled. While (ii) undoubtedly belongs to semi-supervised learning, (i) may be controversial. In fact, the definition of weakly- and semi-supervised learning is often overlapping in the literature. For example, the Wikipedia page[2] seems to give similar definitions to these two tasks: "**Weak supervision**, also called **semi-supervised learning**, is a paradigm in machine learning..." Previous surveys [9, 12] identify a key property in semi-supervised learning: labeled and unlabeled data should be (although from the same distribution) independent of each other. In contrast, the labeled frames in a sparsely labeled event sequence are not independent of the unlabeled frames in the same sequence. On the other hand, another survey on weakly-supervised learning [14] regards "incomplete supervision where only a subset of training data are given with labels" as one type of weak supervision, which is similar to our sparse labeling setting. These are the main reasons we term (i) weakly-supervised learning to differentiate it from semi-supervised learning.

However, we note that some works [7, 11] learning video object detection with sparsely labeled frames call their setting semi-supervised learning. Moreover, if we employ a feedforward detector, i.e., detectors that do not leverage temporal information, setting (i) becomes closer to semi-supervised learning as labeled and unlabeled timesteps become less relevant. Nevertheless, we believe recurrent detectors are the future trend in event-based object detection as they lead to significantly stronger performance.

[2] https://en.wikipedia.org/wiki/Weak_supervision

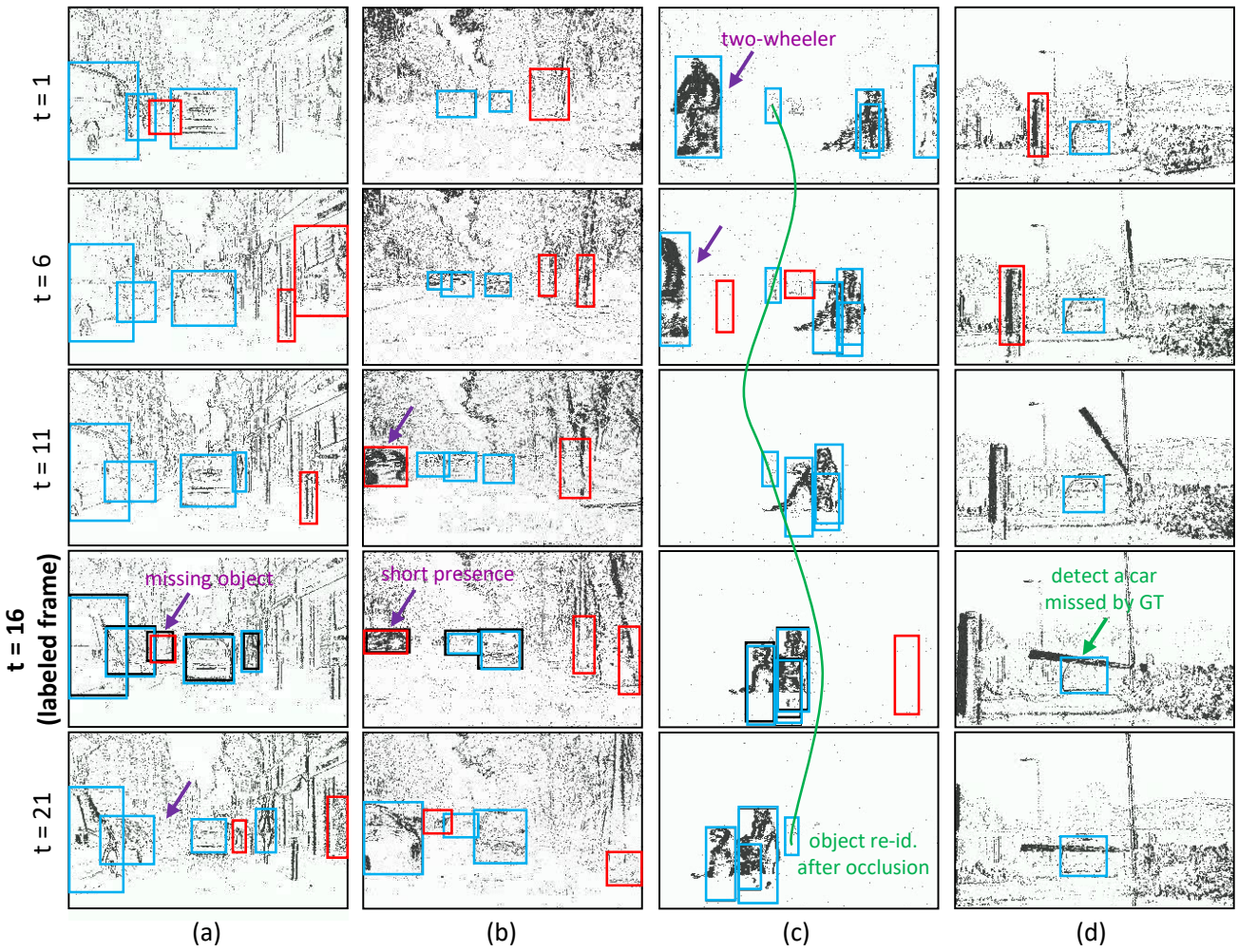


Figure 2. We visualize some pseudo labels on Gen1 that are generated by an RVT-S after one round of self-training. Blue boxes are pseudo labels kept for model training while red boxes are those removed by tracking-based post-processing. Black boxes at $t = 16$ are ground-truth annotations. The t here denotes timesteps of the event frame representation instead of seconds in the real world. Purple arrows highlight some failure cases of our method while green arrows highlight some desired behaviors.

E. Societal Impact

This paper proposes a framework to learn better event-based object detectors with limited labeled data. Object detection is a core task in computer vision that is used across a wide variety of applications including healthcare, entertainment, communication, mobility, and defense. While only a subset of scenarios in this application can benefit from event-camera data, it is still difficult to predict the overall impact of the technology. Moreover, event-based detectors may introduce biases that are different from those encountered in classical cameras and better understanding such biases is an open research problem. While we do not see any immediate risks of human rights or security violations introduced by our work, future work building upon it will carefully need to investigate implications on its particular application area.

References

[1] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *ICIP*, 2016. 2

[2] Pierre De Tournemire, Davide Nitti, Etienne Perot, Davide Migliore, and Amos Sironi. A large scale event-based detection dataset for automotive. *arXiv preprint arXiv:2001.08499*, 2020. 2

[3] Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. Yolox: Exceeding yolo series in 2021. *arXiv preprint arXiv:2107.08430*, 2021. 2

[4] Mathias Gehrig and Davide Scaramuzza. Recurrent vision transformers for object detection with event cameras. In *CVPR*, 2023. 2

[5] Elad Hoffer, Itay Hubara, and Daniel Soudry. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. *NeurIPS*, 2017. 2

[6] Minghua Liu, Yin Zhou, Charles R Qi, Boqing Gong, Hao Su, and Dragomir Anguelov. Less: Label-efficient semantic segmentation for lidar point clouds. In *ECCV*, 2022. 4

[7] Ishan Misra, Abhinav Shrivastava, and Martial Hebert. Watch and learn: Semi-supervised learning for object detectors from video. In *CVPR*, 2015. 4

[8] Etienne Perot, Pierre De Tournemire, Davide Nitti, Jonathan Masci, and Amos Sironi. Learning to detect objects with a 1 megapixel event camera. *NeurIPS*, 2020. 2

[9] Jesper E Van Engelen and Holger H Hoos. A survey on semi-supervised learning. *Machine learning*, 2020. 4

[10] Xudong Wang, Rohit Girdhar, Stella X Yu, and Ishan Misra. Cut and learn for unsupervised object detection and instance segmentation. In *CVPR*, 2023. 4

[11] Pengxiang Yan, Guanbin Li, Yuan Xie, Zhen Li, Chuan Wang, Tianshui Chen, and Liang Lin. Semi-supervised video salient object detection using pseudo-labels. In *ICCV*, 2019. 4

[12] Xiangli Yang, Zixing Song, Irwin King, and Zenglin Xu. A survey on deep semi-supervised learning. *TKDE*, 2022. 4

[13] Lunjun Zhang, Anqi Joyce Yang, Yuwen Xiong, Sergio Casas, Bin Yang, Mengye Ren, and Raquel Urtasun. Towards unsupervised object detection from lidar point clouds. In *CVPR*, 2023. 2

[14] Zhi-Hua Zhou. A brief introduction to weakly supervised learning. *National Science Review*, 2018. 4