Sim-to-Real 6D Object Pose Estimation via Iterative Self-training for Robotic Bin Picking

Kai Chen[1], Rui Cao[1], Stephen James[2], Yichuan Li[1],
Yun-Hui Liu[1], Pieter Abbeel[2], and Qi Dou[1]

[1] The Chinese University of Hong Kong
[2] University of California, Berkeley

Abstract. 6D object pose estimation is important for robotic bin-picking, and serves as a prerequisite for many downstream industrial applications. However, it is burdensome to annotate a customized dataset associated with each specific bin-picking scenario for training pose estimation models. In this paper, we propose an iterative self-training framework for sim-to-real 6D object pose estimation to facilitate cost-effective robotic grasping. Given a bin-picking scenario, we establish a photo-realistic simulator to synthesize abundant virtual data, and use this to train an initial pose estimation network. This network then takes the role of a teacher model, which generates pose predictions for unlabeled real data. With these predictions, we further design a comprehensive adaptive selection scheme to distinguish reliable results, and leverage them as pseudo labels to update a student model for pose estimation on real data. To continuously improve the quality of pseudo labels, we iterate the above steps by taking the trained student model as a new teacher and re-label real data using the refined teacher model. We evaluate our method on a public benchmark and our newly-released dataset, achieving an ADD(-S) improvement of 11.49% and 22.62% respectively. Our method is also able to improve robotic bin-picking success by 19.54%, demonstrating the potential of iterative sim-to-real solutions for robotic applications. Project homepage: www.cse.cuhk.edu.hk/~kaichen/sim2real_pose.html.

Keywords: Sim-to-Real Adaptation, 6D Object Pose Estimation, Iterative Self-training, Robotic Bin Picking

1 Introduction

6D object pose estimation aims to identify the position and orientation of objects in a given environment. It is a core task for robotic bin-picking and plays an increasingly important role in elevating level of automation and reducing production cost for various industrial applications [10,28,29,50,49]. Though recent progress has been made, this task still remains highly challenging, given that industrial objects are small and always cluttered together, leading to heavy occlusions and incomplete point clouds. Early methods have relied on domain-specific knowledge and designed descriptors based on material textures or shapes of industrial objects [11,20]. Recently, learning-based models [19,34,41,48] that

directly regress 6D pose from RGB-D inputs have emerged as more general solutions with promising performance for scalable application in industry.

However, training deep networks requires a large amount of annotated data. Different from labeling for ordinary computer vision tasks such as object detection or segmentation, annotations of 6D object pose is extremely labor-intensive (if possible), as it manages RGB-D data and involves several cumbersome steps, such as camera localization, scene reconstruction, and point cloud registration [2,15,32]. Without an easy way to get pose labels for real environments, the practicality of training object pose estimation networks lessens.

Perhaps then, the power of simulation can be leveraged to virtually generate data with labels of object 6D pose to train deep learning models for this task [22]. However, due to the simulation-to-reality gap, if we exclusively train a model with synthetic data, it is not guaranteed to also work well on real data. Sim-to-real adaptation methods are therefore required. Some methods use physical-based simulation [23] or mixed reality [53] to make simulators as realistic as possible. Other works [37,39,45] use domain randomization [24,43] to vary the simulated environments with different textures, illuminations or reflections. Some recent methods resort to features that are less affected by the sim-to-real gap for object pose estimation [14,30]. Alternatively, some methods initially train a model on synthetic data, and then refine the model by self-supervised learning on unlabeled real data [7,52]. Unfortunately, these sim-to-real methods either rely on a customized simulation to produce high-quality synthetic data or need a customized network architecture for extracting domain-invariant signal for sim-to-real pose estimation. Given that a robot would be applied to various upcoming bin-picking scenarios, customizing the method to fit for various different real scenes could also be labor-intensive. In addition, since the adaptation to complex bin-picking scenarios is not sufficient, obvious performance gap between simulation and real evaluation is still present in existing sim-to-real methods.

These current limitations encourage us resorting to a new perspective to tackle this challenging task. In general, a model trained on synthetic data should have acquired essential knowledge about how to estimate object poses. If we directly expose it to real data, its prediction is not outrageously wrong. If the model could directly self-train itself leveraging the successful practices, it is highly likely to efficiently adapt to reality, just with unlabeled data. Collecting a large amount of unlabeled real data is easy to achieve for a robot. More importantly, this adaptation process is scalable and network-agnostic. In any circumstances, the robot can be deployed with the most suitable deep learning network for its target task, record sufficient unlabeled real data, and then leverage the above self-training adaptation idea for accurate object pose estimation in its own scenario. Promisingly, similar self-training paradigms have been recently explored for some other computer vision applications such as self-driving and medical image diagnosis, for handling distribution shift towards real-world use [33,36,55]. However, the question of how it can be used for reducing annotation cost while keeping up real-data performance of 6D object pose estimation for robots is unknown.

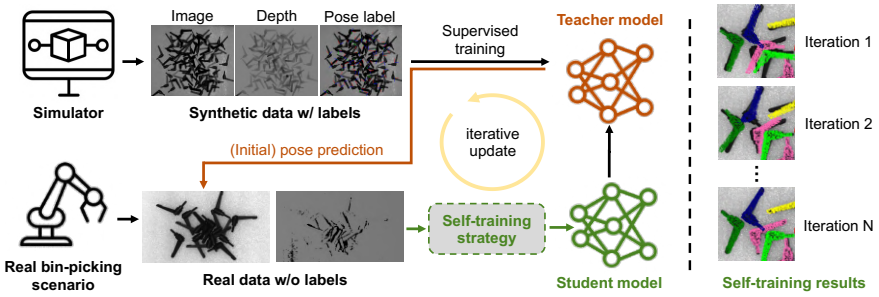


Fig. 1. For industrial bin-picking, we build a simulator to generate synthetic data for learning a teacher model. Starting from it, our novel iterative self-training method can select reliable pseudo-labels and progressively improve the 6D object pose accuracy on real data, without the need for any manual annotation.

In this paper, we propose the first sim-to-real iterative self-training framework for 6D object pose estimation. Fig. 1 depicts an overview of our framework. First, we establish a photo-realistic simulator with multiple rendering attributes to synthesize abundant data, on which a pose estimation network is initially well-trained. Then, it is taken as a teacher model to generate pose predictions for unlabeled real data. From these results, we design a new pose selection method, which comprehensively harnesses both 2D appearance and 3D geometry information of objects to carefully distinguish reliable predictions. These high-quality pose predictions are then leveraged as pseudo labels for the corresponding real data to train the pose estimation network as a student model. Moreover, to progressively improve the quality of pseudo labels, we consider the updated student model as a new teacher, refine pose predictions, and re-select the highest-quality pseudo labels again. Such an iterative self-training scheme ensures to simultaneously improve the quality of pseudo labels and model performance on real data. According to extensive experiments on a public benchmark dataset and our constructed *Sim-to-real Industrial Bin-Picking (SIBP)* dataset, our proposed iterative self-training method significantly outperforms the state-of-the-art sim-to-real method [39] by 11.49% and 22.62% in terms of the ADD(-S) metric.

2 Related Works

2.1 6D Object Pose Estimation for Bin-Picking

Though recent works [8,23,34,37] show superior performance on household object datasets (*e.g.,* LineMOD [20] and YCB Video [56]), 6D pose estimation for bin-picking still remains a challenging task for highly cluttered and complex scenes. Conventional methods typically leveraged Point Pair Features (PPF) to estimate 6D object pose in a bin [3,4,11,46]. As a popular method, it mainly relies on depth or point cloud to detect the pose in the scene and achieves promising results when scenes are clear and controlled [23]. However, due to the reliance

on the point cloud, the PPF-based methods are sensitive to the noise and occlusion in cluttered scenes, which is very common in real bin-picking scenarios such as industrial applications [27,60]. Current state-of-the-art methods [9,27] and datasets [28,60] rely on deep learning models for handling complex bin-picking scenarios. Dong et al. [9] proposed a deep-learning-based voting scheme to estimate 6D pose using a scene point cloud. Yang et al. [60] modified AAE [38] with a detector as a deep learning baseline, showing a better performance than the PPF-based method for bin-picking scenes. For these methods, regardless of the particular network design, a large amount of real-world labeled data is always required in order to achieve a high accuracy.

2.2 Visual Adaptation for Sim-to-real Gap

Sim-to-real transfer is a crucial topic to tackle the bottleneck for deep learning-based robotic applications, which is essential to bridge the domain gap between simulated data and real-world observation through robotic visual systems [31,52]. A promising technique for achieving sim-to-real transfer is domain randomization [26,31,45], which samples a huge diversity of simulation settings (e.g. camera position, lighting, background texture, etc.) in a simulator to force the trained model to learn domain-invariant attributes and to enhance generalization on real-world data. Visual domain adaptation [37,38,39] has also had recent success for sim-to-real transfer, where the source domain is usually the simulation, and the target domain is the real world [1,25]. Recently, some works [8,23,44,45,52] leverage physically-based renderer (PBR) data to narrow the sim-to-real gap by simulating more realistic lighting and texture in the simulator. However, with the additional synthetic data, current works on 6D pose estimation still mainly rely on annotated real data to ensure performance.

2.3 Self-training via Iterative Update

A typical self-training paradigm [16,61] first trains a teacher model based on the labeled data, then utilizes the teacher model to generate pseudo labels for unlabeled data. After that, the unlabeled data with their pseudo labels are used to train a student model. As a typical label-efficient methodology, self-training techniques have recently gained attention in many fields. Some methods apply self-training to image classification [47,59]. They use the teacher model to generate soft pseudo labels for unlabeled images, and train the student model with a standard cross-entropy loss. The input consistency regularization [57] and noisy student training [58] are widely used techniques to further enhance the student model with unlabeled data. Some works adopt self-training in semantic segmentation [33,63]. Instead of training the student model with merely pseudo labels, they train the student by jointly using a large amount of data with pseudo labels and a small number of data with manual labels. Some other works resort to self-training to perform unsupervised domain adaptation [36,64], which train the teacher and student models on different domains. Recently, some methods study

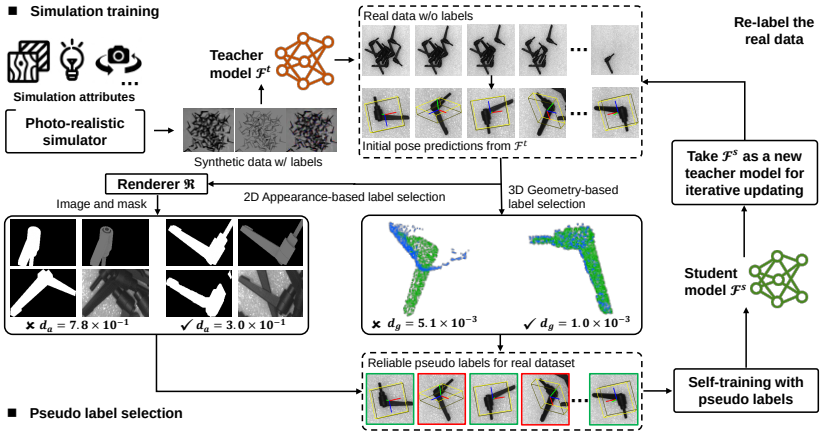


Fig. 2. Overview of our proposed iterative self-training method for 6D object pose estimation. We first build a photo-realistic simulation with multiple rendering attributes to generate synthetic data. We then train a teacher model on the synthetic data, which is used to generate initial pose predictions for the unlabeled real data. Subsequently, a pseudo label selection scheme with both 2D appearance and 3D geometry metrics are adopted to select data with reliable pose predictions, which are used to train a student model for the real data. We iterate this self-training scheme multiple times by taking the trained student model as a new teacher for the next iteration.

pseudo label selection. They pick out reliable pseudo labels based on the probability outputs or the uncertainty measurements derived from the network [13,35]. Though the effectiveness of self-training has been revealed in many fields, we find that self-training has not been investigated for 6D object pose estimation.

3 Method

In this paper, we take advantage of advanced self-training to solve the problem of sim-to-real object pose estimation. Our method is designed to be scalable and network-agnostic, and aims to dramatically improve pose estimation accuracy on real-world data without need of any manual annotation.

3.1 Overview of Sim-to-Real Object Pose Estimation

Without loss of generality, let $\mathcal{F}$ denote an arbitrary 6D object pose estimation network, which takes RGB-D data of an object as input, and outputs its corresponding 6D pose with respect to the camera coordinate frame. Let I be the RGB image and G denote the 3D point cloud recovered from the depth map, the 6D object pose p is predicted by $\mathcal{F}$ as:

$$p = [R|t] = \mathcal{F}(I, G), \tag{1}$$

where $R \in SO(3)$ is the rotation and $t \in \mathbb{R}^3$ is the translation. Fig. 2 summarizes our network-agnostic, sim-to-real pose estimation solution. Given a target bin-picking scenario, we first create a photo-realistic simulation to generate abundant synthetic data with a low cost, which is used to initially train a teacher model $\mathcal{F}^t$ with decent quality. We then apply $\mathcal{F}^t$ on real data to predict their object poses (Sec. 3.2). Based on these initial estimations, a new robust label selection scheme (Sec. 3.3) is designed to select the most reliable predictions for pseudo labels, which are incorporated by self-training to get a student model $\mathcal{F}^s$ for the real data (Sec. 3.4). Importantly, we iterate the above steps by taking the trained $\mathcal{F}^s$ as a new teacher model, in order to progressively leverage the knowledge learned from unlabeled real data to boost the quality of pseudo labels as well as the student model performance on real data.

3.2 Simulation Training

First of all, we construct a photo-realistic simulator to generate synthetic data for the bin-picking scenario. To mimic a real-world cluttered bin-picking scenario, we randomly generate realistic scenes with multiple objects closely stacked in diverse poses. Note that our simulator is light-weight without the need for complex scene-orientated modules (such as mixed/augmented reality), which means that once built, the simulator can be widely applied to different industrial bin-picking scenarios. Using the simulator, we generate a large amount of data with precise labels, and use it to train a teacher model $\mathcal{F}^t$ with acceptable initial performance. After that, when we expose $\mathcal{F}^t$ to real data (I_i, G_i), we can obtain its pose prediction as $\widetilde{p_i} = \mathcal{F}^t(I_i, G_i)$. As shown in Fig. 2, due to the unavoidable gap between virtual and real data, $\widetilde{p_i}$ is not guaranteed to always be correct. In order to self-train the model on unlabeled real data, we need to carefully pick out reliable predictions of $\mathcal{F}^t$. This step is known as pseudo label selection in a typical self-training paradigm.

3.3 Pose Selection for Self-training

Given unlabeled real data $\{(I_i, G_i)\}_{i=1}^m$ and their initial pose predictions $\{\widetilde{p_i} = [R_i|t_i]\}_{i=1}^m$, pose selection aims to find out reliable pose results. Existing pseudo label selection methods [13,35,58] are limited to classification-based tasks, such as image recognition and semantic segmentation. But for object pose estimation, which is typically formulated as a regression learning process, it is unclear how to select pseudo labels. In order to address this problem, we propose a comprehensive pose selection method. The core idea is to first virtually generate the observation data that corresponds to the predicted pose. Subsequently, we compare the generated data with the real collected one to determine whether the predicted pose is reliable or not. Note that for a robust pose selection, both the 2D image and the 3D point cloud are important. As shown in Fig. 3, although 2D image provides rich features for discriminating object poses, they could be ambiguous due to the information loss when projecting the 3D object onto the 2D image plane. Using 3D point cloud to assess the object pose can avoid this

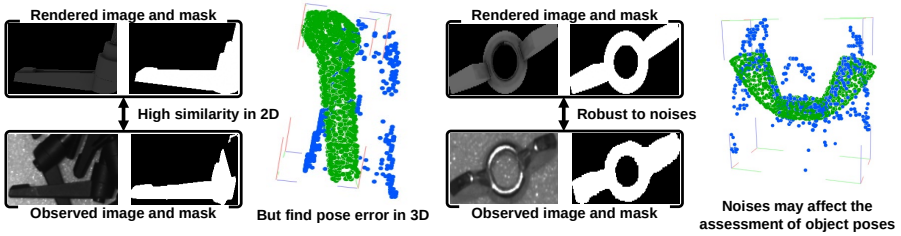


Fig. 3. Both 2D appearance and 3D geometry are important for reliable pseudo label selection. On the one hand, the 3D geometry metric can notice pose errors that are hard to detect in 2D. On the other hand, 3D geometry metric gets unreliable for noisy point cloud, but the 2D appearance metric is robust to these noises.

issue. But different from 2D images, the 3D point cloud could be quite noisy in the bin-picking scenario, which affects the assessment result. Our proposed method therefore leverages the complementary advantages of 2D image and 3D point cloud for a comprehensive pose selection for self-training.

2D Appearance-based Pose Selection. To obtain the image that corresponds to the predicted pose, we leverage an off-the-shelf renderer [6], which is denoted as $\Re$. The renderer takes an object pose and the corresponding object CAD model as inputs, and will virtually generate an RGB image I^r and a binary object mask M^r that correspond to the predicted pose[3]:

$$\{I^r, M^r\} = \Re(\widetilde{p}). \tag{2}$$

The binary mask exhibits clear object contour, while the RGB image contains richer texture and semantic information (see Fig. 2). We make combined use of them for appearance-based pose selection on the 2D image plane.

Let M^o denote the object mask for the observed RGB image, which can be accurately acquired by an existing segmentation network [18] trained on synthetic data. Rather than directly measuring the contour similarity [31] and to ensure that the mask-based metric is robust to segmentation noise, we evaluate the pose quality based on the pixel-wise overlap of M^r and M^o, which is less affected by imperfect segmentation:

$$s(M^r, M^o) = \frac{1}{2} \times \left(\frac{1}{N_+} \sum_{q \in M^o_+} \mathbb{1}(q \in M^r_+) + \frac{1}{N_-} \sum_{q \in M^r_-} \mathbb{1}(q \in M^o_-)\right), \tag{3}$$

where $M^{\{r,o\}}_+$ denotes the foreground region of the mask and $M^{\{r,o\}}_-$ denotes the background region of the mask. N_+ and N_- are the number of pixels of M^o_+ and M^r_-, respectively. Given the mask overlap, the corresponding mask-based pose distance metric could be computed as $d_{mask} = 1 - s(M^r, M^o)$. This mask-based

[3] For clarity, we will omit the subscript i for formulations in this section.

metric coarsely leverages the object contour information for pose evaluation, but neglects the detailed texture and semantic information of the object. As a consequence, d_{mask} would be sensitive to occlusion and not reliable enough for objects with a complex shapes.

In this regard, we further leverage the rendered RGB image to enhance the pose assessment. In order to extract representative features from the RGB image, we use a pre-trained CNN Φ to transform I^r and I^o into multi-level high-dimension features, based on which we measure the perceptual distance [62] between I^r and I^o as:

$$d_{image} = \sum_{l=1}^{L} \frac{1}{N_l} \sum_{q} \|\Phi_l^r(q) - \Phi_l^o(q)\|_2, \tag{4}$$

where $\Phi_l^{\{r,o\}}$ denotes the l-th level normalized feature of Φ. N_l is the number of pixels of the corresponding feature map. The perceptual distance d_{image} complements d_{mask} with low-level texture and high-level semantic features. We then integrate them as $d_a = d_{mask} \times d_{image}$ to assess the initial pose prediction $\widetilde{p}$ on the 2D image plane.

3D Geometry-based Pose Selection. The 2D appearance-based metric is good at measuring the in-plane translation and rotation with informative features on 2D image plane. Nevertheless, it is not sufficient to comprehensively assess a complete 6D pose. To address this limitation, we propose to further leverage the object point cloud G^o to enhance the pose selection scheme in 3D space. In order to use the point cloud consistency in 3D space to assess the pose quality, we need to generate the object point cloud that corresponds to the predicted pose. With the object CAD model $\mathbb{C}$ and the predicted pose $\widetilde{p}$, we perform the following 2 steps. First, we apply $\widetilde{p}$ to the CAD model: $\mathbb{C}^r = \widetilde{R} \times \mathbb{C} + \widetilde{t}$. It transforms the original CAD model $\mathbb{C}$ in the object coordinate frame to a model $\mathbb{C}^r$ in the camera coordinate frame. After that, since $\mathbb{C}^r$ is a complete point cloud while G^o only contains the surface point cloud that could be observed from a view point, directly evaluating the point cloud consistency between $\mathbb{C}^r$ and G^o cannot precisely reveal the quality of $\widetilde{p}$. To mitigate this problem, we further apply a projection operator proposed by [17] to the CAD model in camera coordinate frame: $G^r = \mathbb{P}(\mathbb{C}^r)$. This generates the surface points of $\mathbb{C}^r$ that can be observed from the predicted pose. Intuitively, if $\widetilde{p}$ is precise, G^r should be well aligned with the observed point cloud G^o in 3D space. We use the Chamfer Distance (CD) between G^r and G^o as a 3D metric to quantify the quality of $\widetilde{p}$:

$$d_g = \frac{1}{N_1} \sum_{x \in G^r} \min_{y \in G^o} \|x - y\|_2 + \frac{1}{N_2} \sum_{y \in G^o} \min_{x \in G^r} \|x - y\|_2, \tag{5}$$

where x and y denote 3D points from G^r and G^o, respectively. N_1 and N_2 are the total numbers of points for G^r and G^o.

Comprehensive Selection. Given the complementary advantages of d_a and d_g, we leverage both of them to single out reliable pose predictions. Pose predictions are designated as pseudo labels and included in the next iteration of self-training when $d_a < \tau_a$ and $d_g < \tau_g$. These two thresholds τ_a and τ_g can be determined flexibly according to the metric value distribution on unlabeled real data. In this paper, we adaptively set $\tau_{(a,g)} = \mu_{(a,g)} + \sigma_{(a,g)}$, where $\mu_{(a,g)}$ and $\sigma_{(a,g)}$ are the mean and standard deviation of d_a and d_g on real data.

3.4 Iterative Self-training on Real Data

After label selection, we incorporate data with reliable pseudo labels to the 6D object pose estimation network to train a student model $\mathcal{F}^s$. Our self-training scheme aims to enforce the prediction of $\mathcal{F}^s$ to be consistent with pose pseudo labels. In this regard, we define the self-training loss function as $\mathcal{L}_{\text{self-training}} = \frac{1}{M}\sum_{i=1}^{M} \ell(\widetilde{p}_i, p'_i)$, where p'_i is the pose prediction from the student model, ℓ could be any loss function of object pose estimation and is depended on the architecture of $\mathcal{F}$. Through self-training, we transfer the knowledge that the teacher model has learned from the synthetic data to the student model on real data.

To further improve the capability of the student model, we iterate the above process for multiple runs. With progressive training on the real data, the pseudo labels generated by the updated teacher model will have a higher quality than the previous iteration's labels. Our experiments demonstrate that this iterative optimization scheme can effectively boost the student model on real data.

4 Experiments

In this section, we will answer the following questions: (1) Does the proposed self-training method improve the performance of sim-to-real object pose estimation? (2) Do the appearance-based and geometry-based pose selection metrics improve the student model performance? (3) Could the proposed self-training framework be applied to different backbone architectures? (4) How many iterations are required to self-train the pose estimation network? (5) Does the proposed self-training indeed improve the bin-picking success rate on real robots? To answer question (1)-(4), we evaluate our method and compare it with state-of-the-art sim-to-real methods on both the ROBI [60] dataset and our proposed SIBP dataset. To answer question (5), we deploy our pose estimation model after self-training on a real robot arm and conduct real-world bin-picking experiments.

4.1 Experiment Dataset

ROBI Dataset. ROBI [60] is a recent public dataset for industrial bin-picking. It contains seven reflective metal objects, with two of them being composed of different materials[4]. Since ROBI only contains real RGB-D data, we use the

[4] Din-Connector and D-Sub Connector

simulator presented in Sec. 3.2 to extend ROBI by randomly synthesizing 1000 virtual scenes for each object. For real data, we use 3129 real RGB-D scenes of ROBI. The 3129 real scenes are in two cluttered levels. 1288 scenes are low-bin scenarios, in which a bin contains only 5-10 objects and corresponds to easy cases. 1841 sceres are full-bin scenarios, in which a bin is full with tens of objects and correspond to hard cases. We use 2310 scenes (910 low-bin and 1400 full-bin scenarios) without using their annotations for self-training and use the remaining scenes for evaluation. Please refer to [60] and the supplementary for more detailed information of ROBI.

Sim-to-real Industrial Bin-Picking (SIBP) Dataset. We further build a new SIBP dataset. It provides both virtual and real RGB-D data for six textureless objects in industrial bin-picking scenarios. Compared with ROBI, objects in SIBP are more diverse in color, surface material (3 plastic, 2 metallic and 1 alloyed) and object size (from a few centimeters to one decimeter). We use SIBP for a more comprehensive evaluation. For synthetic data, we use the same simulator to generate 6000 synthetic RGB-D scenes. For real data, we use a Smarteye Tech HD-1000 industrial stereo camera to collect 2743 RGB-D scenes in a real-world bin-picking environment. We use 2025 scenes without using their annotations for self-training, and use the remaining scenes for evaluation. Please refer to supplementary for more detailed information of SIBP.

4.2 Evaluation Metrics

We follow [37,40,52] and compare pose estimation results of different methods *w.r.t.* the ADD metric [20], which measures whether the average distance between models transformed by the predicted pose and the ground-truth pose is smaller than 10% of object diameter. For symmetric objects, we adopted the ADD-S metric [21] when computing the average distance of models. We report the average recall of ADD(-S) for quantitative evaluation.

4.3 Comparison with State-of-the-art Methods

We compared our self-training model with a baseline DC-Net [42] trained exclusively with synthetic data, and two state-of-the-art sim-to-real object pose estimation methods: MP-AAE [37] and AAE [39]. Note that all methods are evaluated based on the same segmentation. Tab. 1 and Tab. 2 presents the quantitative evaluation results. In comparison with the baseline model, on ROBI, our self-training model outperforms DC-Net by 16.73%. On SIBP, it also exceeds DC-Net by 14.94%. These results demonstrate the effectiveness of our self-training method for mitigating the sim-to-real gap for object pose estimation.

Moreover, we compared our method with two SOTA sim-to-real methods. AAE first trained an Augmented Autoencoder on synthetic data and then used the embedded feature of Autoencoder for object pose estimation on real data. MP-AAE further resorted to multi-path learning to learn a shared embedding space for pose estimation of different objects on real data. Our self-training

Table 1. Quantitative comparison with state-of-the-art methods on ROBI dataset.The average recall (%) of the ADD(-S) metric is reported. * indicates asymmetric objects.

Object	Zigzag*	Chrome Screw	Gear	Eye Bolt	Tube Fitting	Din* Connector	D-Sub* Connector	Mean
MP-AAE [37]	22.76	47.49	66.22	50.85	69.54	15.92	4.70	39.64
AAE [39]	25.80	64.02	90.64	67.31	91.58	21.83	7.35	52.65
DC-Net [42]	30.96	67.74	77.56	53.52	74.84	18.72	10.60	47.41
Ours	**45.83**	**79.88**	**97.81**	**89.21**	**97.45**	**24.21**	**14.62**	**64.14**

Table 2. Quantitative comparison with state-of-the-art methods on SIBP dataset.The average recall (%) of the ADD(-S) metric is reported. * indicates asymmetric objects.

Object	Cosmetic	Flake	Handle*	Corner	Screw Head	T-Shape Connector	Mean
MP-AAE [37]	58.04	21.56	27.26	39.59	36.67	32.44	35.93
AAE [39]	58.93	53.82	21.67	55.93	39.04	47.23	46.10
DC-Net [42]	68.54	58.74	43.44	53.96	44.53	53.47	53.78
Ours	**79.59**	**69.94**	**75.24**	**62.57**	**56.98**	**68.02**	**68.72**

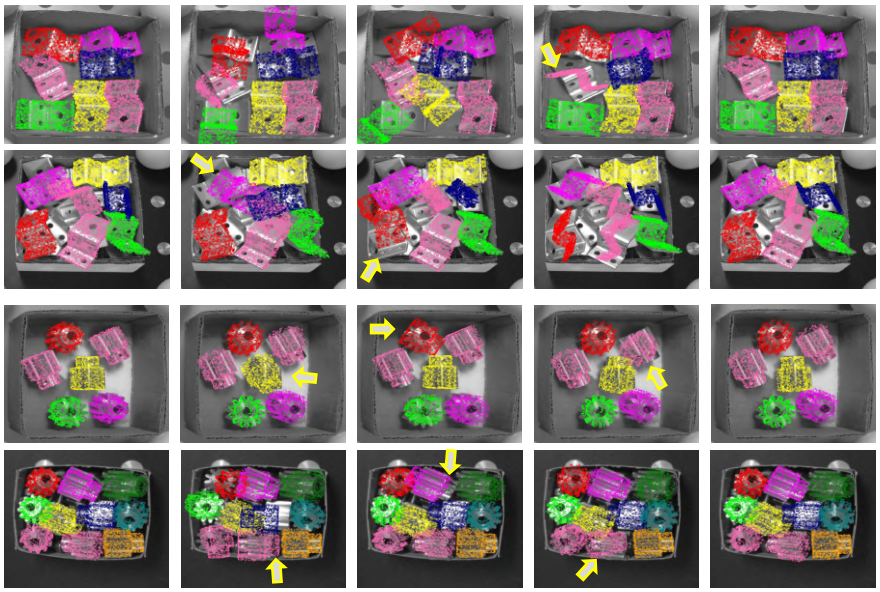
method consistently outperforms these two SOTA methods for object pose estimation. On ROBI, our model outperforms MP-AAE by 24.5% and exceeds AAE by 11.49%. On SIBP, our self-training model achieves an average recall of 68.72%, which is 32.79% higher than MP-AAE and 22.62% higher than AAE. Fig. 4 presents corresponding qualitative comparisons on ROBI. MP-AAE and AAE have difficulties in adapting the model to a real bin-picking environment. The occlusions caused by the container or by neighboring objects may affect their pose estimation results. In comparison, our method can directly adapt the pose estimation model to real data with self-training. It can smoothly handle complex industrial bin-picking scenarios and achieve high pose estimation accuracy. Please refer to the supplementary for more qualitative results.

4.4 Validation of the Pose Selection Strategy

We study the performance of our self-training method with different pose selection strategies. Specifically, we separately remove the appearance-based metric d_a and geometry-based metric d_g from the framework, and evaluate their performances on ROBI. Tab. 3 reports the comparative results. Compared with the lower-bound model (LB), both the proposed appearance metric and geometry metric help to significantly improve the pose accuracy by self-training. Among three different settings, using both appearance and geometry information for pose selection achieves the best pose accuracy. Removing d_g results in a larger pose accuracy drop than removing d_a. These experimental results indicate that merely assessing the pose label in 2D is not sufficient for self-training of 6D object pose. Using both 2D appearance and 3D geometry information provides the most reliable scheme for pose selection, and achieves the best pose accuracy.

Table 3. Quantitative evaluation of the proposed self-training method with different pose selection settings. d_a denotes the appearance-based metric and d_g denotes the geometry-based metric. LB denotes the lower bound of our model, which is trained with only synthetic data. UB denotes the upper bound of our model, which additionally uses pose labels of real data for training.

d_a	d_g	Zigzag*	Chrome Screw	Gear	Eye Bolt	Tube Fitting	Din* Connector	D-Sub* Connector	Mean
✓		38.54	78.00	91.29	85.09	95.52	22.83	12.06	60.48
	✓	43.10	**79.98**	95.20	87.84	96.91	21.61	13.95	62.66
✓	✓	**45.83**	79.88	**97.81**	**89.21**	**97.45**	**24.21**	**14.62**	**64.14**
LB		30.96	67.74	77.56	53.52	74.84	18.72	10.60	47.41
UB		56.90	93.71	98.90	96.12	99.46	44.16	33.86	74.73



GT MP-AAE [37] AAE [39] DC-Net [42] Ours

Fig. 4. Qualitative comparison with state-of-the-arts. The 6D object pose estimation results are depicted by projecting the object model onto the image plane using the object pose and camera intrinsic parameters. DC-Net is our baseline model w/o our proposed self-training. Colored points denote the projected model (best viewed in color).

4.5 Self-training with Different Backbone Networks

In contrary to other self-supervised/self-training methods [7,13,33,52], our proposed self-training framework can be easily applied to different networks for sim-to-real object pose estimation. In this section, we change the backbone network to DenseFusion [51], another popular RGB-D based object pose estimation network. Compared with DC-Net, DenseFusion is a two-stage method which is composed of an initial *estimator* and a pose *refiner*. We then use the same synthetic data and real data with previous experiments for self-training and eval-

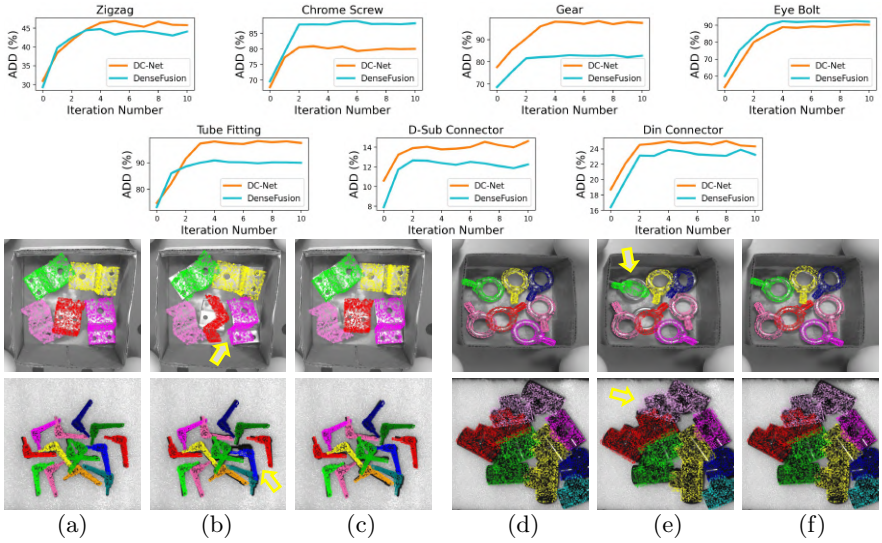


Fig. 5. Object pose estimation results of our iterative self-training model. Top: The average recall of ADD(-S) metric on ROBI dataset with different number of iterations. Bottom: Qualitative results. (a)(d) are ground-truth results. (b)(e) are results of one-time self-training. (c)(f) are results of five-time iterative self-training.

uation. Tab. 4 reports the experiment results. Even with a different backbone network, the proposed self-training method can consistently adapt the model to real data and significantly improve the pose accuracy on real data.

4.6 Continuous Improvement with Iterative Self-training

In order to study the effect of iteration numbers on the final pose accuracy, we perform iterative self-training with a maximal number of 10 iterations, and compared the pose accuracy after each iteration. Fig. 5 depicts the experiment results. In general, the pose error reduces as the number of self-training iterations increase. In most cases, the first iteration usually accounts for the most improvement of the pose accuracy. After that, the second and third iteration can continuously enhance the pose accuracy, with a relatively smaller improvement. After about 5 iterations, the performance slowly saturates without further improvement. Fig. 5 further presents qualitative comparisons between one-time iteration model and five-time iteration model. With the increase of iterations, the pose estimation results get visually closer to the ground truth.

4.7 Effectiveness Demonstration on Robots

In order to further demonstrate the effectiveness of proposed self-training object pose estimation method, we deploy the trained pose estimation model on a

Table 4. Results of the proposed self-training method with a different backbone network [51]. w/o ST and w/ ST denote models without and with proposed self-training.

	Zigzag*	Chrome Screw	Gear	Eye Bolt	Tube Fitting	Din* Connector	D-Sub* Connector	Mean
w/o ST	29.29	69.61	68.59	60.07	73.29	16.42	7.92	46.46
w/ ST	**44.46**	**87.87**	**82.73**	**92.36**	**90.88**	**23.88**	**12.66**	**62.12**

Table 5. Results of the robot bin-picking experiments. The *left* picture illustrates the robot bin-picking environment. **Inst.** denotes the total number of objects to be grasped. **Tria.** denotes the total number of trials needed to access all of the objects.

	Inst. / Tria.					Mean
	01	02	03	04	05	
w/o ST	14/24	14/20	14/18	14/22	14/19	67.96%
w/ ST	14/14	14/16	14/18	14/16	14/16	**87.50%**

Franka Emika Panda robot arm and conduct real-world bin-picking experiments on the *'Handle'* object from SIBP. The grasp configuration is generated offline based on the object CAD model [12], and then projected to the camera coordinate system according to estimated object pose. This means that overall grasping success rate is highly reliant on pose accuracy. Tab. 5 presents the experiment results. Following self-training of the pose estimation network, the grasping success rate dramatically improves. These results demonstrate the effectiveness of our proposed self-training method for practical industrial bin-picking.

5 Conclusion

With the goal of achieving accurate 6D object pose estimation in real-world scenes, we have presented a sim-to-real method that first trains a 6D pose estimation on high-fidelity simulation data, and then performs our iterative self-training method. Our method provides an efficient solution for pose estimation where labeled data is hard (or even impossible) to collect. Extensive results demonstrate that our approach can significantly improve the predicted pose quality, with great potential to be applied to industrial robotic bin-picking scenarios.

Currently, our method assumes access to object models. Although this is acceptable in the industrial bin-picking domain and is indeed common in the pose estimation literature, it would nonetheless be interesting to extend this iterative self-training method to pose estimation of previously unseen objects [5,54]. Our method exploits object mask predicted by a segmentation network trained with synthetic data. It would be meaningful to extend our iterative self-training method to joint instance segmentation and object pose estimation.

Acknowledgement. The work was supported by the Hong Kong Centre for Logistics Robotics.

References

1. Bousmalis, K., Irpan, A., Wohlhart, P., Bai, Y., Kelcey, M., Kalakrishnan, M., Downs, L., Ibarz, J., Pastor, P., Konolige, K., et al.: Using simulation and domain adaptation to improve efficiency of deep robotic grasping. In: ICRA (2018) 4
2. Brachmann, E., Rother, C.: Visual camera re-localization from rgb and rgb-d images using dsac. TPAMI (2020) 2
3. Buch, A.G., Kiforenko, L., Kraft, D.: Rotational subgroup voting and pose clustering for robust 3d object recognition. In: ICCV (2017) 3
4. Buch, A.G., Kraft, D., Robotics, S., Odense, D.: Local point pair feature histogram for accurate 3d matching. In: BMVC (2018) 3
5. Chen, K., Dou, Q.: Sgpa: Structure-guided prior adaptation for category-level 6d object pose estimation. In: ICCV (2021) 14
6. Chen, W., Ling, H., Gao, J., Smith, E., Lehtinen, J., Jacobson, A., Fidler, S.: Learning to predict 3d objects with an interpolation-based differentiable renderer. NeurIPS (2019) 7
7. Deng, X., Xiang, Y., Mousavian, A., Eppner, C., Bretl, T., Fox, D.: Self-supervised 6d object pose estimation for robot manipulation. In: ICRA (2020) 2, 12
8. Di, Y., Manhardt, F., Wang, G., Ji, X., Navab, N., Tombari, F.: So-pose: Exploiting self-occlusion for direct 6d pose estimation. In: ICCV (2021) 3, 4
9. Dong, Z., Liu, S., Zhou, T., Cheng, H., Zeng, L., Yu, X., Liu, H.: Ppr-net: point-wise pose regression network for instance segmentation and 6d pose estimation in bin-picking scenarios. In: IROS (2019) 4
10. Drost, B., Ulrich, M., Bergmann, P., Hartinger, P., Steger, C.: Introducing mvtec itodd-a dataset for 3d object recognition in industry. In: ICCVW (2017) 1
11. Drost, B., Ulrich, M., Navab, N., Ilic, S.: Model globally, match locally: Efficient and robust 3d object recognition. In: CVPR (2010) 1, 3
12. Fang, H.S., Wang, C., Gou, M., Lu, C.: Graspnet-1billion: A large-scale benchmark for general object grasping. In: CVPR (2020) 14
13. Gal, Y., Ghahramani, Z.: Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In: ICML (2016) 5, 6, 12
14. Georgakis, G., Karanam, S., Wu, Z., Kosecka, J.: Learning local rgb-to-cad correspondences for object pose estimation. In: ICCV (2019) 2
15. Gojcic, Z., Zhou, C., Wegner, J.D., Guibas, L.J., Birdal, T.: Learning multiview 3d point cloud registration. In: CVPR (2020) 2
16. Grandvalet, Y., Bengio, Y.: Semi-supervised learning by entropy minimization. In: NeurIPS (2004) 4
17. Gu, J., Ma, W.C., Manivasagam, S., Zeng, W., Wang, Z., Xiong, Y., Su, H., Urtasun, R.: Weakly-supervised 3d shape completion in the wild. In: ECCV (2020) 8
18. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: ICCV (2017) 7
19. He, Y., Sun, W., Huang, H., Liu, J., Fan, H., Sun, J.: Pvn3d: A deep point-wise 3d keypoints voting network for 6dof pose estimation. In: CVPR (2020) 1
20. Hinterstoisser, S., Lepetit, V., Ilic, S., Holzer, S., Bradski, G., Konolige, K., Navab, N.: Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In: ACCV (2012) 1, 3, 10
21. Hodaň, T., Matas, J., Obdržálek, Š.: On evaluation of 6d object pose estimation. In: ECCV (2016) 10
22. Hodan, T., Michel, F., Brachmann, E., Kehl, W., GlentBuch, A., Kraft, D., Drost, B., Vidal, J., Ihrke, S., Zabulis, X., et al.: Bop: Benchmark for 6d object pose estimation. In: ECCV (2018) 2

23. Hodaň, T., Sundermeyer, M., Drost, B., Labbé, Y., Brachmann, E., Michel, F., Rother, C., Matas, J.: Bop challenge 2020 on 6d object localization. In: ECCV (2020) 2, 3, 4
24. James, S., Davison, A.J., Johns, E.: Transferring end-to-end visuomotor control from simulation to real world for a multi-stage task. In: CoRL (2017) 2
25. James, S., Wohlhart, P., Kalakrishnan, M., Kalashnikov, D., Irpan, A., Ibarz, J., Levine, S., Hadsell, R., Bousmalis, K.: Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks. In: CVPR (2019) 4
26. Kehl, W., Manhardt, F., Tombari, F., Ilic, S., Navab, N.: Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In: ICCV (2017) 4
27. Kleeberger, K., Huber, M.F.: Single shot 6d object pose estimation. In: ICRA (2020) 4
28. Kleeberger, K., Landgraf, C., Huber, M.F.: Large-scale 6d object pose estimation dataset for industrial bin-picking. In: IROS (2019) 1, 4
29. Li, X., Cao, R., Feng, Y., Chen, K., Yang, B., Fu, C.W., Li, Y., Dou, Q., Liu, Y.H., Heng, P.A.: A sim-to-real object recognition and localization framework for industrial robotic bin picking. RAL (2022) 1
30. Li, Z., Hu, Y., Salzmann, M., Ji, X.: Sd-pose: Semantic decomposition for cross-domain 6d object pose estimation. In: AAAI (2021) 2
31. Manhardt, F., Kehl, W., Navab, N., Tombari, F.: Deep model-based 6d pose refinement in rgb. In: ECCV (2018) 4, 7
32. Murez, Z., van As, T., Bartolozzi, J., Sinha, A., Badrinarayanan, V., Rabinovich, A.: Atlas: End-to-end 3d scene reconstruction from posed images. In: ECCV (2020) 2
33. Pastore, G., Cermelli, F., Xian, Y., Mancini, M., Akata, Z., Caputo, B.: A closer look at self-training for zero-label semantic segmentation. In: CVPR (2021) 2, 4, 12
34. Peng, S., Zhou, X., Liu, Y., Lin, H., Huang, Q., Bao, H.: Pvnet: pixel-wise voting network for 6dof object pose estimation. TPAMI (2020) 1, 3
35. Rizve, M.N., Duarte, K., Rawat, Y.S., Shah, M.: In defense of pseudo-labeling: An uncertainty-aware pseudo-label selection framework for semi-supervised learning. In: ICLR (2020) 5, 6
36. RoyChowdhury, A., Chakrabarty, P., Singh, A., Jin, S., Jiang, H., Cao, L., Learned-Miller, E.: Automatic adaptation of object detectors to new domains using self-training. In: CVPR (2019) 2, 4
37. Sundermeyer, M., Durner, M., Puang, E.Y., Marton, Z.C., Vaskevicius, N., Arras, K.O., Triebel, R.: Multi-path learning for object pose estimation across domains. In: CVPR (2020) 2, 3, 4, 10, 11, 12
38. Sundermeyer, M., Marton, Z.C., Durner, M., Brucker, M., Triebel, R.: Implicit 3d orientation learning for 6d object detection from rgb images. In: ECCV (2018) 4
39. Sundermeyer, M., Marton, Z.C., Durner, M., Triebel, R.: Augmented autoencoders: Implicit 3d orientation learning for 6d object detection. IJCV (2020) 2, 3, 4, 10, 11, 12
40. Thalhammer, S., Leitner, M., Patten, T., Vincze, M.: Pyrapose: Feature pyramids for fast and accurate object pose estimation under domain shift. In: ICRA (2021) 10
41. Tian, M., Ang, M.H., Lee, G.H.: Shape prior deformation for categorical 6d object pose and size estimation. In: ECCV (2020) 1
42. Tian, M., Pan, L., Ang, M.H., Lee, G.H.: Robust 6d object pose estimation by learning rgb-d features. In: ICRA (2020) 10, 11, 12

43. Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., Abbeel, P.: Domain randomization for transferring deep neural networks from simulation to the real world. In: IROS (2017) 2
44. Tremblay, J., To, T., Birchfield, S.: Falling things: A synthetic dataset for 3d object detection and pose estimation. In: CVPRW (2018) 4
45. Tremblay, J., To, T., Sundaralingam, B., Xiang, Y., Fox, D., Birchfield, S.: Deep object pose estimation for semantic robotic grasping of household objects. In: CoRL (2018) 2, 4
46. Tuzel, O., Liu, M.Y., Taguchi, Y., Raghunathan, A.: Learning to rank 3d features. In: ECCV (2014) 3
47. Veit, A., Alldrin, N., Chechik, G., Krasin, I., Gupta, A., Belongie, S.: Learning from noisy large-scale datasets with minimal supervision. In: CVPR (2017) 4
48. Wada, K., Sucar, E., James, S., Lenton, D., Davison, A.J.: Morefusion: Multi-object reasoning for 6d pose estimation from volumetric fusion. In: CVPR (2020) 1
49. Wada, K., James, S., Davison, A.J.: Reorientbot: Learning object reorientation for specific-posed placement. ICRA (2022) 1
50. Wada, K., James, S., Davison, A.J.: Safepicking: Learning safe object extraction via object-level mapping. ICRA (2022) 1
51. Wang, C., Xu, D., Zhu, Y., Martín-Martín, R., Lu, C., Fei-Fei, L., Savarese, S.: Densefusion: 6d object pose estimation by iterative dense fusion. In: CVPR (2019) 12, 14
52. Wang, G., Manhardt, F., Shao, J., Ji, X., Navab, N., Tombari, F.: Self6d: Self-supervised monocular 6d object pose estimation. In: ECCV (2020) 2, 4, 10, 12
53. Wang, H., Sridhar, S., Huang, J., Valentin, J., Song, S., Guibas, L.J.: Normalized object coordinate space for category-level 6d object pose and size estimation. In: CVPR (2019) 2
54. Wang, J., Chen, K., Dou, Q.: Category-level 6d object pose estimation via cascaded relation and recurrent reconstruction networks. arXiv:2108.08755 (2021) 14
55. Wang, X., Chen, H., Xiang, H., Lin, H., Lin, X., Heng, P.A.: Deep virtual adversarial self-training with consistency regularization for semi-supervised medical image classification. Medical Image Analysis (2021) 2
56. Xiang, Y., Schmidt, T., Narayanan, V., Fox, D.: Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. In: RSS (2018) 3
57. Xie, Q., Dai, Z., Hovy, E., Luong, T., Le, Q.: Unsupervised data augmentation for consistency training. NeurIPS (2020) 4
58. Xie, Q., Luong, M.T., Hovy, E., Le, Q.V.: Self-training with noisy student improves imagenet classification. In: CVPR (2020) 4, 6
59. Yalniz, I.Z., Jégou, H., Chen, K., Paluri, M., Mahajan, D.: Billion-scale semi-supervised learning for image classification. arXiv:1905.00546 (2019) 4
60. Yang, J., Gao, Y., Li, D., Waslander, S.L.: Robi: A multi-view dataset for reflective objects in robotic bin-picking. In: IROS (2021) 4, 9, 10
61. Yarowsky, D.: Unsupervised word sense disambiguation rivaling supervised methods. In: ACL (1995) 4
62. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: CVPR (2018) 8
63. Zhu, Y., Zhang, Z., Wu, C., Zhang, Z., He, T., Zhang, H., Manmatha, R., Li, M., Smola, A.: Improving semantic segmentation via self-training. arXiv:2004.14960 (2020) 4
64. Zou, Y., Yu, Z., Kumar, B., Wang, J.: Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In: ECCV (2018) 4