Annotation-Free and One-Shot Learning for Instance Segmentation of Homogeneous Object Clusters

Zheng Wu[1], Ruiheng Chang[1], Jiaxu Ma[1], Cewu Lu[1], Chi Keung Tang[2]
[1] Shanghai Jiao Tong University
[2] HKUST
{14wuzheng,crh19970307,alexma,lucewu}@sjtu.edu.cn, cktang@cse.ust.hk

Abstract

We propose a novel approach for instance segmentation given an image of homogeneous object cluster (HOC). Our learning approach is one-shot because a single video of an object instance is captured and it requires no human annotation. Our intuition is that images of homogeneous objects can be effectively synthesized based on structure and illumination priors derived from real images. A novel solver is proposed that iteratively maximizes our structured likelihood to generate realistic images of HOC. Illumination transformation scheme is applied to make the real and synthetic images share the same illumination condition. Extensive experiments and comparisons are performed to verify our method. We build a dataset consisting of pixel-level annotated images of HOC. The dataset and code will be released.

1 Introduction

Homogeneous object clusters (HOC) are ubiquitous. From microscopic cells to gigantic galaxies, they tend to cluster together. Figure 1 shows typical examples. Delineating individual homogeneous objects from their cluster gives an accurate estimate of the number of instances which enables many important applications: for example, in medicine, various blood cell counts give crucial information on a patient's health.

Instance segmentation for HOCs is by no means a trivial task despite the uniformity of the target objects. Directly applying current best performing instance segmentation methods [He *et al.*, 2017; Li *et al.*, 2017], many of which are based on some kind of Deep Convolutional Neural Network (DCNN), meets a bottleneck: unaffordable annotation cost. All of these segmentation models require a large number of annotated images for training purpose. Unlike typical object segmentation (e.g., car, chair), homogeneous clustered objects are densely distributed in a given image with various degrees of occlusion. Pixel-wise labeling these objects is extremely time consuming. However, in many realistic scenarios (e.g., merchandise sold in a supermarket), we have tens of thousands of categories to process. Category-specific annotation is impractical to the instance segmentation problem we address in this paper. We need to automatically generate

Figure 1: Typical homogeneous object clusters.

large amount training data (HOC images with segmentation annotation) with cheap cost.

Generative adversarial nets [Goodfellow *et al.*, 2014] has been widely used to generate images and is a seemingly promising direction. However, GAN framework cannot generate images with pixel-level annotation, so the segmentation models mentioned above cannot be trained using such images. RenderGAN [Sixt *et al.*, 2016] is proposed to generate images from labels. However the method also cannot generate images with pixel-level annotation. Image-to-image translation [Isola *et al.*, 2016] based on conditional GAN [Mirza and Osindero, 2014] can get the annotation from images but it requires a large collection of image-annotation pairs to train, and thus it still needs a lot of laborious annotation.

Driven by the above considerations, in this paper, we propose a novel framework to tackle the challenging instance segmentation problem. Inspired by [Fei-Fei *et al.*, 2006], our learning framework is **one-shot** because it learns by looking only once the single sample captured in a single short video, which avoids the cumbersome collection of large-scale image datasets for training. Then, these single-object video frames are used to automatically synthesize realistic images of homogeneous object clusters. In doing so, we can acquire a large amount of training data automatically. Therefore, our framework is **annotation-free**. However, generating visually realistic images is a non-trivial task, since structural con-
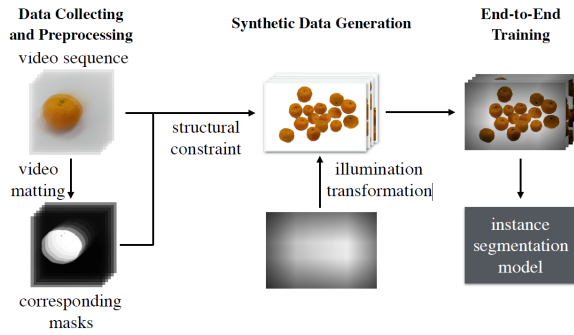
Figure 2: Overview of our pipeline. Our system (1) takes single-object video as input, extracts the mask for each frame, (2) generates synthetic images of homogeneous objects in cluster, then (3) uses the synthetic images to train an instance segmentation model.

straint (i.e., cluster layout should look reasonable) and illumination should be taken into consideration. In this paper, we propose a novel image synthesis framework to capture key priors from real images depicting HOCs. Structure prior is captured by learning a structured likelihood function. We generate structurally realistic synthetic images of HOC by placing synthetic objects in a specific way that maximizes the structured likelihood given by our defined function. Illumination is simulated through an efficient illumination transformation method we develop to transform both synthetic images and real images to share a similar illumination condition.

To benchmark our performance, a dataset is built that consists of 200 properly selected and annotated images of homogeneous clustered objects. The extensive experiments show that our approach significantly improves the segmentation performance over baseline methods on the proposed dataset.

The contributions of this paper are summarized as follows:

1. We propose an efficient framework for instance segmentation of HOCs. Our proposed method significantly reduces the cost of data collection and labeling.

2. We propose an efficient method to generate realistic synthetic training data which significantly improves instance segmentation performance.

3. We build a dataset consisting of HOC images. The dataset is used to evaluate our proposed method. The dataset and codes will be published with the paper.

2 Related Work

Instance segmentation has witnessed rapid progress recently in computer vision. A series of work reporting good performance has been proposed [He et al., 2017; Li et al., 2017], in which all of them use some kind of DCNN trained on large datasets [Lin et al., 2014; Cordts et al., 2016]. However, none of these current representative instance segmentation benchmarks has adequately addressed the task of segmenting HOCs at the instance level. Unlike the classical instance segmentation problem, the testing images for our problem have much more homogeneous instances on average, and exhibit a much higher degree of occlusion. Since currently all of the best models in performing instance segmentation require a large

volume of pixel-level annotated images during training, and that no public dataset is available for our specific task, the traditional training framework is inadequate to address our problem of HOC instance segmentation.

Our work is inspired by one-shot learning [Fei-Fei et al., 2006]. One-shot learning learns object categories from one training sample. Recently, it has been applied to solve image segmentation problem [Caelles et al., 2017; Rong and Yang, 2016]. [Caelles et al., 2017] use intrinsic continuity to segment objects in a video, but pixel-level annotation for the first frame is required. The work of [Rong and Yang, 2016] addresses the problem of segmenting gestures in video frames by learning a probability distribution vector (PDV) that describes object motions. However, these methods still require a number of images with pixel-level annotation.

In this paper we use single-object images to synthesize images of HOC as our training data. Recently, many studies have been done to use different methods to generate synthetic images to train a DCNN model. In [Papon and Schoeler, 2015] realistic synthetic scenes are generated by placing object models at random in a virtual room, and uses the synthetic data to train a DCNN. In the area of object detection, the authors in [Peng et al., 2015] propose to use 3D CAD models to generate synthetic training images automatically to train an object detector, while in 3D recognition, the work in [Su et al., 2015] synthesizes images by overlaying images rendered from 3D model on top of real images. Among these methods using synthetic data generation, [Papon and Schoeler, 2015] is most similar to ours (i.e., synthesizing images by placing objects on a background sequentially). While they simply place the synthesized objects randomly, we propose to learn how to place synthesized objects on the knowledge learned from realistic images of HOCs.

3 Our Method

3.1 Data Collection and Preprocessing

We aim to collect single-object images quickly and efficiently, so videos are the natural choice. We capture a short video for a single object, which is processed to extract individual frames alongside with the corresponding masks. The details are described in the following.

One-Shot Video Collection

Suppose we want to segment each orange from a cluster of oranges. A common solution is to collect a large number of images of orange clusters in different layouts, followed by annotating all of them and then training an instance segmentation model using the annotated images. In this paper, instead of adopting such conventional data collection, we perform the following: put one single orange at the center of a contrastive background, and take a video of the orange at different angles and positions. Such video typically lasts for about 20 seconds. It takes only a few minutes to acquire the data we require, which significantly reduces the cost of data collection compared to the previous methods. Since in our framework this single short video capturing one single object is all we need for learning, our framework may be regarded as a close kin to **one-shot learning** (i.e., learning by looking only once, one video in our case).

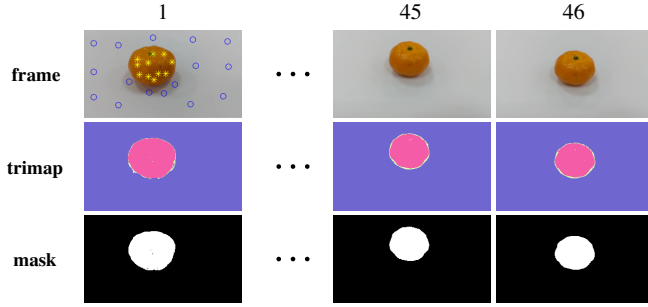|	1		45	46
frame		...		
trimap		...		
mask		...		

Figure 3: Example of our video processing method. For the first frame, seeds of foreground and background are automatically sampled based on color and location priors. Trimaps are interpolated across the video volume using optical flow. Matting technique uses the flowed trimaps to yield high-quality masks of the moving orange.

Video Matting

Our goal here is to automatically obtain the mask of each single-object image (i.e., the video frames) in an annotation-free manner. Since the foreground and background are controlled, and the image object is at the center, we apply color and location prior in the first frame to automatically sample seeds of the object and background respectively. Then we take the seeds as input and apply KNN matting [Chen *et al.*, 2013] on the video sequence to produce the mask of each frame. We take into consideration temporal coherence, and instead of producing a hand-drawn trimap (a pre-segmented image consisting of three regions: foreground, background and unknown) for every single frame, classical optical flow algorithm is applied for trimap generation and propagation. Figure 3 shows an example.

3.2 Synthetic Data Generation

Given the segmented single-object images from the previous stage, we generate synthetic images in realistic structure and illumination. Realistic images of HOC are generated in an iterative manner, followed by illumination transformation to make synthetic images and real images share a similar illumination condition.

Structural Constraint

Structured likelihood function The problem of generating realistic images of HOC deals with finding a solution to properly place N objects $\{O_1, O_2, ..., O_N\}$ sequentially in order to maximize its likelihood of being a real image of HOC. Let $\mathbf{I}_k$ denote the image we obtain after placing k objects $\{O_1, ..., O_k\}, 1 < k \leq N$, and $P(\cdot)$ denote the likelihood of being a real image. Our problem is equivalent to finding a solution to maximize $P(\mathbf{I}_N)$.

The image we get after placing k objects, $\mathbf{I}_k$, is determined by three factors: the images we get after placing $k-1$ objects $\mathbf{I}_{k-1}$, the k-th chosen object O_k and the way we place O_k in $\mathbf{I}_{k-1}$. We model our object placement by a four-parameter operation set $\Omega = \{\theta, \gamma, x, y\}$, in which θ denotes the rotation, γ denotes the resize factor, and x and y denote the coordinates of the center of O_k in $\mathbf{I}_{k-1}$. The relation between $\mathbf{I}_k$ and $\mathbf{I}_{k-1}, O_k, \Omega_k$ can be written as:

$$\mathbf{I}_k = g(\mathbf{I}_{k-1}, O_k, \Omega_k) \tag{1}$$

Given a sequence of $\{O_1, O_2, ..., O_N\}$, our goal is to find a $(\Omega_1, ..., \Omega_N)$ that maximizes $P(\mathbf{I}_N)$. Since $\{O_1, O_2, ..., O_N\}$ is given, finding a solution to maximize $P(\mathbf{I}_N)$ is equivalent to finding an optimal $(\Omega_1, ..., \Omega_N)$ that maximizes $P(\mathbf{I}_N)$. Since searching for an optimal $(\Omega_1, ..., \Omega_N)$ is intractable, we simplify the problem by applying greedy search algorithm, that is, in each iteration, the operation that maximizes $P(\mathbf{I}_k)$ is applied. Given $\mathbf{I}_{k-1}$ and O_k $(1 < k \leq N)$, we search for an optimal Ω_k that maximizes $P(\mathbf{I}_k)$:

$$\begin{aligned} \overline{\Omega}_k &= \arg \max_{\Omega_k \in \mathcal{D}} P(\mathbf{I}_k) \\ &= \arg \max_{\Omega_k \in \mathcal{D}} P(g(\mathbf{I}_{k-1}, O_k, \Omega_k)) \end{aligned} \tag{2}$$

where $\mathcal{D}$ is the feasible set of object placement. When $\mathbf{I}_{k-1}$ and O_k are given, $P(g(\mathbf{I}_{k-1}, O_k, \Omega_k))$ is the function of Ω_k. Let $f(\Omega_k) = P(g(\mathbf{I}_{k-1}, O_k, \Omega_k))$, then

$$\overline{\Omega}_k = \arg \max_{\Omega_k \in \mathcal{D}} f(\Omega_k; \mathbf{I}_{k-1}, O_k) \tag{3}$$

We aim to find an optimal solution $\overline{\Omega}_k$ that maximizes $f(\Omega_k)$. Since $f(\Omega_k; \mathbf{I}_{k-1}, O_k) = P(I_k)$, $f(\Omega_k; \mathbf{I}_{k-1}, O_k)$ represents the likelihood of $\mathbf{I}_k$ being a real image given $\mathbf{I}_{k-1}$ and O_k. We turn to DCNN to judge whether $\mathbf{I}_{k-1}$ is real or otherwise. In detail, we build the classifier by finetuning a pre-trained ImageNet model, where we use object proposals of real images as positive samples, and object proposals of randomly synthesized images (i.e., images synthesized by placing objects at random) as negative samples. We take the tight bounding box [Su *et al.*, 2012] of $\mathbf{I}_k$ as the input of the learnt classifier. The softmax output of the real class is taken as the output of $f(\Omega_k; \mathbf{I}_{k-1}, O_k)$.

Bayesian optimization framework Since $f(\Omega_k)$ is a black-box function (i.e., we do not have a specific expression of the function), we cannot optimize f by computing its derivative. We solve this problem by adopting bayesian optimization [Močkus, 1975]. Bayesian optimization is a powerful strategy for optimization of black-box function.

Here, we use bayesian optimization to optimize our continuous objective function $f(\Omega_k)$. Bayesian optimization behaves in an iterative manner. At iteration m, given the observed point set $D_{m-1} = \{(\Omega_k^1, f(\Omega_k^1)), ..., (\Omega_k^{m-1}, f(\Omega_k^{m-1}))\}$, we model a posterior function distribution by the observed points. Then, the acquisition function (i.e., a utility function constructed from the model posterior) is maximized to determine where to sample the next point $(\Omega_k^m, f(\Omega_k^m))$. $(\Omega_k^m, f(\Omega_k^m))$ is collected and the process is repeated. Iteration ends when $m = M$ (M is a user-defined parameter). For more detail on optimization framework please refer to [Močkus, 1975].

Overall algorithm We generate our realistic HOC images by placing objects sequentially in an iterative manner (see Figure 4). We start by directly placing one object on a white background in iteration 1. In iteration k $(1 < k \leq N)$, given $\mathbf{I}_{k-1}$ and O_k, bayesian optimization is applied to approximate an optimal $(\theta_k, \gamma_k, x_k, y_k)$ that maximizes the structured likelihood function $f(\Omega_k; \mathbf{I}_{k-1}, O_k)$. Iteration ends
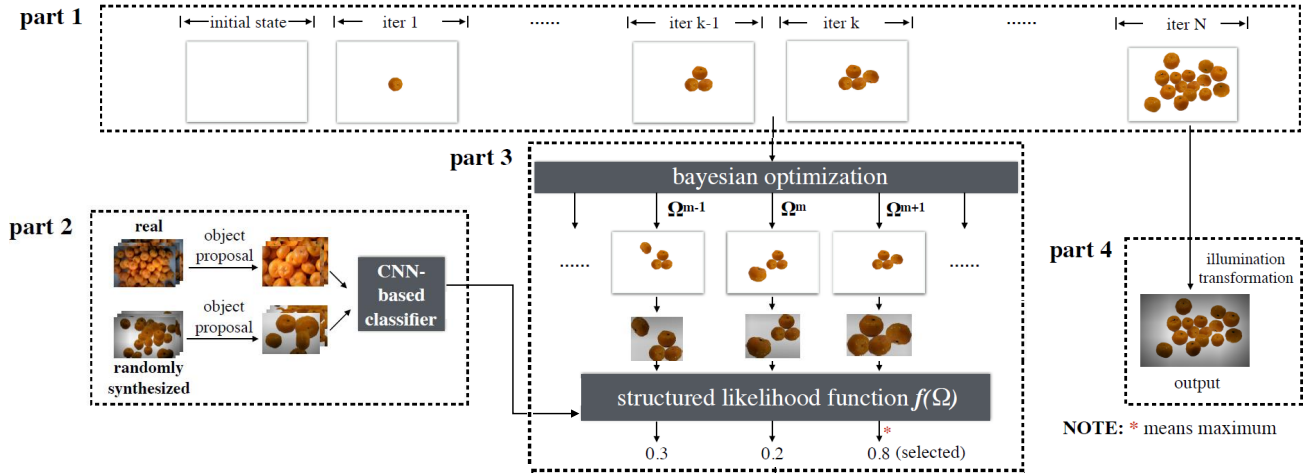
Figure 4: Synthetic images generation. N objects are selected and placed on a white background sequentially in an iterative manner (**part 1**). We start by placing one object directly on the background in iteration 1. For iteration $k, 1 < k \leq N$, we use bayesian optimization (**part 3**) to approximate an optimal placement that is guided by our structured likelihood function (**part 2**). During bayesian optimization process, M points $\{\Omega^1, ..., \Omega^M\}$ are sampled, while we only plot three due to space constraint. Each point is scored by $f(\Omega)$. The optimal placement is the point with the highest score. Then we apply illumination transformation (**part 4**) on the output of iteration N to obtain the final result.

when $k = N$. N is a prior given by user. User provides a range of how many instances per image in his (or her) case, thus, N is a random integral in this range. For example, in our dataset, N is a random integral between 10 and 30.

Illumination Transformation

To simulate lighting condition, we develop an efficient method to transform the synthetic and the real image so that they share a similar illumination condition.

We are inspired by [Leong *et al.*, 2003] where an uneven illumination correction method was proposed using Gaussian smoothing. We first convert both the synthetic image and the real image from RGB color space to HSV space, where V represents illumination information. Then, we implement detail removing, by using a large kernel Gaussian smoothing on both images to model general illumination condition. This general illumination condition (denoted as $blur(V_{real})$) is imposed on both real and synthetic images (after mean subtraction) to unify illumination.

$$V_{syn} = V_{syn} - \text{mean}(V_{syn}) + blur(V_{real}) \quad (4)$$

$$V_{real} = V_{real} - \text{mean}(V_{real}) + blur(V_{real}) \quad (5)$$

Here, V_{syn} and V_{real} respectively denote the V channel of the synthetic image and the real image, and mean$(\cdot)$ denotes the global mean value of 2-D matrix.

Finally, we convert the synthetic image and real image from HSV space back to RGB space. Figure 5 shows that our illumination transformation algorithm makes real and synthetic images share the similar illumination condition.

3.3 End-to-End Training

Finally, we use the synthetic images together with their corresponding annotations generated by the above method as our training data, and train an instance segmentation model in an end-to-end manner.
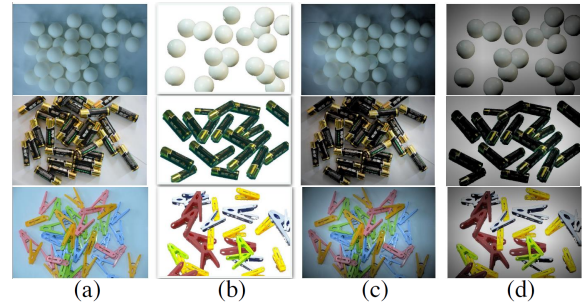

(a)　　(b)　　(c)　　(d)

Figure 5: Examples of our illumination transformation method. (a) real image before illumination transformation, (b) synthetic image before illumination transformation, (c) real image after illumination transformation, (d) synthetic image after illumination transformation.

In this paper, we adapt Mask R-CNN [He *et al.*, 2017] to handle our task. We train the model using our synthetic data. In testing phase, the model works on real image. Before the testing image is fed to the model, we adjust its illumination using Eq. 5 to ensure its illumination condition is similar to training images.

4 Experimental Evaluation

4.1 Dataset

Our task is totally new and none of current datasets is suitable for evaluating this problem. Our dataset is designed for benchmarking methods for instance segmentation of HOCs. Our dataset contains 10 types of objects; some are solid (*e.g.*, ping pong ball) while some are fluid (*e.g.*, bagged milk). For each type of object, 20 images are taken under different illumination conditions. We annotate all of the instances with

	badminton	battery	clothespin	grape	milk	hexagon nut	orange	ping pong	tissue	wing nut	mAP
Single	12.1	23.2	4.4	19.3	8.2	17.5	17.6	14.2	13.1	21.1	15.1
Random	40.6	50.9	38.4	50.8	26.7	52.9	63.8	67.2	83.9	32.9	50.8
Random+illumination	44.6	48.7	34.3	41.6	26.0	46.3	54.9	64.2	68.9	39.4	46.9
Random+structure	34.2	39.3	52.6	**72.7**	31.3	62.8	**90.3**	**81.7**	**90.7**	23.7	57.9
Ours	**53.0**	**69.5**	**67.7**	72.5	**52.6**	**73.6**	90.0	81.2	90.4	**48.4**	**69.9**

Table 1: Results on mAP^r@0.5 on our dataset. All numbers are percentages %.

	badminton	battery	clothespin	grape	milk	hexagon nut	orange	ping pong	tissue	wing nut	mAP
Single	8.7	21.2	2.0	16.8	5.0	15.2	16.0	11.8	9.9	18.8	12.5
Random	33.1	44.7	29.3	44.8	20.9	43.4	56.3	59.5	68.4	27.7	42.8
Random+illumination	36.5	43.2	28.4	37.2	20.8	38.3	50.4	56.4	56.8	35.5	40.4
Random+structure	29.8	36.5	36.7	**66.5**	22.0	45.2	83.8	**76.4**	80.4	20.7	49.8
Ours	**44.2**	**60.3**	**46.2**	65.4	**41.3**	**54.8**	**84.0**	75.6	80.4	**39.3**	**59.2**

Table 2: Results on mAP^r@[0.5:0.95] on our dataset. All numbers are percentages %.

less than 80% occlusion. Our dataset has $3,669$ instances in total, each image has 18.3 instances on average. We do not build an extremely larger dataset due to budget constraints, but it is sufficient to benchmark this problem.

4.2 Implementation Details

Structured likelihood function In the training step, we construct the likelihood function by finetuning the AlexNet [Krizhevsky *et al.*, 2012] pretrained model. For each type of object, we use [Zitnick and Dollár, 2014] to generate object proposals for both real images and randomly synthesized images (i.e., images synthesized by placing objects at random), and filter out proposals whose area are less than 0.01 of the original image. Then, we randomly choose 30,000 proposals from real images as positive examples, and 30,000 proposals from synthetic images as negative examples. We finetune the model for 30,000 iterations with a learning rate of 0.001, minibatch size of 32, momentum of 0.9 and weight decay of 0.01.

In the inference step, given $\mathbf{I}_{k-1}$, O_k and $(\theta_k, \gamma_k, x_k, y_k)$, we rotate O_k by θ_k degrees, resize it by a factor of γ_k, and place the center of O_k after rotation and resizing at location (x_k, y_k). When we place O_k, it can only be overlapped by previous objects. Then the tight bounding box of the new image we get after placing O_k is taken as input of the finetuned model. The softmax output of the real class is taken as the output of our structured likelihood function.

Instance segmentation model We adopt the Mask R-CNN [He *et al.*, 2017] as our instance segmentation model. We finetune the VGG-16 model [Simonyan and Zisserman, 2014] using the synthetic data generated by our method for 20,000 iterations with a learning rate of 0.002, minibatch size of 4, momentum of 0.9 and weight decay of 0.0001 on four Titan X GPU. Other settings are identical with [He *et al.*, 2017].

4.3 Results and Analysis

We follow the protocols used in [Hariharan *et al.*, 2014; He *et al.*, 2017; Li *et al.*, 2017] for evaluating our proposed method for solving the problem of instance segmentation of

HOCs. We evaluate using standard COCO evaluation metric, mAP^r@[0.5:0.95], as well as traditional mAP^r@0.5 metric. Table 1 and Table 2 respectively show the results on the mAP^r@0.5 and the mAP^r@[0.5:0.95] metrics on our dataset.

1. **Single** is baseline method, which uses outputs from Section 3.1 (i.e., images containing one single object and corresponding annotation) as training data to train Mask R-CNN [He *et al.*, 2017]. The baseline method reports mAP^r@0.5 of 15.1% and mAP^r@[0.5:0.95] of 12.5%. Since the training images of **Single** only contain a single object each, while testing realistic images contain multiple instances, the poor result is within our expectation.

2. **Random** is the result of training with randomly synthesized images (i.e., images synthesized by placing objects at random), which achieves mAP^r@0.5 of 50.8% and mAP^r@[0.5:0.95] of 42.8%. This is a large improvement over **Single**, although **Single** is not designed for the task, while the comparison is indicative it is somewhat unfair.

3. **Random+illumination** applies the illumination transformation method in Section 3.2 to transform synthetic images generated in **Random** and real images to share similar illumination condition. It uses synthetic images after illumination transformation as training data to train Mask R-CNN [He *et al.*, 2017] model. The result shows a slight decrease in performance using the mAP^r@0.5 metric (of about 3.9%) and mAP^r@[0.5:0.95] metric (of about 2.4%) over **Random**, indicating that illumination transformation does not help when training with randomly synthesized images.

4. **Random+structure** uses the method proposed in Section 3.2 to generate structurally realistic synthetic images *without* illumination transformation as training data. The result shows a 7.1% improvement on mAP^r@0.5 and a 7.0% improvement on mAP^r@[0.5:0.95] over **Random**. The significant improvement demonstrates the efficacy of our method which synthesizes realistic HOC images.

5. **Ours** is our method which trains the model using synthetic images generated under both the structural

constraint and with illumination transformation. Our method reports 12.0% higher in mAP^r@0.5 and 9.4% higher in mAP^r@[0.5:0.95] over **Random+structure**. While illumination transformation does not show effectiveness for training with randomly synthesized images, it indeed significantly improves performance for training with structurally realistic synthetic images.

Following the evaluation in [Hariharan *et al.*, 2014], Figure 6 plots the improvements on mAP^r over **Single** at 9 different IoU thresholds. As shown in Figure 6, the performance of our proposed method is superior to other methods at every threshold, which demonstrates the robustness of our algorithm. We also show some qualitative results in Figure 7, where we can observe that **Random** (trained with images synthesized by placing objects at random) can only segment objects that are not occluded by other objects, while our method can segment out a much larger number of partially occluded instances. This demonstrates that our image synthesis method is able to learn the occlusion pattern of real images. Besides, as shown in Table 1 and Table 2, the significant gap between **Random+structure** and **Ours** (12.0% in mAP^r@0.5 and 9.4% in mAP^r@[0.5:0.95]) indicates the effectiveness of our illumination transformation algorithm.

4.4 Comparison of Training with Existing Dataset

To further demonstrate the efficacy of our method which synthesizes HOC images is to compare our model with one that is trained on equal number of real HOC images with instance segmentation annotation. However, due to our limited budget, we cannot afford to build such a large dataset of HOC images with thorough annotation. Alternatively, we validate the effectiveness of our proposed method by comparing with state-of-the-art an instance segmentation model trained on existing dataset. Since our proposed dataset of HOCs and the COCO dataset have *orange* class in common, we use Mask R-CNN trained on COCO to directly test for object categories in common (i.e., *orange*) on our dataset. Mask R-CNN trained on COCO achieves 86.4% mAP^r@0.5 and 72.3% mAP^r@[0.5:0.95] on *orange* class in our dataset while our method achieves 90.0% mAP^r@0.5 and 84.0% mAP^r@[0.5:0.95], which significantly outperforms the base-

Figure 7: Qualitative segmentation results on our dataset. Each color denotes one instance. Our method can segment more instances than other baseline methods.
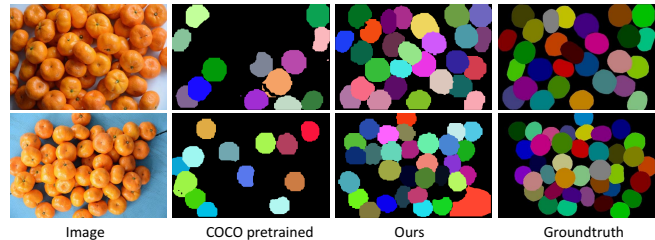
Figure 8: Comparison of COCO pre-trained model and our method. Our method is better at handling occlusion.

line method. The reason could be the baseline model does not train on large-scale HOC images, which are difficult to obtain due to expensive human annotation. By contrast, our proposed method can generate high-quality annotated HOC images with little effort which helps to achieve better performance. Figure 8 shows some qualitative results.

5 Conclusion and Future Work

Our work is the first attempt in tapping into learning how to segment homogeneous objects in clusters via single-object samples. The key to our proposed framework is an image synthesis technique based on the intuition that images of homogeneous clustered objects can be synthesized based on priors from real images. We build a dataset consisting of 200 carefully selected and annotated images of homogeneous object of clusters to evaluate our algorithm. While it indeed shows promises, the problem is far from solved. Our current framework may show poor performance on deformable objects (*e.g.*, human) since the collected single-object samples may not cover adequately all the possible appearances of the object, and training with synthetic images generated by these samples makes DCNN prone to overfitting. Nonetheless, given its odds in its present form, we believe this is an interesting work that will spawn future work and useful applications. We plan to produce a larger dataset in the future.
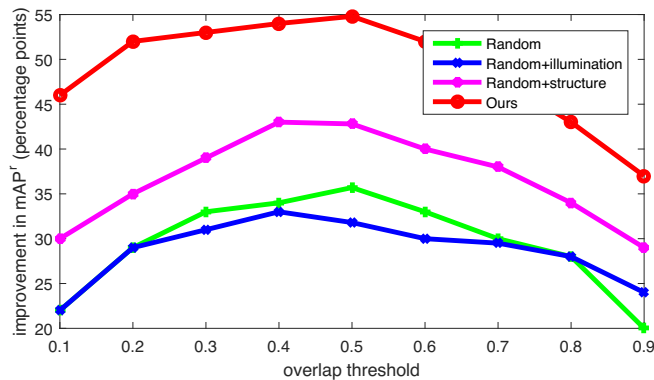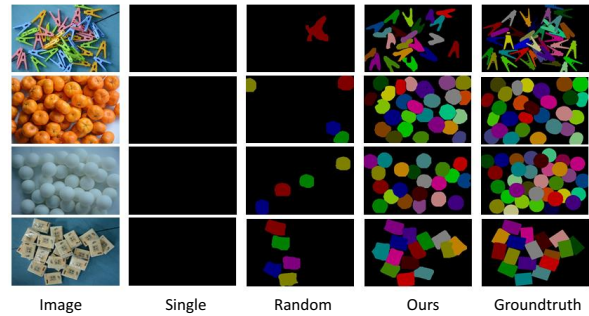

Figure 6: Improvement in mAP^r over **Single** for a variety of overlap thresholds. Our method reports improvements on all thresholds.

References

[Caelles *et al.*, 2017] Sergi Caelles, Kevis-Kokitsi Maninis, Jordi Pont-Tuset, Laura Leal-Taixé, Daniel Cremers, and Luc Van Gool. One-shot video object segmentation. In *CVPR 2017*. IEEE, 2017.

[Chen *et al.*, 2013] Qifeng Chen, Dingzeyu Li, and Chi-Keung Tang. Knn matting. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(9):2175–2188, Sept 2013.

[Cordts *et al.*, 2016] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[Fei-Fei *et al.*, 2006] Li Fei-Fei, Rob Fergus, and Pietro Perona. One-shot learning of object categories. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28:594–611, 2006.

[Goodfellow *et al.*, 2014] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[Hariharan *et al.*, 2014] Bharath Hariharan, Pablo Arbeláez, Ross Girshick, and Jitendra Malik. Simultaneous detection and segmentation. In *European Conference on Computer Vision*, pages 297–312, 2014.

[He *et al.*, 2017] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2980–2988. IEEE, 2017.

[Isola *et al.*, 2016] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint arXiv:1611.07004*, 2016.

[Krizhevsky *et al.*, 2012] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[Leong *et al.*, 2003] FJ WM Leong, M Brady, and JO McGee. Correction of uneven illumination (vignetting) in digital microscopy images. *Journal of clinical pathology*, 56(8):619–621, 2003.

[Li *et al.*, 2017] Yi Li, Haozhi Qi, Jifeng Dai, Xiangyang Ji, and Yichen Wei. Fully convolutional instance-aware semantic segmentation. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2359–2367, 2017.

[Lin *et al.*, 2014] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[Mirza and Osindero, 2014] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[Močkus, 1975] J Močkus. On bayesian methods for seeking the extremum. In *Optimization Techniques IFIP Technical Conference*, pages 400–404. Springer, 1975.

[Papon and Schoeler, 2015] Jeremie Papon and Markus Schoeler. Semantic pose using deep networks trained on synthetic rgb-d. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 774–782, 2015.

[Peng *et al.*, 2015] Xingchao Peng, Baochen Sun, Karim Ali, and Kate Saenko. Learning deep object detectors from 3d models. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1278–1286, 2015.

[Rong and Yang, 2016] Tao Rong and Ruoyu Yang. One-shot-learning gesture segmentation and recognition using frame-based pdv features. In *Pacific Rim Conference on Multimedia*, pages 355–365. Springer, 2016.

[Simonyan and Zisserman, 2014] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[Sixt *et al.*, 2016] Leon Sixt, Benjamin Wild, and Tim Landgraf. Rendergan: Generating realistic labeled data. *arXiv preprint arXiv:1611.01331*, 2016.

[Su *et al.*, 2012] Hao Su, Jia Deng, and Li Fei-Fei. Crowdsourcing annotations for visual object detection. In *Workshops at the Twenty-Sixth AAAI Conference on Artificial Intelligence*, volume 1, 2012.

[Su *et al.*, 2015] Hao Su, Charles R Qi, Yangyan Li, and Leonidas J Guibas. Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2686–2694, 2015.

[Zitnick and Dollár, 2014] C. Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In *ECCV*, 2014.