OCN: LEARNING OBJECT-CENTRIC REPRESENTATIONS FOR UNSUPERVISED MULTI-OBJECT SEGMENTATION

Anonymous authors
Paper under double-blind review

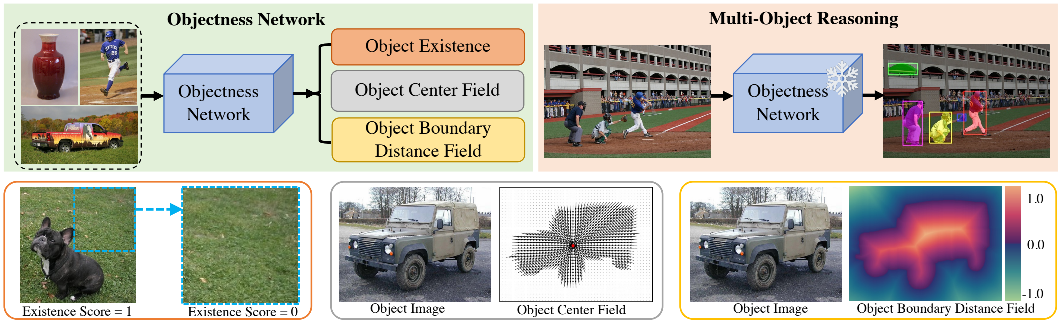
ABSTRACT

We study the challenging problem of unsupervised multi-object segmentation on single images. By relying on an image reconstruction objective to learn objectness or leveraging pretrained image features to group similar pixels as objects, existing methods can either segment simple synthetic objects or discover a rather limited number of real-world objects. In this paper, we introduce **OCN**, a new two stage pipeline to discover many complex objects on real-world images. The key to our approach is to explicitly learn our carefully defined three level object-centric representations in the first stage. After that, our multi-object reasoning module directly leverages the learned object priors to discover multiple objects in the second stage. Such a reasoning module is completely network-free and does not need any human labels to train. Extensive experiments show that our OCN clearly surpasses all existing unsupervised methods by a large margin on 7 real-world benchmark datasets including the particularly challenging COCO dataset, achieving the state-of-the-art object segmentation results. Most notably, our method demonstrates superior results on extremely crowded images where all baselines collapse.

1 INTRODUCTION

By age two, humans can learn around 300 object categories and recognize multiple objects in unseen scenarios (Frank et al., 2016). For example, after reading a book of Animal Kingdom where each page illustrates a single creature, children can effortlessly recognize multiple similar animals at a glance when visiting a zoo without needing extra teaching on site. Inspired by such an efficient skill of perceiving objects and scenes, we aim to introduce a new framework to identify multiple objects from single images just by learning object-centric representations, instead of relying on costly scene-level human annotations for supervision.

Existing works for unsupervised multi-object segmentation mainly consist of two categories: 1) Slot-based methods represented by SlotAtt (Locatello et al., 2020) and its variants (Sajjadi et al., 2022; Didolkar et al., 2024). They usually rely on an image reconstruction objective to drive the slot-structured bottlenecks to learn object representations. While achieving successful results on synthetic datasets (Karazija et al., 2021; Greff et al., 2022), they often fail to scale to complex real-world images. 2) Self-supervised feature distillation based methods such as TokenCut (Wang et al., 2022b), DINOSAUR (Seitzer et al., 2023), CutLER (Wang et al., 2023a), and CuVLER (Arica et al., 2024). Thanks to the strong object localization hints emerging from self-supervised pretrained features such as DINO/v2 (Caron et al., 2021; Oquab et al., 2023), these methods explore such a property to discover multiple objects via feature reconstruction or pseudo mask creation for supervision. Despite obtaining very promising segmentation results on real-world datasets such as COCO (Lin et al., 2014), they still fail to discover a satisfactory number of objects. Primarily, this is because the simple feature reconstruction or pseudo mask creation for supervision tends to distill or define rather weak objectness followed by ineffective object search, resulting in only a few objects correctly discovered. In fact, unsupervised multi-object segmentation of a single image is hard and not straightforward, as it involves two critical issues: 1) the definition of *what objects are (i.e., objectness)* is unclear, 2) there is a lack of an effective way to discover *those objects* at unseen scenes.



(a) Object Existence (b) Object Center Field (c) Object Boundary Distance Field

Figure 1: The upper two blocks illustrate our overall framework, whereas the lower three blocks show our three level object-centric representations.

In this paper, to tackle these issues, we propose a two-stage pipeline consisting of an object-centric representation learning stage followed by an effective multi-object reasoning stage, akin to human's innate skill of perceiving objects and scenes. As illustrated in the upper left block of Figure 1, in the first stage, we aim to train an **objectness network** to learn our explicitly defined object-centric representations from monolithic object images such as ImageNet. In the second stage as illustrated in the right block of Figure 1, we introduce a **multi-object reasoning module** to automatically discover individual objects on single images just by leveraging our pretrained and frozen objectness network, instead of requiring any human annotations for supervision.

Figure 2: Object images.

Regarding the **objectness network**, our key insight is that, given an input image or patch, it should be able to answer three essential questions: 1) is there an object inside (*i.e.*, *object existence*)? 2) if so, where is it (*i.e.*, *object location/center*)? and 3) what is the object shape (*i.e.*, *object boundary*)? Basically, training such an objectness network would be analogous to the learning process of infants to form concepts of objects in mind. As shown in Figure 2, we can easily see that there is no salient object in image #1, but images #2/#3 have a similar dog at different locations, whereas image #4 has another object with different shape boundaries. By training on such images, our objectness network aims to explicitly capture these top-down (existence/location) and bottom-up (boundary) object-centric representations. To achieve this goal, we introduce the corresponding three levels of objectness to learn in parallel: 1) a binary object existence score, 2) an object center field, and 3) an object boundary distance field, as illustrated in Figure 1.

With respect to the **multi-object reasoning module**, we aim to discover as many individual objects as possible on scene-level images. Our insight is that, given a multi-object image, if a randomly cropped patch happens to include a single valid object inside, its three levels of objectness representations must satisfy a certain threshold when querying against our pretrained objectness network. Otherwise, that patch should be discarded or its position and size should be effectively updated until a valid object is found. To this end, we introduce a center-boundary-aware reasoning algorithm to iteratively regress accurate multi-object bounding boxes and masks according to the learned three levels of object-centric representations from our pretrained objectness network. Notably, our algorithm has two nice properties: 1) it requires no human labels and the reasoning module is completely network-free; 2) albeit designed in a heuristic way, it explicitly exploits the mutual dependencies between three level object-centric representations, thus being effective to discover multiple objects.

Our framework, named **OCN**, learns **o**bject-**c**entric representations via the objectness **n**etwork, enabling us to directly identify multiple objects on single images. Our contributions are:

- We introduce a new pipeline comprising object-centric learning and multi-object reasoning, and propose three levels of explicit object-centric representations including object existence, object center field, and object boundary distance field learned by an objectness network.
- We design a center-boundary aware reasoning algorithm to iteratively discover multiple objects on single images. The algorithm is network-free and does not require any human labels to supervise.
- We demonstrate superior object segmentation results and clearly surpass the state-of-the-art unsupervised methods on 7 benchmark datasets including the challenging COCO (Lin et al., 2014).

2 RELATED WORK

Object-centric Learning without Pretrained Features: Object-centric learning involves the unsupervised discovery of multiple objects in a scene. A plethora of methods have been proposed in past years (Yuan et al., 2023). They primarily rely on an image reconstruction objective to learn objectness from scratch without needing any human labels or pretrained image features. Early models aim to learn object factors such as size, position, and appearance from raw images by training (variational) autoencoders (AE/VAE) (Kingma & Welling, 2014), including AIR (Eslami et al., 2016), SPACE (Lin et al., 2020) and others (Greff et al., 2016; 2017; Crawford & Pineau, 2019; Burgess et al., 2019; Greff et al., 2019). Recently, with the success of slot based methods (Locatello et al., 2020; Engelcke et al., 2020), most succeeding works (Engelcke et al., 2021; Sajjadi et al., 2022; Löwe et al., 2022; Biza et al., 2023; Löwe et al., 2023; Foo et al., 2023; Brady et al., 2023; Jia et al., 2023; Stanić et al., 2023; Lachapelle et al., 2023; Kirilenko et al., 2024; Gopalakrishnan et al., 2024; Wiedemer et al., 2024; Didolkar et al., 2024; Mansouri et al., 2024; Kori et al., 2024a;b; Jung et al., 2024; Fan et al., 2024) extend the slot structure from various aspects to improve the object segmentation performance. Although achieving excellent results, they often fail to scale to complex real-world images as investigated in (Yang & Yang, 2022). To overcome this limitation, a line of works (Weis et al., 2021) use additional information such as motion and depth as grouping signals to identify objects. Unfortunately, this precludes learning on most real-world images which do not have motion or depth information.

Object-centric Learning with Pretrained Features: Very recently, with the advancement of self-supervised learning techniques, strong object semantic and localization hints emerge from these features like DINO/v2 (Caron et al., 2021; Oquab et al., 2023) pretrained on ImageNet (Deng et al., 2009) without any annotation. An increasing number of methods leverage such features for unsupervised salient/single object detection (Voynov et al., 2021; Shin et al., 2022; Tian et al., 2024) or multi-object segmentation (Siméoni et al., 2024), or video object segmentation (Aydemir et al., 2023; Zadaianchuk et al., 2024). Representative works include the early LOST (Siméoni et al., 2021), ODIN (Hénaff et al., 2022), TokenCut (Wang et al., 2022b), and the recent DINOSAUR (Seitzer et al., 2023), CutLER (Wang et al., 2023a), and UnSAM (Wang et al., 2024). These methods and their variants (Wang et al., 2022a; Singh et al., 2022; Ishtiak et al., 2023; Wang et al., 2023c;b; Niu et al., 2024; Zhang et al., 2024) achieve very promising object segmentation results on challenging real-world datasets, demonstrating the value of pretrained features. However, they still fail to discover a satisfactory number of objects and the estimated object bounding boxes and masks often suffer from under-segmentation issues. Essentially, this is because these methods tend to simply group pixels with similar features (obtained from pretrained models) as a single object, lacking the ability to discern boundaries between objects. As a consequence, for example, they usually group two chairs nearby into just one object. By contrast, our introduced three level object-centric representations are designed to jointly retain unique and explicit objectness features for each pixel, *i.e.*, how far away to the object boundary and in what direction to the object center.

3 OCN

3.1 PRELIMINARY

The core of our method is the objectness net, and we aim to learn three levels of object-centric representations from the large-scale ImageNet dataset. Thanks to the advanced self-supervised learning techniques which give us semantic and location information of objects in pretrained models, we opt to use pretrained features to extract object regions on ImageNet to bootstrap our objectness network.

In particular, we exactly follow the VoteCut method proposed in CuVLER (Arica et al., 2024) to obtain a single object mask (binary) on each image of ImageNet. First, each image of ImageNet is fed into self-supervised pretrained DINO/v2, obtaining patch features. Second, An affinity matrix is constructed based on the similarity of patch features, followed by Normalized Cut (Shi & Malik, 2000) to obtain multiple object masks. Third, the most salient mask of each image is selected as the rough foreground object. For more details, refer to CuVLER. These rough masks will be used to learn our object-centric representations in Section 3.2.

3.2 OBJECTNESS NETWORK

With single object images and the prepared (rough) masks on ImageNet (the object image denoted as $\boldsymbol{I} \in \mathcal{R}^{H\times W\times 3}$, object mask as $\boldsymbol{M} \in \mathcal{R}^{H\times W\times 1}$), the key to train our objectness network is the definitions of three levels of object-centric representations which are elaborated as follows.

Object Existence Score: For an image $\boldsymbol{I}$, its object existence score f^e is simply defined as 1 (positive sample) if it has a valid object, *i.e.*, $sum(\boldsymbol{M}) >= 1$, and 0 otherwise (negative sample). In the preliminary stage of processing ImageNet, since every image has a valid object, we then create a twin negative sample by cropping the largest rectangle on background pixels excluding the tightest object bounding box. As illustrated in Figure 1 (a), image #1 is an original sample from ImageNet, whereas image #2 is a twin negative sample created by us.

Object Center Field: For an image $\boldsymbol{I}$ with a valid object mask $\boldsymbol{M}$ inside, its object center field $\boldsymbol{f}^c$ is designed to indicate the position/center of the object, *i.e.*, the tightest object bounding box center. As illustrated in Figure 1(b), each pixel within the object mask is assigned a unit vector pointing to the object center $[C_h, C_w]$, and the pixel outside mask is assigned as a zero vector. Formally, the center field value at the $(h, w)^{th}$ pixel, denoted as $\boldsymbol{f}^c_{(h,w)}$, is defined as follows. Basically, this center field aims to capture the relative position of an object with respect to pixels of an image.

$$\boldsymbol{f}^c_{(h,w)} = \begin{cases} \frac{[h,w]-[C_h,C_w]}{\|[h,w]-[C_h,C_w]\|}, & \text{if } \boldsymbol{M}_{(h,w)} = 1 \\ [0, 0], & \text{otherwise} \end{cases} \quad \text{and} \quad \boldsymbol{f}^c \in \mathcal{R}^{H\times W\times 2} \tag{1}$$

We notice that prior works (Gall & Lempitsky, 2009; Gall et al., 2011; Qi et al., 2019) use Hough Transform to transform pixels/points to object centroids for 2D/3D object detection, which requires to learn both directions and distances to object centers. However, our object center field is just defined as unit directions pointing to object centers, as we only need to learn such directions to identify multi-center proposals instead of recovering object masks as detailed in Step #2 of Section 3.3.

Object Boundary Distance Field: For the same image $\boldsymbol{I}$ and its object mask $\boldsymbol{M}$, this boundary distance field $\boldsymbol{f}^b$ is designed to indicate the shortest distance from each pixel to the object boundary. To discriminate a pixel being inside or outside of an object, we first compute the simple signed distance field, where the distance values inside the object mask are assigned to be positive, outside negative, and boundary pixels zeros. This signed distance field is denoted as $\boldsymbol{S} \in \mathcal{R}^{H\times W\times 1}$ for the whole image, and its value at the $(h, w)^{th}$ pixel $S_{(h,w)}$ is calculated as follows:

$$S_{(h,w)} = \begin{cases} \|[h, w] - [\bar{h}, \bar{w}]\|, & \text{if } \boldsymbol{M}_{(h,w)} = 1 \\ -\|[h, w] - [\bar{h}, \bar{w}]\|, & \text{otherwise} \end{cases} \tag{2}$$

where the location $(\bar{h}, \bar{w})$ is the nearest pixel position on the object boundary corresponding to the pixel (h, w). Detailed steps of calculation are in Appendix A.1. These signed distance values are measured by the number of pixels and could vary significantly across images with differently-sized objects. Notably, the maximum signed distance value within an object mask $\boldsymbol{M}$, assuming appearing at the $(\hat{h}, \hat{w})^{th}$ pixel location, *i.e.*, $S_{(\hat{h},\hat{w})} = max(\boldsymbol{S} * \boldsymbol{M})$, actually indicates the object size. The higher $S_{(\hat{h},\hat{w})}$, the likely the object is larger or its innermost pixel is further away from the boundary.

To stabilize the training process, we opt to normalize signed distance values as our object boundary distances. Particularly, we normalize the foreground and background signed distances separately. For the $(h, w)^{th}$ pixel, our object boundary distance field, denoted as $\boldsymbol{f}^b_{(h,w)}$, is defined as follows:

$$\boldsymbol{f}^b_{(h,w)} = \begin{cases} \frac{S_{(h,w)}}{max(\boldsymbol{S}*\boldsymbol{M})}, & \text{if } \boldsymbol{M}_{(h,w)} = 1 \\ \frac{S_{(h,w)}}{min\left(\boldsymbol{S}*(\boldsymbol{1}-\boldsymbol{M})\right)}, & \text{otherwise} \end{cases} \quad \text{and} \quad \boldsymbol{f}^b \in \mathcal{R}^{H\times W\times 1} \tag{3}$$

where * represents element-wise multiplication. Figure 1(c) shows an example of an object image and its final boundary distance field. Our above definition of boundary distance field has a nice property that the maximum signed distance value $S_{(\hat{h},\hat{w})}$ can be easily recovered based on the norm of the gradient of $\boldsymbol{f}^b$ at any pixel inside of object as follows. This property is crucial to quickly search object boundaries at the stage of multi-object reasoning as discussed in Section 3.3.

$$S_{(\hat{h},\hat{w})} = 1/\|\frac{\partial \boldsymbol{f}^b_{(h,w)}}{\partial h}, \frac{\partial \boldsymbol{f}^b_{(h,w)}}{\partial w}\|, \qquad \text{if } \boldsymbol{f}^b_{(h,w)} > 0 \tag{4}$$

Notably, the concept of boundary distance field is successfully used for shape reconstruction (Park et al., 2019; Xie et al., 2022). Here, we demonstrate its effectiveness for object discovery.

Overall, for all original images of ImageNet, three levels of object-centric representations are clearly defined based on the generated rough object masks in our preliminary stage. We also create twin negative images with zero existence scores.

Objectness Network Architecture and Training: Having the defined representations on images, we just choose two commonly-used existing networks in parallel as our objectness network, particularly, using ResNet50 (He et al., 2016) as a binary classifier to predict *object existence scores* $\tilde{f}^e$, using DPT-large (Ranftl et al., 2021) followed by two CNN-based heads to predict *object center field* $\tilde{\boldsymbol{f}}^c$ and *object boundary distance field* $\tilde{\boldsymbol{f}}^b$ respectively. To train the whole model, the cross-entropy loss is applied for learning existence scores, L2 loss for the center field, and L1 for the boundary distance field. Our total loss is defined as follows and more details are provided in Appendix A.2.

$$\ell = CE(\tilde{f}^e, f^e) + \ell_2(\tilde{\boldsymbol{f}}^c, \boldsymbol{f}^c) + \ell_1(\tilde{\boldsymbol{f}}^b, \boldsymbol{f}^b) \tag{5}$$

3.3 Multi-Object Reasoning Module

With the objectness network well-trained on ImageNet, our ultimate goal is to identify as many objects as possible on complex scene images without needing human labels for supervision. Given a single scene image, a naïve solution is to endlessly crop many patches with different resolutions at different locations, and then feed them into our pretrained objectness network to verify each patch's objectness. Apparently, this is inefficient and infeasible in practice. To this end, we introduce a network-free multi-object reasoning module consisting of the following steps.

Step #0 - Initial Object Proposal Generation: Given a scene image $\mathcal{I} \in \mathcal{R}^{M\times N\times 3}$, we randomly and uniformly initialize a total of T bounding box proposals by selecting a set of anchor pixels on the entire image. At each anchor pixel, multiple sizes and aspect ratios are chosen to create initial bounding boxes. More details are provided in Appendix A.3. For each proposal P, its top-left and bottom-right corner positions at the original scene image will always be tracked and denoted as $[P^{u_1}, P^{v_1}, P^{u_2}, P^{v_2}]$. We also linearly scale up or down all proposals to be the same resolution of 128×128 to feed into our objectness network subsequently.

Step #1 - Existence Checking: For each bounding box proposal P, we feed the corresponding image patch (cropped from $\mathcal{I}$) into our pretrained and frozen objectness network, obtaining its existence score f^e_p. The proposal will be discarded if f^e_p is smaller than a threshold τ^e. The higher the τ^e predefined, the more aggressive it is to ignore potential objects.

Step #2 - Center Reasoning: For the proposal P with a higher enough object existence score, we then obtain its center field $\boldsymbol{f}^c_p$ from our objectness network. This step #2 aims to evaluate whether $\boldsymbol{f}^c_p$ has only one center or ≥ 2 centers. If there is just one center, the non-zero center field vectors of $\boldsymbol{f}^c_p$ are likely pointing to a common position. Otherwise, those vectors are likely pointing to multi-positions. In the latter case, the proposal P needs to be safely split into subproposals at pixels whose center field vectors are facing opposite directions. Thanks to this nice property, we propose the following simple kernel-based operation for multi-center detection and proposal splitting.

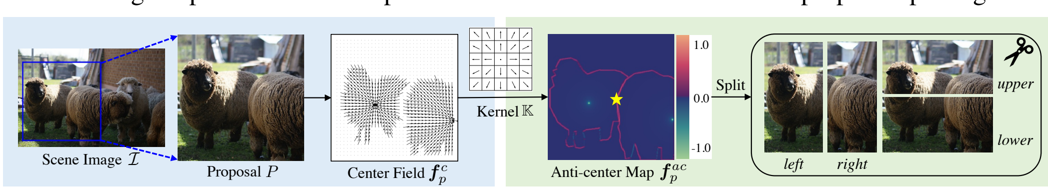


Figure 3: An illustration of kernel-based operation for multi-center detection and proposal splitting.

As shown in the left block of Figure 3, given the center field $\boldsymbol{f}^c_p \in \mathcal{R}^{128\times 128\times 2}$ of a proposal P, we predefine a kernel $\mathbb{K} \in \mathbf{R}^{5\times 5\times 2}$ where each of the (5×5) vectors has a unit length and points outward against the kernel center. Details of kernel values are in Appendix A.3. By applying this kernel on top of $\boldsymbol{f}^c_p$ with a stride of 1×1 and zero-paddings, we obtain an anti-center map, denoted as $\boldsymbol{f}^{ac}_p \in \mathcal{R}^{128\times 128\times 1}$. The higher the anti-center value at a specific pixel, the more likely that pixel is in between multiple crowded objects. Otherwise, that pixel is more likely near an object center or belongs to the background. Clearly, the former case is more likely to incur under-segmentation.

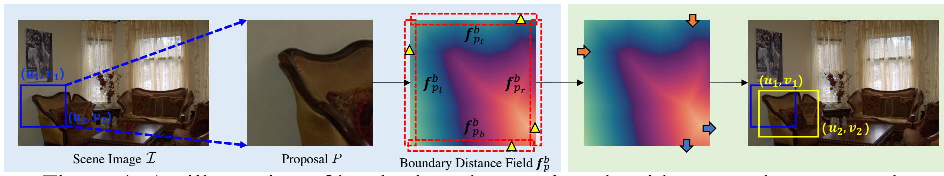


Figure 4: An illustration of border-based reasoning algorithm to update proposals.

For this anti-center map $\boldsymbol{f}_p^{ac}$ of the proposal P, 1) if its highest value among all pixels is greater than a threshold τ^c, this proposal P is likely to have ≥ 2 crowded objects and will be split at the corresponding pixel location with the highest value. As shown in the right block of Figure 3, we safely split the proposal P into 4 subproposals at the highest anti-center value (yellow star): $\{left, right, upper, lower\}$ halves. Each subproposal is regarded as a brand-new one and will be evaluated from **Step #1** again. With this design, the particularly challenging under-segmentation issue often incurred by multiple crowded objects can be naturally solved.

2) If the highest value of $\boldsymbol{f}_p^{ac}$ is smaller than the threshold τ^c, the proposal P is likely to have just one object, or multiple objects but they are far away from each other, *i.e.*, more than 5 pixels apart. In this regard, we simply adopt the connected-component method used in CuVLER (Arica et al., 2024) to split the proposal P into subproposals. Particularly, for its center field $\boldsymbol{f}_p^c$, all pixels that are spatially connected and have non-zero unit vectors are grouped into one subproposal. Each subproposal is regarded as a brand-new one and will be evaluated from **Step #1** again.

Step #3 - Boundary Reasoning: At this step, the proposal P is likely to have a single object and we obtain its boundary distance field $\boldsymbol{f}_p^b$ from our objectness network. The ultimate goal of this step is to correctly update this proposal's location and size, *i.e.*, the two corner positions $[P^{u_1}, P^{v_1}, P^{u_2}, P^{v_2}]$ at its original scene image $\mathcal{I}$, such that the proposal could converge to a tight bounding box of the object inside. Recall that, in Equations 3&4, our definition of boundary distance field and its gradient have a crucial property. Particularly, the value at a specific pixel of the boundary distance field $\boldsymbol{f}_p^b$ indicates how far away from the nearest object's boundaries. This means that we can directly use $\boldsymbol{f}_p^b$ to help update the two corner positions.

Intuitively, if the proposal P has an incomplete object, its borders need to expand. If it has many background pixels, its borders need to contract. With this insight, we only need to focus on boundary distance values of the four borders of $\boldsymbol{f}_p^b$ to decide the margins to expand or contract. To this end, we introduce the following border-based reasoning algorithm to update $[P^{u_1}, P^{v_1}, P^{u_2}, P^{v_2}]$.

As illustrated in Figure 4, for the boundary distance field $\boldsymbol{f}_p^b \in \mathcal{R}^{128\times128\times1}$ of a proposal P, we first collect values at four boarders {*topmost row, leftmost column, bottommost row, rightmost column*} highlighted by red dotted lines, denoted by four vectors: $\{\boldsymbol{f}_{p_t}^b, \boldsymbol{f}_{p_l}^b, \boldsymbol{f}_{p_b}^b, \boldsymbol{f}_{p_r}^b\} \in \mathcal{R}^{128}$. Each of the four borders of proposal P is designed to update as follows:

$$P^{u_1} \leftarrow P^{u_1} - \frac{max(\boldsymbol{f}_{p_t}^b)}{\|\frac{\partial \boldsymbol{f}_{p_t}^b}{\partial u}, \frac{\partial \boldsymbol{f}_{p_t}^b}{\partial v}\|}, (u,v) = argmax \boldsymbol{f}_{p_t}^b; P^{v_1} \leftarrow P^{v_1} - \frac{max(\boldsymbol{f}_{p_l}^b)}{\|\frac{\partial \boldsymbol{f}_{p_l}^b}{\partial u}, \frac{\partial \boldsymbol{f}_{p_l}^b}{\partial v}\|}, (u,v) = argmax \boldsymbol{f}_{p_l}^b \quad (6)$$
$$P^{u_2} \leftarrow P^{u_2} + \frac{max(\boldsymbol{f}_{p_b}^b)}{\|\frac{\partial \boldsymbol{f}_{p_b}^b}{\partial u}, \frac{\partial \boldsymbol{f}_{p_b}^b}{\partial v}\|}, (u,v) = argmax \boldsymbol{f}_{p_b}^b; P^{v_2} \leftarrow P^{v_2} + \frac{max(\boldsymbol{f}_{p_r}^b)}{\|\frac{\partial \boldsymbol{f}_{p_r}^b}{\partial u}, \frac{\partial \boldsymbol{f}_{p_r}^b}{\partial v}\|}, (u,v) = argmax \boldsymbol{f}_{p_r}^b$$

Because $\{max(\boldsymbol{f}_{p_t}^b), max(\boldsymbol{f}_{p_l}^b), max(\boldsymbol{f}_{p_b}^b), max(\boldsymbol{f}_{p_r}^b)\}$ could be positive or negative, making the four borders of the proposal P to expand or contract by itself. As shown in rightmost block of Figure 4, the proposal P is updated from the blue rectangle to the yellow one whose bottom and right borders expand to include more object parts because their maximum boundary distance values are positive, whereas its top and left borders contract to exclude more background pixels because their maximum boundary distance values are negative. As boundary distance values are physically meaningful, each expansion step will not go far outside of the tightest bounding box and each contraction step will not step deep into the tightest bounding box.

Among the total four steps, the center-boundary-aware reasoning **Steps #2/#3** are crucial and complementary to tackle the core under-/over-segmentation issues. Once the two corners of a proposal P are updated, we will feed the updated proposal into **Step #3** until the corner converges to stable values. During this iterative updating stage, we empirically find that it is more efficient to take a slightly larger step size for expansion, a smaller step size for contraction. More details are in Appendix A.3.

Once the size and location of a proposal P converge, a valid object is discovered. After all proposals are processed in parallel through **Steps #1/#2/#3**, we collect all bounding boxes and apply the standard NMS to filter out duplicated detections. For each final bounding box, we obtain its object mask by taking the union of positive values within its boundary distance field and non-zero vectors within its center field. We also compute a confidence score for each object based on its object existence score, center field, and boundary distance field. More details are in Appendix A.4.

Optionally Training a Detector: As shown in CutLER (Wang et al., 2023a) and CuVLER (Arica et al., 2024), the discovered objects from scene images can be used as pseudo labels to train a separate detector from scratch. We select and weight each discovered object based on its confidence score. Intuitively, the selected objects should have high object existence scores, homogeneous center fields and boundary fields. More details about the pseudo label selection and processing are provided in Appendix A.5. Lastly, following CuVLER (Arica et al., 2024), we train the same class agnostic detector using the same training strategy based on our pseudo labels from scratch.

4 EXPERIMENTS

Datasets: Evaluation of existing unsupervised multi-object segmentation methods is primarily conducted on the challenging COCO validation set (Lin et al., 2014). However, we empirically find that a large number of objects are actually not annotated in validation set. This may not be an issue for evaluating fully-supervised methods in literature, but likely gives inaccurate evaluation of unsupervised object discovery. To this end, we further manually augment object annotations of COCO validation set by labelling additional 197 object categories. It is denoted as **COCO*** validation set and will be released to the community. Details of the additional annotations are in Appendix A.12.

We also evaluate on **COCO20K** (Lin et al., 2014), **LVIS** (Gupta et al., 2019), **VOC** (Everingham et al., 2010), **KITTI** (Geiger et al., 2012), **Object365** (Shao et al., 2019), **OpenImages** (Kuznetsova et al., 2020), and a medical image dataset **GlaS** (Sirinukunwattana et al., 2017).

Evaluation Protocols: Our method can directly discover multiple objects on scene images, or optionally train a detector with pseudo labels. Following prior works CutLER/ CuVLER for a comprehensive comparison, we validate our method and different baselines in the following three protocols:

- Direct Object Discovery: In this protocol, our method, named **OCN**$_{disc}$, directly discovers objects on COCO* val set without training an additional detector, as discussed in Section 4.1.
- Training a Detector: In this protocol, our method, named **OCN**, will train an additional detector using discovered objects as pseudo labels from scratch, as discussed in Section 4.2.
- Zero-shot Detection: We will directly use the trained detector to evaluate on the other 7 datasets: COCO20K / LVIS / VOC / KITTI / Object365 / OpenImages / GlaS, as discussed in Section 4.3.

4.1 DIRECT OBJECT DISCOVERY

We directly discover objects on images of the COCO* validation set using our multi-object reasoning module via querying against our trained objectness network, and compare with the following baselines. Since all baselines and our OCN$_{disc}$ do not rely on any human labels or training additional multi-object detectors, this is the fairest unsupervised setting we can establish for comparison.

- VoteCut: It is proposed in CuVLER (Arica et al., 2024) to directly discover multi-objects based on both DINO and DINOv2 features.
- MaskCut: It is proposed in CutLER (Wang et al., 2023a) to directly discover multi-objects based on DINO features. The hyperparameter cut number K is set as both 3 and 10 in its favor.
- FreeMask: It is proposed in FreeSOLO (Wang et al., 2022a) to directly discover multi-objects based on DenseCL features.
- DINOSAUR (Seitzer et al., 2023): It discovers multi-objects by reconstructing DINO features.
- FOUND (Siméoni et al., 2023): This is a salient object detection method.

Note that, all other baselines (except for MaskCut with different choices of K) do not have other hyperparameters to tune for our newly annotated COCO* val set in an unsupervised setting.

Results: Table 1 compares our OCN$_{disc}$ and baselines on COCO* val set via standard **AP/AR/ Precision/ Recall scores** at different thresholds for object bounding boxes and masks. Our method is

Table 1: Quantitative results of direct object discovery on COCO* validation set.

	AP_{50}^{box}	AP_{75}^{box}	AP^{box}	AR_{100}^{box}	AP_{50}^{mask}	AP_{75}^{mask}	AP^{mask}	AR_{100}^{mask}	Pre_{50}^{mask}	Rec_{50}^{mask}	Pre_{75}^{mask}	Rec_{75}^{mask}
DINOSAUR	2.0	0.2	0.6	4.8	1.1	0.1	0.3	2.9	13.1	10.0	3.0	2.2
FOUND	4.4	1.8	2.1	3.6	3.3	1.3	1.5	3.0	**51.1**	5.5	26.9	2.9
FreeMask	3.7	0.6	1.3	4.6	3.1	0.3	0.9	3.5	22.8	9.1	5.3	2.1
MaskCut(K=3)	6.0	2.4	2.9	6.7	5.1	1.8	2.3	5.8	50.4	10.1	**30.0**	5.7
MaskCut(K=10)	6.2	2.6	2.9	7.2	5.3	2.0	2.3	6.2	48.0	10.9	27.3	6.1
VoteCut	10.8	4.9	5.5	11.3	9.5	4.0	4.6	9.8	21.0	17.2	10.6	9.7
OCN$_{disc}$ (Ours)	**19.1**	**9.0**	**10.1**	**19.6**	**17.8**	**8.7**	**9.5**	**18.9**	35.5	**30.0**	22.1	**19.6**

nearly two times better than the powerful VoteCut and three times better than others on AP/AR/Rec metrics, showing the superiority of our OCN$_{disc}$. The middle block of Figure 5 shows qualitative results of baselines and their used DINO/v2 features for grouping objects, whereas the right block shows the results of our OCN$_{disc}$ together with the learned center field and boundary distance field.

Analysis: From Table 1, we can see that the baselines such as FOUND and MaskCut can achieve high precision scores, but have rather low recall scores, meaning that they tend to correctly discover just a few objects. By contrast, our OCN$_{disc}$ achieves balanced precision and recall scores, meaning that we can correctly discover much more objects. Fundamentally, this is because the baselines mainly rely on grouping similar per-pixel features (obtained from pretrained DINO/v2) as objects, resulting in multiple similar objects being grouped as just one, as shown in Figure 5 where two cabinets are detected as one. However, our method learns clear object centers and boundaries, allowing us to easily discover individual objects especially on crowded scenes. To further validate this insight, we separately calculate scores on images with more than 5/9/13 ground truth objects respectively in Table 6 of Appendix A.8. Our method constantly maintains high scores on crowded images, whereas other baselines collapse. Results on the original COCO validation set (fewer annotations) are also provided in Appendix A.9.1. More qualitative results are in Appendix A.11 and A.13. The efficiency of our direct object discovery method is also investigated in Appendix A.14.

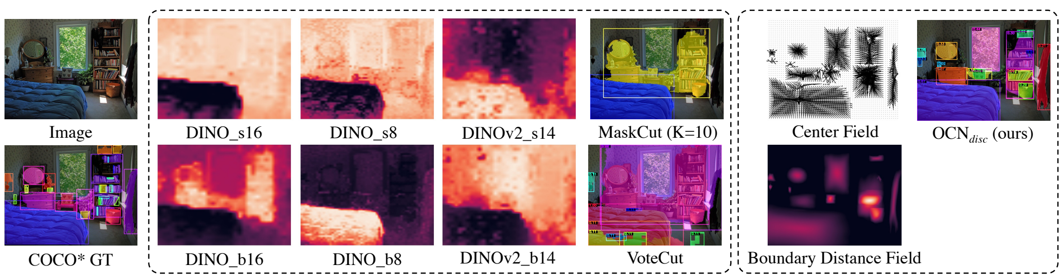


Figure 5: Qualitative results for direct object discovery on COCO* validation set. For MaskCut and VoteCut, their used DINO/v2 features for the eigenvectors of the second smallest eigenvalue are visualized. For OCN$_{disc}$, the center and boundary object representations are visualized.

4.2 Training a Detector

Exactly following CuVLER (Arica et al., 2024) for a more extensive comparison, we also train a Cascade Mask R-CNN (Cai & Vasconcelos, 2018) using our discovered objects as pseudo labels. We select CuVLER, CutLER and unSAM (Wang et al., 2024) as baselines with a diverse range of settings as follows. Note that, all final evaluation is conducted on COCO* val set which is completely held out. Since all baselines and our OCN are trained with an additional multi-object detector using their own pseudo labels, this is the fairest setting we can establish for comparison.

1) For our method, named OCN, we train two separate detectors under two settings:

- Setting #1: It is trained only on pseudo objects discovered by our method on COCO train set.
- Setting #2: It is trained on two groups of pseudo labels: one group from our discovered objects on COCO train set, another from object pseudo labels generated by VoteCut on ImageNet train set.

2) For CuVLER, it has four detectors trained under four settings below. The Settings #1/#2 are fairly comparable with our Settings #1/#2, whereas its Settings #3/#4 are from the original paper.

- Setting #1: It is trained only on pseudo objects discovered by its own VoteCut on COCO train set.

Table 2: Quantitative results of detectors with different settings on COCO* validation set.

	Training Settings	AP_{50}^{box}	AP_{75}^{box}	AP^{box}	AR_{100}^{box}	AP_{50}^{mask}	AP_{75}^{mask}	AP^{mask}	AR_{100}^{mask}
unSAM	Setting #1	3.5	2.1	2.3	30.5	3.2	2.0	2.1	27.2
	Setting #2	10.2	6.3	6.4	36.1	10.2	6.2	6.3	34.1
CutLER	Setting #1	21.2	10.8	11.6	33.4	18.2	8.1	9.1	27.7
	Setting #2	23.6	11.8	12.6	33.7	19.8	8.3	9.5	28.4
	Setting #3	26.0	14.2	14.7	37.9	22.7	11.2	11.8	32.7
CuVLER	Setting #1	26.1	13.2	14.1	36.0	22.6	10.3	11.3	30.6
	Setting #2	27.0	13.0	14.2	35.0	23.2	10.1	11.4	29.8
	Setting #3	27.2	14.0	14.9	37.2	23.2	10.7	11.8	30.2
	Setting #4	28.0	14.8	15.5	37.8	24.4	11.7	12.6	32.1
OCN (Ours)	Setting #1	31.2	15.6	16.8	40.0	28.8	12.7	14.9	36.1
	Setting #2	**32.6**	**17.2**	**18.0**	**40.9**	**29.6**	**14.4**	**15.5**	**36.5**

- Setting #2: It is trained on two groups of pseudo labels: one group from its discovered objects on COCO train set, another from object pseudo labels generated by VoteCut on ImageNet train set.
- Setting #3: It is trained only on object pseudo labels generated by VoteCut on ImageNet train set.
- Setting #4: It first uses the detector of Setting #3 to infer object pseudo labels on COCO train set, and then trains a new detector on these pseudo labels.

3) For CutLER, it has three detectors trained under three settings below. The Settings #1/#2 are fairly comparable with our Settings #1/#2, whereas its Setting #3 is from the original paper.

- Setting #1: It is trained on pseudo objects discovered by its own MaskCut on COCO train set.
- Setting #2: It is trained on two groups of pseudo labels: one group from its discovered objects on COCO train set, another from object pseudo labels generated by MaskCut on ImageNet train set.
- Setting #3: It is trained on object pseudo labels generated by MaskCut on ImageNet train set.

4) For unSAM, it has two detectors trained under two settings below. Both models are from the original paper and are included for reference.

- Setting #1: It trains a detector on pseudo objects discovered by MaskCut on ImageNet train set, and then the detector is used to infer scene images jointly with MaskCut.
- Setting #2: The detector trained in its Setting #1 is used to infer pseudo objects on SA-1B train set. Another Mask2Former is trained on these pseudo labels for inference on scene images.

Results & Analysis: Table 2 compares our method and baselines on the COCO* validation set under various training settings. We can see that: 1) Our method clearly surpasses all methods by a large margin and achieves the state-of-the-art performance. 2) Both CutLER and CuVLER can achieve reasonable results because additional detectors are likely to discover more objects. 3) The latest unSAM appears to be incapable of identifying objects precisely, although it has a rather high AR score when its detector is trained on the large-scale SA-1B dataset from SAM (Kirillov et al., 2023). Results on the original COCO validation set (fewer annotations) are also provided in Appendix A.9.2. More qualitative results are included in Appendix A.11.

4.3 Zero-shot Detection

For each method, we select its best performing detector in Table 2 and directly test it on multiple new datasets. As shown in Table 3, our OCN achieves the highest accuracy on all datasets across almost all metrics, demonstrating the generalization of our method in zero-shot detection.

Table 3: Quantitative results of zero-shot detection. Each method uses its best model in Table 2.

	COCO20K				LVIS				KITTI		VOC		Object365		OpenImages		GlaS	
	AP_{50}^{box}	AR_{100}^{box}	AP_{50}^{mask}	AR_{100}^{mask}	AP_{50}^{box}	AR_{100}^{box}	AP_{50}^{mask}	AR_{100}^{mask}	AP_{50}^{box}	AR_{100}^{box}	AP_{50}^{box}	AR_{100}^{box}	AP_{50}^{box}	AR_{100}^{box}	AP_{50}^{box}	AR_{100}^{box}	AP_{50}^{mask}	AR_{100}^{mask}
CutLER	22.4	33.1	19.6	27.2	8.5	21.8	6.7	18.7	20.8	28.9	36.8	44.0	21.7	34.2	17.2	29.6	8.8	**21.5**
CuVLER	24.1	32.6	21.1	27.2	8.9	20.8	7.2	17.9	18.8	27.9	39.4	43.7	21.9	32.5	18.3	**29.8**	3.2	11.1
OCN (Ours)	**25.9**	**35.4**	**23.6**	**30.5**	**10.4**	**24.1**	**8.9**	**21.4**	**26.7**	**34.8**	**40.4**	**47.4**	**24.7**	**35.9**	**19.0**	29.5	**9.6**	18.9

5 Ablations

As the objectness network is the core of our framework, we mainly conduct extensive ablation studies to validate our object-centric representations. Particularly, we choose different combinations of object-centric representations to train the objectness network, and then use it to discover objects as pseudo labels for training a final detector.

1) Only using a binary mask as the object-centric representation: In the task of object segmentation, a binary mask is probably the most commonly-used object representation. In particular, we remove all of our three object-centric representations, but just train the same objectness network to predict a binary mask. Then, when discovering multi-objects on scene images, we manually set a suitable step size to extensively search object candidates by querying the pretrained network.

2) Only using a binary mask and an object existence score: This is to evaluate whether the object existence score can be useful for better object segmentation. In the absence of object boundary field, the binary mask representation can update bounding boxes.

3) Only using a binary mask and an object center field: This is to evaluate whether the object center field can be useful for better object segmentation. In the absence of object boundary field, the binary mask representation can update bounding boxes.

4) Using a binary mask, an object existence score and center field: This is to evaluate whether both object existence score and center field can be useful for better object segmentation. In the absence of object boundary field, the binary mask representation can update bounding boxes.

5) Only using an object boundary field: This is to verify the importance of object boundary field.

6) Only using an object boundary field and existence score: This is to evaluate whether adding the existence score can help object segmentation on top of the object boundary field.

7) Only using an object boundary field and center field: This is to evaluate whether adding the center field can help object segmentation on top of the object boundary field.

8) Our full three-level object-centric representations: This is our full framework for reference.

With the above ablated versions, each method generates its own pseudo labels on COCO train set, and then a detector is trained on these labels together with the same pseudo labels of ImageNet train set, exactly following the Setting #2 of our full method in Section 4.2

Table 4: Ablation results of different choices of object-centric representations on COCO* validation.

	AP_{50}^{box}	AP_{75}^{box}	AP^{box}	AR_{100}^{box}	AP_{50}^{mask}	AP_{75}^{mask}	AP^{mask}	AR_{100}^{mask}
1) binary mask	23.4	10.7	11.8	33.8	19.6	8.0	9.4	35.7
2) binary mask + existence score	27.2	13.0	14.2	35.6	23.0	9.8	11.3	30.9
3) binary mask + center field	29.2	14.9	15.8	37.3	25.6	11.8	13.0	32.5
4) binary mask + existence score + center field	29.0	14.4	15.4	36.3	25.0	11.1	12.5	31.0
5) boundary field	30.7	16.1	16.9	40.7	28.1	13.9	14.8	37.0
6) boundary field + existence score	31.4	16.2	17.1	40.1	28.4	13.6	14.7	35.9
7) boundary field + center field	30.1	16.3	17.0	40.6	28.3	13.9	14.9	36.8
8) **full three level object representations**	**32.6**	**17.2**	**18.0**	**40.9**	**29.6**	**14.4**	**15.5**	**36.5**

Results & Analysis: From Table 4, we can see that: 1) The boundary distance field yields the largest performance improvement, as it retains critical information of representing complex object boundaries, thus effectively helping discover more objects in the multi-object reasoning module. 2) Without learning object existence scores and object center fields, the AP score drops, potentially due to false positives or under-segmentation in spite of a high AR score achieved. 3) The commonly-used binary mask is far from sufficient to retain complex object-centric representations. More ablation results regarding our multi-object reasoning module and the data augmentation of objectness network are provided in Appendix A.10.

6 Conclusion

In this paper, we demonstrate that multiple objects can be accurately discovered from complex real-world images, without needing any human annotations in training. This is achieved by our novel two-stage pipeline comprising an object-centric representation learning stage followed by a multi-object reasoning stage. For the first time, we explicitly define three levels of object-centric representations to be learned from the large-scale ImageNet without human labels in the first stage. These representations serve a key enabler for effectively discovering multi-objects on complex scene images in the second stage. Extensive experiments on multiple benchmarks demonstrate the state-of-the-art performance of our approach in multi-object segmentation. It would be interesting to extend our framework to the domain of large-scale 2D image generation, where the large pretrained generative models may further improve the quality of object-centric representations.

REFERENCES

Shahaf Arica, Or Rubin, Sapir Gershov, and Shlomi Laufer. CuVLER: Enhanced Unsupervised Object Discoveries through Exhaustive Self-Supervised Transformers. *CVPR*, 2024.

Görkay Aydemir, Weidi Xie, and Fatma Guney. Self-supervised object-centric learning for videos. *NeurIPS*, 2023.

Ondrej Biza, Sjoerd van Steenkiste, Mehdi S. M. Sajjadi, Gamaleldin F. Elsayed, Aravindh Mahendran, and Thomas Kipf. Invariant Slot Attention: Object Discovery with Slot-Centric Reference Frames. *ICML*, 2023.

Jack Brady, Roland S. Zimmermann, Yash Sharma, Bernhard Schölkopf, Julius von Kügelgen, and Wieland Brendel. Provably Learning Object-Centric Representations. *ICML*, 2023.

Christopher P. Burgess, Loic Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matt Botvinick, and Alexander Lerchner. MONet: Unsupervised Scene Decomposition and Representation. *arXiv:1901.11390*, 2019.

Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *CVPR*, pp. 6154–6162, 2018.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging Properties in Self-Supervised Vision Transformers. *ICCV*, 2021.

Eric Crawford and Joelle Pineau. Spatially Invariant Unsupervised Object Detection with Convolutional Neural Networks. *AAAI*, 2019.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. *CVPR*, 2009.

Aniket Didolkar, Anirudh Goyal, and Yoshua Bengio. Cycle Consistency Driven Object Discovery. *ICLR*, 2024.

Martin Engelcke, Adam R. Kosiorek, Oiwi Parker Jones, and Ingmar Posner. GENESIS: Generative Scene Inference and Sampling with Object-Centric Latent Representations. *ICLR*, 2020.

Martin Engelcke, Oiwi Parker Jones, and Ingmar Posner. GENESIS-V2: Inferring Unordered Object Representations without Iterative Refinement. *NeurIPS*, 2021.

S. M. Ali Eslami, Nicolas Heess, Theophane Weber, Yuval Tassa, Koray Kavukcuoglu, and Geoffrey E. Hinton. Attend, Infer, Repeat: Fast Scene Understanding with Generative Models. *NIPS*, 2016.

Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 88:303–338, 2010.

Ke Fan, Zechen Bai, Tianjun Xiao, Tong He, Max Horn, Yanwei Fu, Francesco Locatello, and Zheng Zhang. Adaptive Slot Attention: Object Discovery with Dynamic Slot Number. *CVPR*, 2024.

Alex Foo, Wynne Hsu, and Mong Li Lee. Multi-Object Representation Learning via Feature Connectivity and Object-Centric Regularization. *NeurIPS*, 2023.

Michael C Frank, Mika Braginsky, Daniel Yurovsky, and Virginia A Marchman. Wordbank: an open repository for developmental vocabulary data. *Journal of Child Language*, 2016.

Juergen Gall and Victor Lempitsky. Class-Specific Hough Forests for Object Detection. *CVPR*, 2009.

Juergen Gall, Angela Yao, Nima Razavi, Luc Luc Van, and Victor Lempitsky. Hough Forests for Object Detection, Tracking, and Action Recognition. *TPAMI*, 2011.

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

Anand Gopalakrishnan, Aleksandar Stanić, Jürgen Schmidhuber, and Michael Curtis Mozer. Recurrent Complex-Weighted Autoencoders for Unsupervised Object Discovery. *arXiv:2405.17283*, 2024.

Klaus Greff, Antti Rasmus, Mathias Berglund, Tele Hotloo Hao, Jürgen Schmidhuber, and Harri Valpola. Tagger: Deep unsupervised perceptual grouping. *NIPS*, 2016.

Klaus Greff, Sjoerd Van Steenkiste, and Jürgen Schmidhuber. Neural Expectation Maximization. *NIPS*, 2017.

Klaus Greff, Raphael Lopez Kaufman, Rishabh Kabra, Nick Watters, Chris Burgess, Daniel Zoran, Loie Matthey, Matthew Botvinick, and Alexander Lerchner. Multi-object representation learning with iterative variational inference. *ICML*, 2019.

Klaus Greff, Charles Herrmann, Francois Belletti, David J Fleet, Thomas Kipf, Etienne Pot, Matan Sela, Henning Meyer, Lucas Beyer, Abhijit Kundu, Tianhao Wu, Daniel Rebain, Austin Stone, Issam Laradji, Fangcheng Zhong, Daniel Duckworth, and Hsueh-ti Derek Liu. Kubric: A scalable dataset generator. *CVPR*, 2022.

Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *CVPR*, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pp. 770–778, 2016.

Olivier J. Hénaff, Skanda Koppula, Evan Shelhamer, Daniel Zoran, Andrew Jaegle, Andrew Zisserman, João Carreira, and Relja Arandjelović. Object discovery and representation networks. *ECCV*, 2022.

Taoseef Ishtiak, Qing En, and Yuhong Guo. Exemplar-FreeSOLO: Enhancing Unsupervised Instance Segmentation with Exemplars. *CVPR*, 2023.

Baoxiong Jia, Yu Liu, and Siyuan Huang. Improving Object-Centric Learning with Query Optimization. *ICLR*, 2023.

Whie Jung, Jaehoon Yoo, Sungjin Ahn, and Seunghoon Hong. Learning to Compose: Improving Object Centric Learning by Injecting Compositionality. *ICLR*, 2024.

Laurynas Karazija, Iro Laina, and Christian Rupprecht. ClevrTex: A Texture-Rich Benchmark for Unsupervised Multi-Object Segmentation. *NeurIPS*, 2021.

Diederik P Kingma and Max Welling. Auto-Encoding Variational Bayes. *ICLR*, 2014.

Daniil Kirilenko, Vitaliy Vorobyov, Alexey K. Kovalev, and Aleksandr I. Panov. Object-Centric Learning with Slot Mixture Module. *ICLR*, 2024.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *ICCV*, pp. 4015–4026, 2023.

Avinash Kori, Francesco Locatello, Fabio De Sousa Ribeiro, Francesca Toni, and Ben Glocker. Grounded Object Centric Learning. *ICLR*, 2024a.

Avinash Kori, Francesco Locatello, Ainkaran Santhirasekaram, Francesca Toni, Ben Glocker, and Fabio De Sousa Ribeiro. Identifiable Object-Centric Representation Learning via Probabilistic Slot Attention. *arXiv:2406.07141*, 2024b.

Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, et al. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *IJCV*, 128(7):1956–1981, 2020.

Sébastien Lachapelle, Divyat Mahajan, Ioannis Mitliagkas, and Simon Lacoste-Julien. Additive Decoders for Latent Variables Identification and Cartesian-Product Extrapolation. *NeurIPS*, 2023.

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. *ECCV*, 2014.

Zhixuan Lin, Yi-Fu Wu, Skand Vishwanath Peri, Weihao Sun, Gautam Singh, Fei Deng, Jindong Jiang, and Sungjin Ahn. SPACE: Unsupervised Object-Oriented Scene Representation via Spatial Attention and Decomposition. *ICLR*, 2020.

Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-Centric Learning with Slot Attention. *NeurIPS*, 2020.

Sindy Löwe, Phillip Lippe, Maja Rudolph, and Max Welling. Complex-Valued Autoencoders for Object Discovery. *TMLR*, 2022.

Sindy Löwe, Phillip Lippe, Francesco Locatello, and Max Welling. Rotating Features for Object Discovery. *NeurIPS*, 2023.

Amin Mansouri, Jason Hartford, Valence Labs, Yan Zhang, and Yoshua Bengio. Object-centric architectures enable efficient causal representation learning. *ICLR*, 2024.

Dantong Niu, Xudong Wang, Xinyang Han, Long Lian, Roei Herzig, and Trevor Darrell. Unsupervised Universal Image Segmentation. *CVPR*, 2024.

Maxime Oquab, Timothée Darcet, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2023.

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. *CVPR*, 2019.

Charles R. Qi, Or Litany, Kaiming He, and Leonidas J. Guibas. Deep Hough Voting for 3D Object Detection in Point Clouds. *ICCV*, 2019.

René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 12179–12188, 2021.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-time Object Detection with Region Proposal Networks. *NIPS*, 2015.

Mehdi S. M. Sajjadi, Daniel Duckworth, Aravindh Mahendran, Sjoerd van Steenkiste, Filip Pavetić, Mario Lučić, Leonidas J. Guibas, Klaus Greff, and Thomas Kipf. Object Scene Representation Transformer. *NeurIPS*, 2022.

Maximilian Seitzer, Max Horn, Andrii Zadaianchuk, Dominik Zietlow, Tianjun Xiao, Carl-Johann Simon-Gabriel, Tong He, Zheng Zhang, Bernhard Schölkopf, Thomas Brox, and Francesco Locatello. Bridging the Gap to Real-World Object-Centric Learning. *ICLR*, 2023.

Shuai Shao, Zeming Li, Tianyuan Zhang, Chao Peng, Gang Yu, Xiangyu Zhang, Jing Li, and Jian Sun. Objects365: A large-scale, high-quality dataset for object detection. In *ICCV*, pp. 8430–8439, 2019.

Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *TPAMI*, 2000.

Gyungin Shin, Samuel Albanie, and Weidi Xie. Unsupervised Salient Object Detection with Spectral Cluster Voting. *CVPRW*, 2022.

Oriane Siméoni, Gilles Puy, Huy V. Vo, Simon Roburin, Spyros Gidaris, Andrei Bursuc, Patrick Pérez, Renaud Marlet, and Jean Ponce. Localizing Objects with Self-Supervised Transformers and no Labels. *BMVC*, 2021.

Oriane Siméoni, Chloé Sekkat, Gilles Puy, Antonin Vobecky, Éloi Zablocki, and Patrick Pérez. Unsupervised Object Localization: Observing the Background to Discover Objects. *CVPR*, 2023.

Oriane Siméoni, Éloi Zablocki, Spyros Gidaris, Gilles Puy, and Patrick Pérez. Unsupervised Object Localization in the Era of Self-Supervised ViTs: A Survey. *IJCV*, 2024.

Gautam Singh, Fei Deng, and Sungjin Ahn. Illiterate DALL-E Learns to Compose. *ICLR*, 2022.

Korsuk Sirinukunwattana, Josien PW Pluim, Hao Chen, Xiaojuan Qi, Pheng-Ann Heng, Yun Bo Guo, Li Yang Wang, Bogdan J Matuszewski, Elia Bruni, Urko Sanchez, et al. Gland segmentation in colon histology images: The glas challenge contest. *Medical Image Analysis*, 2017.

Aleksandar Stanić, Anand Gopalakrishnan, Kazuki Irie, and Jürgen Schmidhuber. Contrastive Training of Complex-Valued Autoencoders for Object Discovery. *NeurIPS*, 2023.

Xin Tian, Ke Xu, and Rynson Lau. Unsupervised Salient Instance Detection. *CVPR*, 2024.

Andrey Voynov, Stanislav Morozov, and Artem Babenko. Object Segmentation Without Labels with Large-Scale Generative Models. *ICML*, 2021.

Xinlong Wang, Zhiding Yu, Shalini De Mello, Jan Kautz, Anima Anandkumar, Chunhua Shen, and Jose M. Alvarez. FreeSOLO: Learning to Segment Objects without Annotations. *CVPR*, 2022a.

Xudong Wang, Rohit Girdhar, Stella X. Yu, and Ishan Misra. Cut and Learn for Unsupervised Object Detection and Instance Segmentation. *CVPR*, 2023a.

Xudong Wang, Jingfeng Yang, and Trevor Darrell. Segment Anything without Supervision. *NeurIPS*, 2024.

Yangtao Wang, Xi Shen, Shell Hu, Yuan Yuan, James Crowley, and Dominique Vaufreydaz. Self-Supervised Transformers for Unsupervised Object Discovery using Normalized Cut. *CVPR*, 2022b.

Yangtao Wang, Xi Shen, Yuan Yuan, Yuming Du, Maomao Li, Shell Xu Hu, James L. Crowley, and Dominique Vaufreydaz. TokenCut: Segmenting Objects in Images and Videos With Self-Supervised Transformer and Normalized Cut. *TPAMI*, 2023b.

Ziyu Wang, Mike Zheng Shou, and Mengmi Zhang. Object-centric Learning with Cyclic Walks between Parts and Whole. *NeurIPS*, 2023c.

Marissa A. Weis, Kashyap Chitta, Yash Sharma, Wieland Brendel, Matthias Bethge, Andreas Geiger, and Alexander S. Ecker. Benchmarking Unsupervised Object Representations for Video Sequences. *JMLR*, 2021.

Thaddäus Wiedemer, Jack Brady, Alexander Panfilov, Attila Juhos, Matthias Bethge, and Wieland Brendel. Provable Compositional Generalization for Object-Centric Learning. *ICLR*, 2024.

Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

Yiheng Xie, Towaki Takikawa, Shunsuke Saito, Or Litany, Shiqin Yan, Numair Khan, Federico Tombari, James Tompkin, Vincent Sitzmann, and Srinath Sridhar. Neural Fields in Visual Computing and Beyond. *Computer Graphics Forum*, 2022.

Yafei Yang and Bo Yang. Promising or Elusive? Unsupervised Object Segmentation from Real-world Single Images. *NeurIPS*, 2022.

Jinyang Yuan, Tonglin Chen, Bin Li, and Xiangyang Xue. Compositional Scene Representation Learning via Reconstruction: A Survey. *TPAMI*, 2023.

Andrii Zadaianchuk, Maximilian Seitzer, and Georg Martius. Object-centric learning for real-world videos by predicting temporal feature similarities. *NeurIPS*, 2024.

Xin Zhang, Jinheng Xie, Yuan Yuan, Michael Bi Mi, and Robby T. Tan. HEAP: Unsupervised Object Discovery and Localization with Contrastive Grouping. *AAAI*, 2024.

A Appendix

The appendix includes:

A.1 Details for object-centric representations

Calculation of Signed Distance Field. Given a binary mask $\boldsymbol{M} \in \mathcal{R}^{H\times W\times 1}$, we calculate the distance between each pixel to its closest boundary point with `distanceTransform()` function in the `opencv` library (`https://docs.opencv.org/4.x/d7/d1b/group__imgproc__misc.html`). The function takes a binary mask as input and computes the shortest path length to the nearest zero pixel for all non-zero pixels. Thus, we first compute the distance field within the object , denoted as $\boldsymbol{S}_{obj}$, using the object binary mask $\boldsymbol{M}$. Then, we compute the distance field within the background, denoted as $\boldsymbol{S}_{bg}$, using $(1-\boldsymbol{M})$. The signed distance field for the whole image is $\boldsymbol{S} = \boldsymbol{S}_{obj} - \boldsymbol{S}_{bg}$. Specifically, when using `distanceTransform()`, we set the distance type as L2 (Euclidean distance) and mask size to be 3.

A.2 Details for Objectness Network.

Objectness Network Architecture. The *object existence* model employs ResNet50 (He et al., 2016) as the backbone. Following the backbone, the classification head consists of a single linear layer with output dimension 1 and a sigmoid activation layer. The prediction for *object center field* and *object boundary distance* shares the same DPT-large (Ranftl et al., 2021) backbone with a 256-dimensional output size. Dense feature maps extracted from this backbone have the same resolution as input images and the number of channels is 256. There are two prediction heads for the prediction of *object center field* and *object boundary distance* separately.

Table 5: Architecture of prediction heads for *object center field* and *object boundary distance*.

center field prediction head					boundary field prediction head				
	type	channels	activation	stride		type	channels	activation	stride
layer 1	conv 1x1	512	RELU	1	layer 1	conv 1x1	512	RELU	1
layer 2	conv 3x3	512	RELU	1	layer 2	conv 3x3	512	RELU	1
layer 3	conv 1x1	1024	RELU	1	layer 3	conv 1x1	1024	RELU	1
layer 4	conv 1x1	2	RELU	1	layer 4	conv 1x1	1	RELU	1

Objectness Network Training Strategy. The object existence model is trained using the Adam optimizer for 100K iterations with a batch size of 64. The learning rate is set to be a constant 0.0001. The object center and boundary models are jointly trained using the Adam optimizer for 50K iterations with a batch size of 16. The learning rate starts at 0.0001 and is divided by 10 at 10K and 20K iteration.

Objectness Network Training Data. We use the ImageNet train set with about 1.28 million images as the training set for the objectness network. For each ImageNet image, its object mask is the most confident mask generated by VoteCut proposed in CuVLER (Arica et al., 2024). For the training of the object existence model, negative samples that do not contain objects are created by cropping the largest rectangle region on the background. For positive samples that contain objects, we apply the random crop augmentation onto the original ImageNet image and discard the crop without a foreground object. For the training of the object center and boundary model, we first calculate the ground truth center field and boundary distance field based on the original full ImageNet image. Then, we apply the random crop augmentation onto the original image as well as the two representations. Specifically, the scale of the random crop is between 0.08 to 1, which implies the lower and upper bounds for the random area of the crop. The aspect ratio range of the random crop is between 0.75 and 1.33. Lastly, each image is resized to 128×128 before feeding into Objectness Network.

A.3 Details for Multi-Object Reasoning Module

Initial Object Proposal Generation. Motivated by anchor box generation in Faster R-CNN (Ren et al., 2015). We use five scales $[32, 64, 128, 256, 512]$ and three aspect ratios $[0.5, 1, 2]$. At each scale, we randomly and uniformly sample proposal centers based on scale sizes. At each sampled center, we generate three boxes with different aspect ratios.

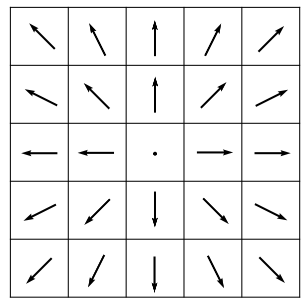


Figure 6: Predefined Kernel for Center Reasoning

Predefined Kernel for Center Reasoning. As illustrated in Figure 6, each position within the kernel is defined as a 2-dimensional unit vector pointing towards the center of the kernel. Specifically, the value at the kernel center with position $[2, 2]$ is $(0, 0)$. The value at the $(i, j)^{th}$ position, denoted as $\mathbb{K}_{i,j}$, is defined and normalized as:

$$\mathbb{K}_{i,j} = \frac{[2, 2] - [i, j]}{\|[2, 2] - [i, j]\|}$$

To evaluate how *Center Field* matches with this anti-center pattern, we apply convolution onto *Center Field* with this kernel to calculate their average cosine similarity for each pixel in the *Center Field*. We set the threshold τ_c to be 0.25.

More Details for Center Reasoning. While deriving the *anti-center map* with the predefined kernel, we also find the boundary of the *Center Field*. Since on the *anti-center map*, values at the boundary of the *Center Field* will also be positive, we thus ignore the values on the *Center Field* boundary. Examples of center reasoning are provided in Figure 10.

More Details for Boundary Reasoning. Let $\boldsymbol{f}_p^b \in \mathcal{R}^{128\times128\times1}$ be the distance field for proposal P and $\nabla\boldsymbol{f}_p^b \in \mathcal{R}^{128\times128\times2}$ is the gradient map for $\boldsymbol{f}_p^b$, where $\nabla\boldsymbol{f}_p^b[u, v] = (\frac{\partial \boldsymbol{f}_p^b}{\partial u}, \frac{\partial \boldsymbol{f}_p^b}{\partial v})$. And $\|\nabla\boldsymbol{f}_p^b\| \in \mathcal{R}^{128\times128\times1}$ is the norm for the gradient map. To make the bounding box update more stable, we use two strategies: (1) Use the averaged distance field gradient to replace the gradient at a single pixel position; (2) Apply adjustment on the calculated update step for a more aggressive expansion and conservative contraction.

(1) Since the distance field within the object and outside the object are normalized separately, the gradient average operation needs to be applied separately. Thus, we first apply sigmoid σ function onto the boundary field to generate mask for foreground $\sigma(\boldsymbol{f}_p^b)$ and background $1 - \sigma(\boldsymbol{f}_p^b)$. Then gradients are averaged separately on the two masks and combined as the averaged gradient norm map for the distance field $AVG(\|\nabla\boldsymbol{f}_p^b\|) \in \mathcal{R}^{128\times128\times1}$. We replace $\|\nabla\boldsymbol{f}_p^b\|)$ with $AVG(\|\nabla\boldsymbol{f}_p^b\|))$ when calculating box updates.

$$AVG(\|\nabla\boldsymbol{f}_p^b\|) = \frac{\sum\sigma(\boldsymbol{f}_p^b)\cdot\|\nabla\boldsymbol{f}_p^b\|}{\sum\sigma(\boldsymbol{f}_p^b)}\cdot\sigma(\boldsymbol{f}_p^b) + \frac{\sum(1-\sigma(\boldsymbol{f}_p^b))\cdot\|\nabla\boldsymbol{f}_p^b\|}{\sum(1-\sigma(\boldsymbol{f}_p^b))}\cdot(1-\sigma(\boldsymbol{f}_p^b)) \quad (7)$$

(2) Empirically, box contraction needs to be more conservative since objects could be overlooked if the proposal is over-tightened. For example, for a person wearing a tie, if the proposal around the person gets shrunk too much, the object of interest may transfer to the tie instead. Also, for efficiency, it is suitable to make more aggressive expansion since objects can still be well seen from a proposal larger than its tightest bounding box. Thus, we further adjust the calculated updates with

an adjustment ratio $\tau_{adjust} = 0.5$. Instead of directly using Eq. 6, we use the following formulas to calculate boundary update:

$$
\begin{aligned}
P^{u_1} &\longleftarrow P^{u_1} - \frac{max(\boldsymbol{f}^b_{p_t})}{\|\frac{\partial \boldsymbol{f}^b_{p_t}}{\partial u}, \frac{\partial \boldsymbol{f}^b_{p_t}}{\partial v}\|} - \tau_{adjust} * \frac{\|max(\boldsymbol{f}^b_{p_t})\|}{\|\frac{\partial \boldsymbol{f}^b_{p_t}}{\partial u}, \frac{\partial \boldsymbol{f}^b_{p_t}}{\partial v}\|}, \qquad \textit{where } (u,v) = argmax \boldsymbol{f}^b_{p_t} \\
P^{v_1} &\longleftarrow P^{v_1} - \frac{max(\boldsymbol{f}^b_{p_l})}{\|\frac{\partial \boldsymbol{f}^b_{p_l}}{\partial u}, \frac{\partial \boldsymbol{f}^b_{p_l}}{\partial v}\|} - \tau_{adjust} * \frac{\|max(\boldsymbol{f}^b_{p_l})\|}{\|\frac{\partial \boldsymbol{f}^b_{p_l}}{\partial u}, \frac{\partial \boldsymbol{f}^b_{p_l}}{\partial v}\|}, \qquad \textit{where } (u,v) = argmax \boldsymbol{f}^b_{p_l} \\
P^{u_2} &\longleftarrow P^{u_2} + \frac{max(\boldsymbol{f}^b_{p_b})}{\|\frac{\partial \boldsymbol{f}^b_{p_b}}{\partial u}, \frac{\partial \boldsymbol{f}^b_{p_b}}{\partial v}\|} + \tau_{adjust} * \frac{\|max(\boldsymbol{f}^b_{p_b})\|}{\|\frac{\partial \boldsymbol{f}^b_{p_b}}{\partial u}, \frac{\partial \boldsymbol{f}^b_{p_b}}{\partial v}\|}, \qquad \textit{where } (u,v) = argmax \boldsymbol{f}^b_{p_b} \\
P^{v_2} &\longleftarrow P^{v_2} + \frac{max(\boldsymbol{f}^b_{p_r})}{\|\frac{\partial \boldsymbol{f}^b_{p_r}}{\partial u}, \frac{\partial \boldsymbol{f}^b_{p_r}}{\partial v}\|} + \tau_{adjust} * \frac{\|max(\boldsymbol{f}^b_{p_r})\|}{\|\frac{\partial \boldsymbol{f}^b_{p_r}}{\partial u}, \frac{\partial \boldsymbol{f}^b_{p_r}}{\partial v}\|}, \qquad \textit{where } (u,v) = argmax \boldsymbol{f}^b_{p_r}
\end{aligned} \tag{8}
$$

Parameters for Proposal Updating. Each proposal undergoes 50 iterations of updates at most. For efficiency, we stop a proposal from being updated once it meets the following criteria. Specifically, the calculated maximum expansion for the proposal should be smaller than 0 (it means the boarder moves outside of object boundary), and the maximum shrinkage should be smaller than a small margin, which we set to be 16 pixels. While it is acceptable for the proposal to be slightly larger than the tightest bounding box, it should not be smaller. Examples of boundary reasoning can be found in Figure 7, 8, 9.

A.4 Details for Object Mask and Confidence Score Calculation.

For a converged proposal P, we can compute its object mask $\boldsymbol{M}_p$ as the union of mask from center field and mask from boundary field:

$$\boldsymbol{M}_p^{center} = \begin{cases} 1, & \text{if } \|f_p^c\| \geq 0.5 \\ 0, & \text{otherwise} \end{cases} \quad \boldsymbol{M}_p^{boundary} = \begin{cases} 1, & \text{if } \sigma(\boldsymbol{f}_p^b) \geq 0.5 \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

$$\boldsymbol{M}_p = \cup(\boldsymbol{M}_p^{center}, \boldsymbol{M}_p^{boundary}) \tag{10}$$

To calculate the confidence score $conf_p$ for proposal P, we consider its object existence score, center field, and boundary field. Specifically, we also consider mask area when calculating the confidence by comparing the object area in P with other objects' areas within the same image. Suppose there are K discovered objects within the image, the final score is calculated as:

$$conf_p = f_p^e * max(\|\boldsymbol{f}_p^c\|) * max(\boldsymbol{f}_p^b) * \left(\frac{\sum \boldsymbol{M}_p}{max_{k \in K} \sum \boldsymbol{M}_k}\right)^{0.25} \tag{11}$$

A.5 Details for Pseudo Label Processing

Given a set of discovered objects from scene images, we perform selection and assign each of them a weight to use them as pseudo labels for training the detector. Following the definition in the Section A.4, an object proposal P will be selected if it satisfies three conditions below:

$$f_p^e \geq \tau_{conf}^e; \quad max(\|\boldsymbol{f}_p^c\|) \geq \tau_{conf}^c; \quad max(\boldsymbol{f}_p^b) \geq \tau_{conf}^b \tag{12}$$

The three threshold correspond to object existence score (τ_{conf}^e), maximum norm in *center field* (τ_{conf}^c) and maximum value in *boundary distance field* (τ_{conf}^b). In our paper, we set:

$$\tau_{conf}^e = 0.5; \quad \tau_{conf}^c = 0.8; \quad \tau_{conf}^b = 0.75 \tag{13}$$

For each selected proposal, its weight for the detector training is determined by its relative area in the scene image: $\left(\frac{\sum \boldsymbol{M}_p}{max_{k \in K} \sum \boldsymbol{M}_k}\right)^{0.25}$.

Table 6: Detailed results of direct object discovery on crowded images of COCO* validation set.

# of objects	>=5				>=9				>=13			
	AP^{box}_{50}	AR^{box}_{100}	AP^{mask}_{50}	AR^{mask}_{100}	AP^{box}_{50}	AR^{box}_{100}	AP^{mask}_{50}	AR^{mask}_{100}	AP^{box}_{50}	AR^{box}_{100}	AP^{mask}_{50}	AR^{mask}_{100}
MaskCut(K=3)	3.7	4.2	3.3	3.7	2.4	2.9	2.2	2.5	1.8	2.1	1.6	1.9
MaskCut(K=10)	4.0	4.7	3.6	4.1	2.7	3.2	2.5	2.8	2.2	2.4	2.0	2.2
VoteCut	7.7	8.2	6.3	7.1	5.7	6.2	4.6	5.4	4.6	5.0	3.5	4.3
OCN_{disc} (Ours)	**16.5**	**17.4**	**15.4**	**16.8**	**15.1**	**15.6**	**13.4**	**15.0**	**14.1**	**14.5**	**12.7**	**13.9**

A.6 Details for Detector Training.

The architecture for the Class Agnostic Detector is Cascade Mask RCNN. All experiments are performed with the Detectron2 (Wu et al., 2019) platform. Detectors are optimized for 25K iterations using SGD optimizer with a learning rate of 0.005 and a batch size of 16. We use a weight decay of 0.00005 and 0.9 momentum. Following CutLER (Wang et al., 2023a), we also use copy-paste augmentation with a uniformly sampled downsample ratio between 0.3 and 1.0.

A.7 Details for Datasets.

COCO (Lin et al., 2014): The MS COCO (Microsoft Common Objects in Context) dataset is a large-scale object detection and segmentation dataset. The COCO in the paper refers to the 2017 version that contains 118K training images and 5K validation images.

COCO 20K (Lin et al., 2014): COCO 20K is a subset of the COCO trainval2014 with 19817 images. Since it contains images from both training and validation set from the 2014 version of COCO, this dataset is generally used to evaluate unsupervised approaches.

LVIS (Gupta et al., 2019): LVIS (Large Vocabulary Instance Segmentation) is a dataset for long tail instance segmentation. It contains 164,000 images with more than 1,200 categories and more than 2 million high-quality instance-level segmentation masks.

KITTI (Geiger et al., 2012): KITTI (Karlsruhe Institute of Technology and Toyota Technological Institute) is one of the most popular datasets for use in mobile robotics and autonomous driving. Our method is evaluated with 7521 images from its trainval split.

PASCAL VOC (Everingham et al., 2010): The PASCAL Visual Object Classes (VOC) 2012 dataset is a widely used benchmark for object detection, containing 1464 training images and 1449 validation images.

Object365 V2 (Shao et al., 2019): Objects365 is a large-scale object detection dataset. It has 365 object categories and over 600K training images. We evaluate our method in terms of object detection on its validation split with 80K images.

OpenImages V6 (Kuznetsova et al., 2020): OpenImages V6 is a large-scale dataset, consists of 9 million training images, 41,620 validation samples, and 125,456 test samples. We evaluate our method in terms of object detection on its validation split.

GlaS (Sirinukunwattana et al., 2017): GlaS is a medical image dataset for gland segmentation. It consists of 165 images derived from 16 H&E stained histological sections of stage T3 or T42 colorectal adenocarcinoma.

A.8 More Experimental Results on COCO* Validation Set

To further validate this insight, we separately calculate scores on images with more than 5/9/13 ground truth objects respectively in Table 6 of Appendix. Our method constantly maintains high scores on crowded images, whereas other baselines collapse. This clearly shows the superiority of our method in discovering many objects on hard images.

A.9 Experiment results on original COCO validation set

This section presents the experiment results evaluated on original COCO validation set.

Table 7: Quantitative results of direct object discovery on COCO validation set.

	AP^{box}_{50}	AP^{box}_{75}	AP^{box}	AR^{box}_{100}	AP^{mask}_{50}	AP^{mask}_{75}	AP^{mask}	AR^{mask}_{100}
DINOSAUR	2.1	0.2	0.6	5.5	0.8	0.1	0.2	2.5
FOUND	4.7	2.1	2.3	4.5	3.7	1.5	1.8	3.7
FreeMask	4.1	0.7	1.4	4.3	3.5	0.4	1.1	3.4
MaskCut(K=3)	6.4	2.5	3.1	7.7	5.4	1.8	2.3	6.5
MaskCut(K=10)	6.0	2.7	3.1	8.2	5.5	1.7	2.2	6.9
VoteCut	11.0	5.0	5.6	12.4	9.4	4.0	4.6	10.5
OCN$_{disc}$ (Ours)	**15.7**	**6.9**	**7.9**	**16.5**	**14.7**	**6.9**	**7.5**	**15.9**

A.9.1 Direct Object Discovery Results on Original COCO Validation Set

A.9.2 Training a Detector Results on original COCO validation set

Table 8: Quantitative results of detectors with different settings on COCO validation set.

	Training Setting	AP^{box}_{50}	AP^{box}_{75}	AP^{box}	AR^{box}_{100}	AP^{mask}_{50}	AP^{mask}_{75}	AP^{mask}	AR^{mask}_{100}
unSAM	Setting #1	2.1	1.1	1.2	27.0	1.8	0.9	1.0	23.5
	Setting #2	5.9	3.2	3.4	30.0	5.9	3.1	3.3	27.4
CutLER	Setting #1	19.3	9.9	10.6	29.4	16.3	7.3	8.2	23.2
	Setting #2	20.8	10.4	11.1	29.7	17.2	7.0	8.1	23.3
	Setting #3	21.9	11.8	12.3	32.7	18.9	9.2	9.7	27.0
CuVLER	Setting #1	22.9	11.7	12.4	31.8	18.7	7.3	8.8	23.9
	Setting #2	23.2	11.3	12.3	31.2	19.7	8.5	9.5	24.9
	Setting #3	22.9	11.8	12.6	32.9	19.3	8.9	9.8	25.1
	Setting #4	23.4	12.1	12.8	32.2	20.4	9.6	10.4	26.8
OCN (Ours)	Setting #1	24.1	11.2	12.5	34.2	22.2	9.9	11.1	29.9
	Setting #2	**25.4**	**12.7**	**13.6**	**35.2**	**22.9**	**10.7**	**11.7**	**30.3**

A.10 More ablations

Selection of Fixed Step Size for Binary Baseline. Since the information provided by binary mask representation is very limited, the final discovered objects can be very sensitive to the step size. In order to choose a good step size in favor of the binary mask baseline, we randomly select 100 images from COCO validation set and evaluate the results for a step size of 5, 15, 20, 30. According to the results shown in Table 9, we select 20 as the fixed step size.

Table 9: Results of different step sizes for binary baseline on COCO validation set.

step size	AP^{box}_{50}	AP^{box}_{75}	AP^{box}	AR^{box}_{100}	AP^{mask}_{50}	AP^{mask}_{75}	AP^{mask}	AR^{mask}_{100}
5	8.7	5.4	4.9	6.0	5.2	2.0	2.8	3.5
15	9.2	5.6	5.2	7.2	5.4	3.0	3.4	4.5
20	9.3	6.0	5.4	7.9	7.8	2.8	3.9	5.5
30	7.2	5.7	4.7	6.6	5.6	2.2	3.3	4.4

Ablation on Parameters for Pseudo Label Processing. We perform ablation studies on the parameters used in A.5. Specifically, we choose a wide range, *i.e.*, ($0 \sim 0.95$) for score thresholds of object existence τ^{e}_{conf}, object center τ^{c}_{conf} and object boundary τ^{b}_{conf} on 7 datasets. As shown in Tables 10&11, more tolerant thresholds lead to higher AR scores because more objects can be discovered, but a decrease in AP because of low-quality detections. On the other hand, if thresholds are too strict, both AR and AP scores drop because only a limited number of objects are discovered. Nevertheless, our method is not particularly sensitive to the selection of thresholds as it demonstrates good performance across different thresholds.

Ablation on Random Cropping Augmentation for the Objectness Network. During training our objectness network on ImageNet, we originally apply random cropping augmentation. Here, we conduct an additional ablation study by omitting the random cropping operation during training the objectness network while keeping all other settings the same. Table 12 shows the quantitative results

Table 10: Ablation results for thresholds of object existence τ^e_{conf}, object center τ^c_{conf} and object boundary τ^b_{conf} on COCO* validation set.

τ^e_{conf}	τ^c_{conf}	τ^b_{conf}	AP^{box}_{50}	AP^{box}_{75}	AP^{box}	AR^{box}_{100}	AP^{mask}_{50}	AP^{mask}_{75}	AP^{mask}	AR^{mask}_{100}
0.0	0.8	0.75	31.2	16.7	17.4	**41.0**	28.7	**14.6**	15.3	**37.2**
0.25	0.8	0.75	31.5	16.7	17.5	40.8	28.6	14.3	15.2	36.7
<u>**0.5**</u>	0.8	0.75	**32.6**	**17.2**	**18.0**	40.9	**29.6**	14.4	**15.5**	36.5
0.75	0.8	0.75	30.8	16.2	16.9	38.9	27.7	13.3	14.3	34.7
0.95	0.8	0.75	28.1	13.4	14.7	34.4	24.3	10.7	12.1	30.1
0.5	**0.0**	0.75	32.5	16.4	17.5	40.0	29.2	13.6	14.9	35.8
0.5	**0.25**	0.75	31.8	16.4	17.3	39.9	28.5	13.5	14.7	35.7
0.5	**0.5**	0.75	31.0	16.2	17.0	40.2	27.7	13.3	14.4	36.0
0.5	<u>**0.8**</u>	0.75	**32.6**	**17.2**	**18.0**	**40.9**	**29.6**	**14.4**	**15.5**	**36.5**
0.5	**0.95**	0.75	29.8	15.8	16.5	38.1	26.8	13.2	14.1	34.2
0.5	0.8	**0.0**	31.8	16.0	17.0	38.7	28.4	13.2	14.5	34.6
0.5	0.8	**0.25**	31.2	16.1	17.0	38.9	27.8	13.2	14.3	34.7
0.5	0.8	**0.5**	31.7	16.9	17.5	40.6	28.4	13.7	14.7	36.0
0.5	0.8	<u>**0.75**</u>	**32.6**	**17.2**	**18.0**	**40.9**	**29.6**	**14.4**	**15.5**	**36.5**
0.5	0.8	**0.95**	31.6	17.5	17.9	39.8	28.0	13.3	14.5	35.0

Table 11: Ablation results for thresholds of object existence τ^e_{conf}, object center τ^c_{conf} and object boundary τ^b_{conf} on COCO20K, LVIS, KITTI, VOC, Object365 and OpenImages.

			COCO				COCO20K				LVIS				KITTI		VOC		Object365		OpenImages	
τ^e_{conf}	τ^c_{conf}	τ^b_{conf}	AP^{box}_{50}	AR^{box}_{100}	AP^{mask}_{50}	AR^{mask}_{100}	AP^{box}_{50}	AR^{box}_{100}	AP^{mask}_{50}	AR^{mask}_{100}	AP^{box}_{50}	AR^{box}_{100}	AP^{mask}_{50}	AR^{mask}_{100}	AP^{box}_{50}	AR^{box}_{100}	AP^{box}_{50}	AR^{box}_{100}	AP^{box}_{50}	AR^{box}_{100}	AP^{box}_{50}	AR^{box}_{100}
0.0	0.8	0.75	23.8	35.1	21.9	**30.8**	24.3	35.2	22.6	**31.1**	10.2	**24.9**	**9.0**	**22.6**	25.3	32.5	38.5	46.9	23.6	**36.3**	18.3	**29.5**
0.25	0.8	0.75	24.1	34.8	22.0	30.3	24.6	35.0	22.6	30.6	10.2	24.4	8.7	21.9	25.0	34.0	39.1	46.6	23.8	36.0	18.7	29.4
<u>**0.5**</u>	0.8	0.75	**25.4**	**35.2**	**22.9**	30.3	**25.9**	**35.4**	**23.6**	30.5	**10.4**	24.1	8.9	21.4	**26.7**	**34.8**	**40.4**	**47.4**	**24.7**	35.9	**19.0**	**29.5**
0.75	0.8	0.75	24.5	33.7	21.9	28.8	25.1	34.1	22.7	29.2	9.9	22.5	8.3	20.0	25.5	33.6	**40.4**	46.7	23.8	36.0	18.7	29.4
0.95	0.8	0.75	23.2	30.2	19.9	25.0	23.8	30.5	20.6	25.3	8.7	18.8	6.9	16.3	21.6	29.6	39.4	43.7	21.6	30.0	18.8	26.5
0.5	**0.0**	0.75	**25.7**	34.5	22.8	29.8	**26.2**	34.8	23.4	30.1	**10.4**	23.3	8.5	20.9	**28.7**	**35.5**	**41.3**	47.0	24.5	35.1	19.7	29.0
0.5	**0.25**	0.75	25.0	34.4	22.2	29.5	25.6	34.8	23.0	29.8	10.1	23.2	8.3	20.6	27.7	33.6	41.0	46.8	23.8	35.1	19.3	29.0
0.5	**0.5**	0.75	24.5	34.7	21.8	29.9	25.1	34.8	22.5	30.1	9.8	23.6	8.0	21.1	24.1	32.7	40.3	46.7	23.3	35.3	**19.9**	**29.7**
0.5	<u>**0.8**</u>	0.75	25.4	**35.2**	**22.9**	**30.3**	25.9	**35.4**	**23.6**	**30.5**	**10.4**	**24.1**	**8.9**	**21.4**	26.7	34.8	40.4	**47.4**	**24.7**	**35.9**	19.0	29.5
0.5	**0.95**	0.75	23.7	32.9	21.1	28.3	24.3	33.2	21.8	28.5	9.6	21.6	8.2	19.3	25.7	33.3	38.6	45.6	22.5	33.2	18.3	28.4
0.5	0.8	**0.0**	24.7	33.4	21.9	28.7	25.3	33.6	22.6	29.0	10.1	22.3	8.2	19.8	27.4	33.4	40.0	45.9	23.6	33.8	19.3	28.3
0.5	0.8	**0.25**	24.6	33.6	21.8	28.9	25.3	34.0	22.5	29.3	9.8	22.4	8.0	19.8	26.7	33.5	40.7	46.1	23.2	34.1	19.7	28.6
0.5	0.8	**0.5**	25.3	**35.2**	22.4	30.0	**25.9**	35.3	23.1	30.4	10.0	23.6	8.4	20.9	25.4	34.3	**41.3**	**47.8**	23.7	35.8	**19.9**	**29.9**
0.5	0.8	<u>**0.75**</u>	**25.4**	**35.2**	**22.9**	**30.3**	**25.9**	**35.4**	**23.6**	**30.5**	10.4	**24.1**	8.9	**21.4**	26.7	34.8	40.4	47.4	**24.7**	**35.9**	19.0	29.5
0.5	0.8	**0.95**	20.4	32.2	19.7	28.6	24.4	34.4	22.7	29.8	**10.5**	23.3	**9.0**	21.0	**29.7**	**35.1**	37.6	46.4	23.8	34.8	17.8	29.2

on the COCO* validation set. We can see that random cropping is indeed helpful for the objectness network to learn robust center and boundary fields. Primarily, this is because during the multi-object reasoning stage, many proposals just have partial or fragmented objects, but the random cropping augmentation inherently enables the objectness network to infer rather accurate center and boundary field for those partial objects, thus driving the proposals to be updated correctly.

Table 12: Ablation results on COCO* validation set for random cropping augmentation of the objectness network.

	AP^{box}_{50}	AP^{box}_{75}	AP^{box}	AR^{box}_{100}	AP^{mask}_{50}	AP^{mask}_{75}	AP^{mask}	AR^{mask}_{100}
OCN_{disc} (with random cropping)	19.1	9.0	10.1	19.6	17.8	8.7	9.5	18.9
OCN_{disc} (w/o random cropping)	15.7	7.5	8.2	18.1	15.6	6.6	7.9	17.4

A.11 More visualizations.

Figures 7& 8& 9 are examples for boundary reasoning. Figure 10 shows examples of center reasoning. Figures 11& 12 present additional qualitative results of Direct Object Discovery as discussed in Section 4.1. Figure 13 presents qualitative results from trained detectors as discussed in Section 4.2.

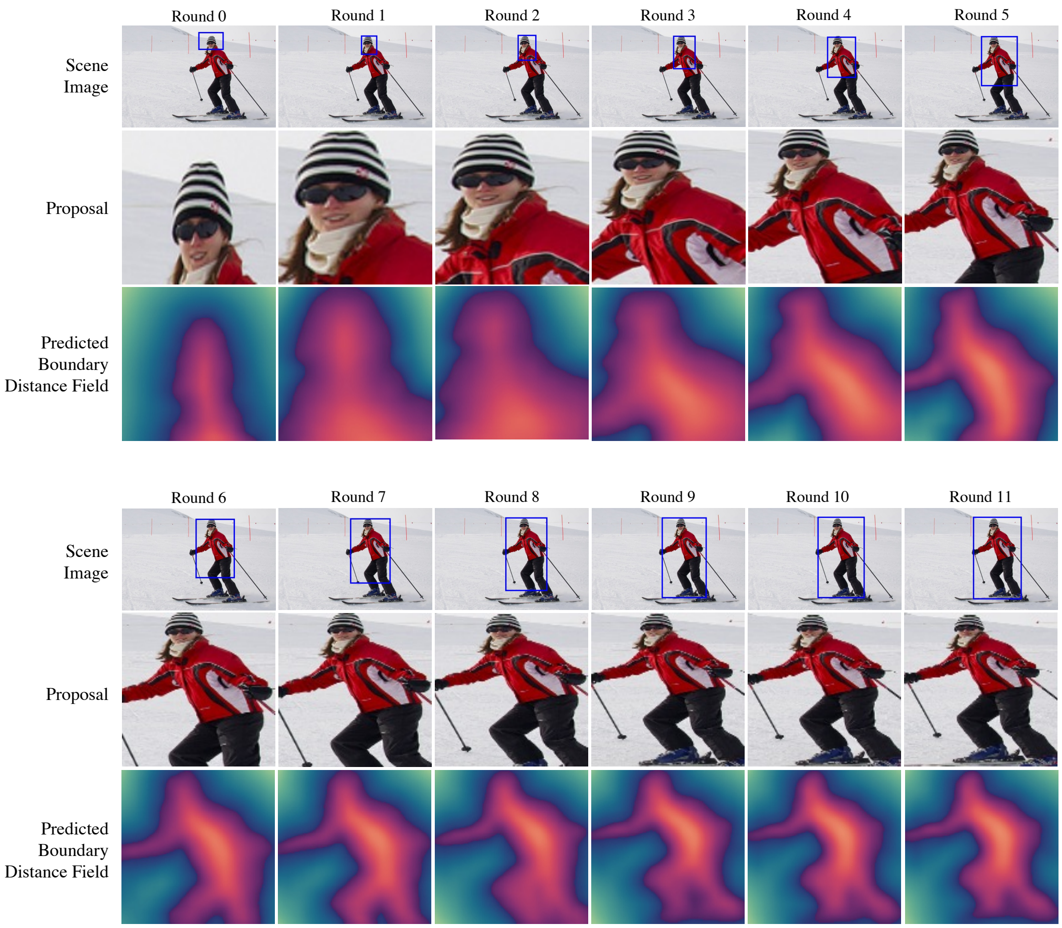


Figure 7: Examples for boundary reasoning.

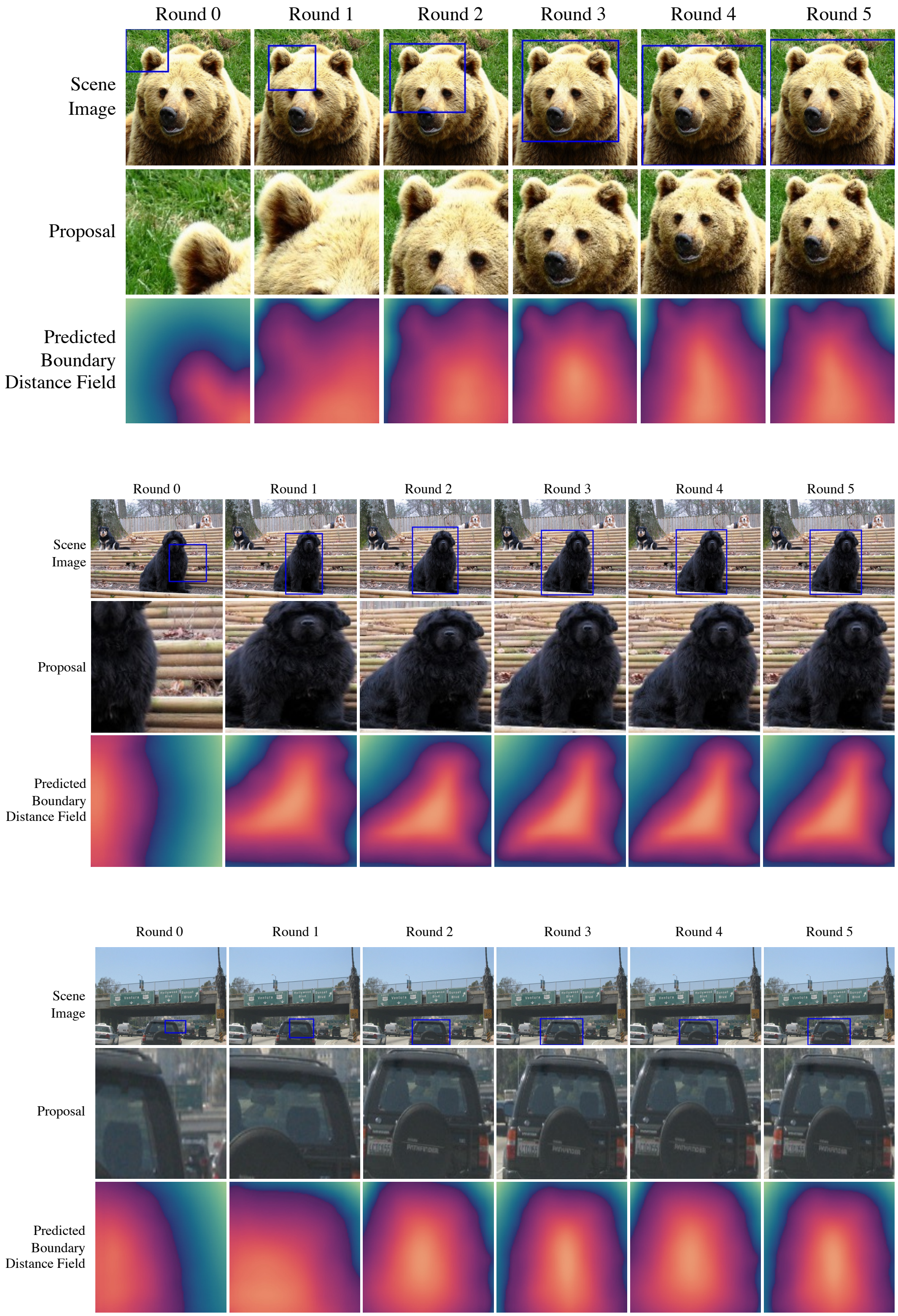


Figure 8: Examples for boundary reasoning.

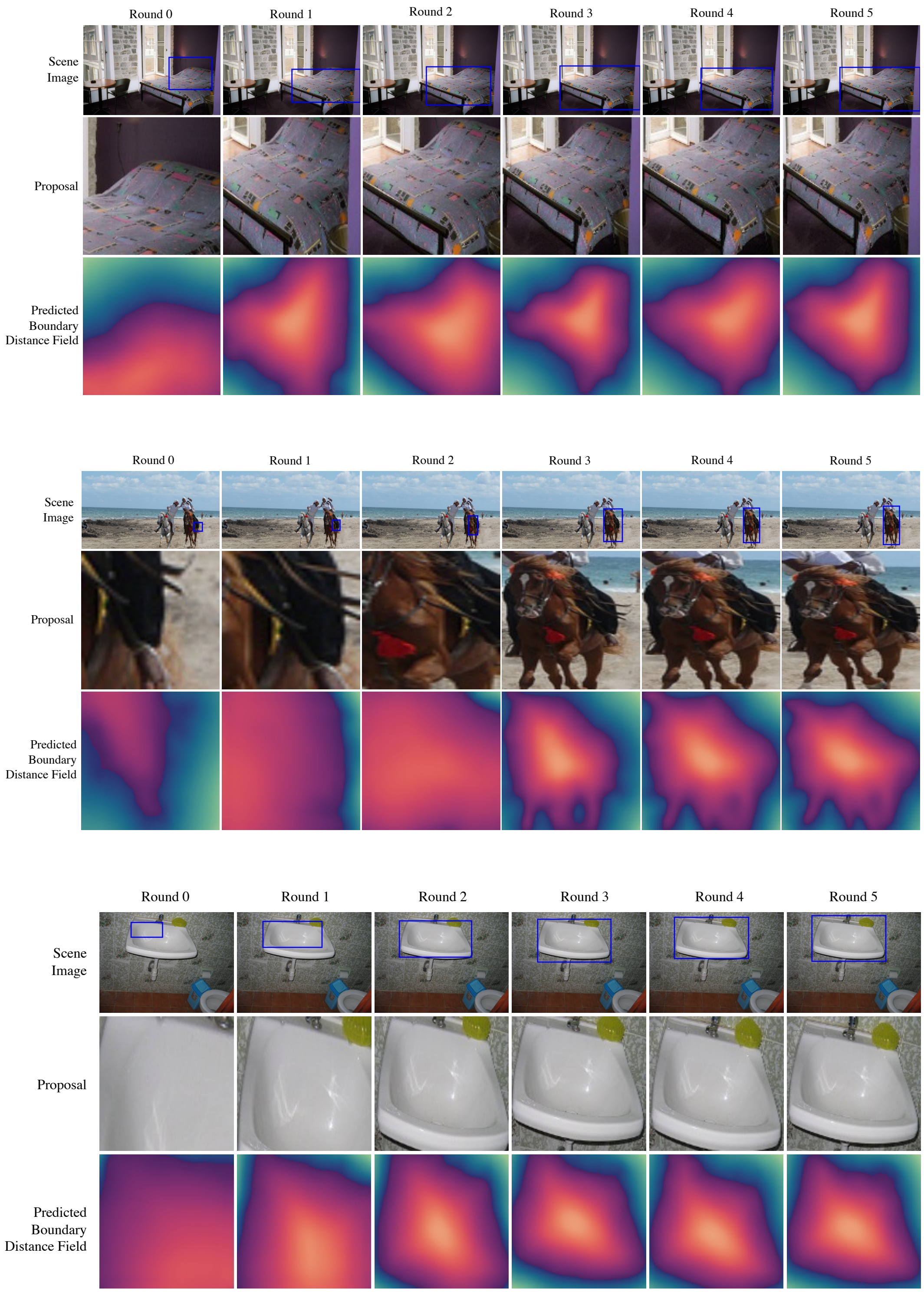


Figure 9: Examples for boundary reasoning.

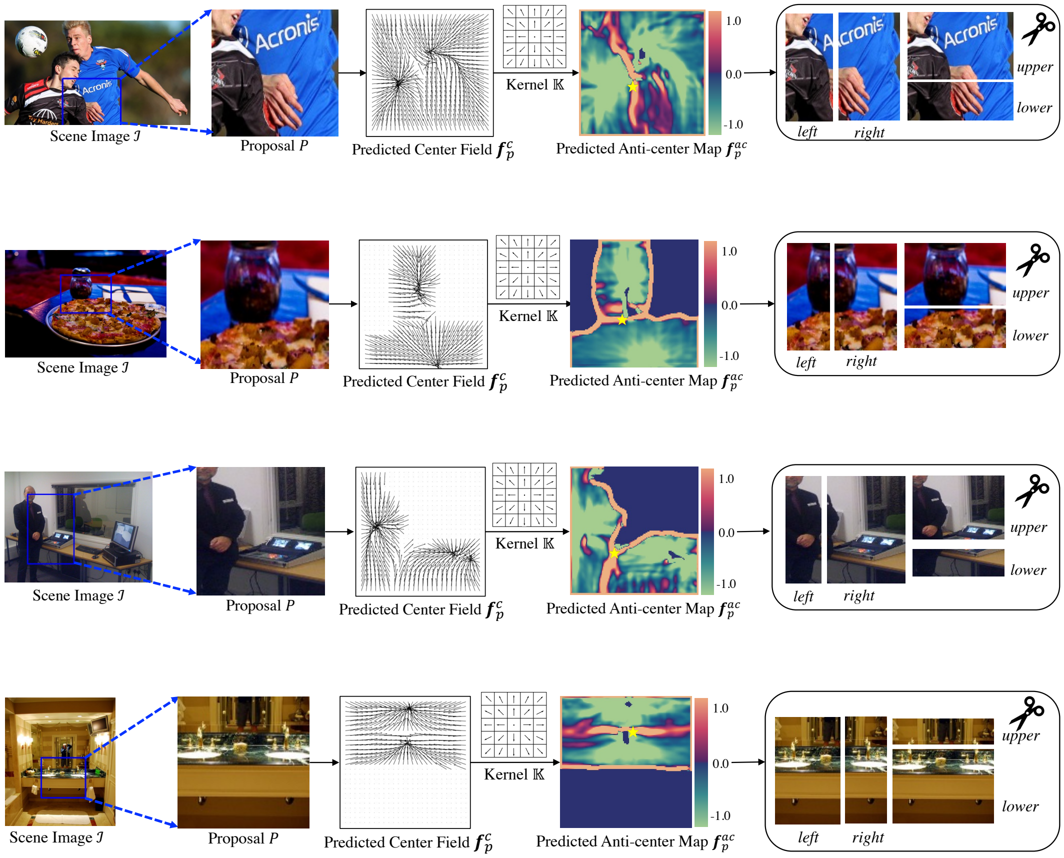


Figure 10: Examples for center reasoning.

Figure 11: Additional qualitative results of Direct Object Discovery as discussed in Section 4.1.

Figure 12: Additional qualitative results of Direct Object Discovery as discussed in Section 4.1.

Figure 13: Additional qualitative results from trained detectors as discussed in Section 4.2.

A.12 DETAILS OF COCO* VALIDATION SET

In COCO*, we exhaustively label objects in the COCO val2017 dataset, which comprises 5,000 images and originally contains 36,781 instances across 90 categories. We have added 197 new object categories and labeled previously unannotated objects within the original COCO categories. In total, COCO* includes 5,000 images, 287 categories, and 47,117 labeled objects. Details for the annotated categories are provided in Table 13.

We use SAM (Kirillov et al., 2023) to expedite the labeling process. We label each object of interest with a tightest bounding box around it. This bounding box, along with the full image, is then fed into the SAM model to generate a dense binary mask.

Table 13: Details of COCO* validation set. This table includes the unique class IDs, class names and the number of newly labeled objects that belong to each class. Specifically, the newly introduced classes are assigned with IDs from 100 to 297. Apart from the 197 new categories, we also label objects belonging to the original COCO classes (the id between 1-90) that are not labeled in COCO validation 2017. In summary, we have labeled 10,336 objects in addition to the original 36,781 objects on COCO validation 2017, resulting in 47,117 objects on 5,000 images.

id	class name	count	*id*	class name	count	*id*	class name	count	*id*	class name	count
3	car	9	*136*	brush	37	*199*	name tag	125	*271*	balance	2
11	fire hydrant	1	*137*	shower	21	*200*	jar	74	*272*	pancake	3
15	bench	6	*138*	beetroot	6	*201*	flag	156	*273*	pepper	8
17	cat	2	*139*	meat	102	*202*	peach	4	*274*	eggplant	2
20	sheep	3	*140*	bridge	11	*203*	radio	5	*275*	napkin	18
33	suitcase	1	*141*	grape	55	*204*	helmet	466	*276*	table stand	3
44	bottle	175	*142*	cheese	10	*205*	cart	32	*277*	kiwifruit	1
47	cup	44	*143*	clothes	102	*206*	toothpaste	14	*278*	fig	1
49	knife	5	*144*	box	186	*207*	coconut	6	*279*	soother	2
50	spoon	8	*145*	curtain	228	*208*	salmon	21	*280*	pomelo	2
51	bowl	17	*146*	beans	15	*209*	tongs	1	*281*	guita	2
53	apple	19	*147*	dustbin	131	*210*	CD player	34	*282*	screen	15
56	broccoli	1	*148*	broom	6	*211*	heater	18	*283*	callbox	2
57	carrot	11	*149*	stand	86	*212*	air conditioner	12	*284*	map	4
59	pizza	4	*150*	statue	69	*213*	butterfly	22	*285*	coffee machine	1
61	cake	12	*151*	fries	16	*214*	tent	15	*286*	dishwasher	1
62	chair	34	*152*	plastic bag	104	*215*	salad	18	*287*	soap stand	1
63	couch	2	*153*	blanket	71	*216*	spagatti	6	*288*	shelf	12
67	dining table	2	*154*	bathtub	38	*217*	gravestone	9	*289*	prize	0
70	toilet	10	*155*	stationary	59	*218*	arcade game machine	1	*290*	tower	5
75	remote	1	*156*	sauce	47	*219*	chips	12	*291*	picture	13
76	keyboard	63	*157*	poster	194	*220*	fish	16	*292*	vent	5
77	cell phone	4	*158*	sail	5	*221*	pig	1	*293*	baggage tag	32
79	oven	11	*159*	rhino	3	*222*	dish	71	*294*	biscuit	7
81	sink	35	*160*	paper	142	*223*	CD	30	*295*	telescope	1
82	refrigerator	1	*161*	hook	28	*224*	doll	29	*296*	pear	5
84	book	18	*162*	hand dryer	1	*225*	watermelon	6	*297*	ferris wheel	2
86	vase	16	*163*	tomato	53	*226*	cherry	4			
101	cabinet	291	*164*	lemon	18	*227*	cream	12			
102	carpet	65	*165*	snail	1	*228*	toy	43			
103	lamp	495	*166*	candle	70	*229*	pomegranate	1			
104	basket	87	*167*	teapot	46	*230*	rolling pin	2			
105	pillow	312	*168*	moon	4	*231*	envolop	3			
106	mirror	67	*169*	strawberry	26	*241*	sticker	51			
107	pot	227	*170*	paperbag	20	*242*	dough	7			
108	lizard	1	*171*	lid	30	*243*	pan	12			
109	scarf	13	*172*	earphone	32	*244*	peanut	1			
110	flower	253	*173*	egg	28	*245*	billboard	154			
111	applicance	82	*174*	butter	10	*246*	ladder	6			
112	can	71	*175*	tap	220	*247*	corn	9			
113	skate shoe	189	*176*	fan	38	*248*	plum	5			
114	glove	143	*177*	switch	128	*249*	MP3 player	6			
115	stove	45	*178*	telephone	34	*250*	garlic	3			
116	watch	38	*179*	socket	114	*251*	scallion	2			
117	ornament	187	*180*	bag	86	*252*	noodle	9			
118	oar	4	*181*	quilt	46	*253*	soup	14			
119	speaker	90	*182*	tank	11	*254*	onion	6			
120	printer	22	*183*	cabbage	24	*255*	sausage	20			
121	monitor	4	*184*	cucumber	39	*256*	vegatable	19			
122	basin	75	*185*	calendar	13	*257*	fishbowl	4			
123	road sign	555	*186*	pinapple	19	*258*	wallet	3			
124	towel	213	*187*	key	11	*259*	buoy	15			
125	ashtray	7	*188*	pumpkin	6	*260*	roadblock	56			
126	plate	190	*189*	ball	15	*261*	chocolate	12			
127	bread	87	*190*	calculator	6	*262*	shell	7			
128	tissue	184	*191*	flashlight	8	*263*	wool	5			
129	rice	27	*192*	usb	13	*264*	avocado	1			
130	painting	445	*193*	potato	15	*265*	charger	9			
131	board	40	*194*	ipad	5	*266*	card	4			
132	ballon	49	*195*	pad	40	*267*	coin	4			
133	camera	71	*196*	banner	174	*268*	wire	9			
134	handler	73	*197*	funnel	3	*269*	piano	6			
135	soap	19	*198*	blender	30	*270*	chinaware	13			

A.13 Representation Comparison

In this section, we provide more insight into the comparison between proposed center-boundary representations with self-supervised features. In particular, we experiment with 4 pre-trained models from DINO and 2 pre-trained models from DINOv2, with different patch sizes and/or model parameter scales.

Motivated by NCut (Shi & Malik, 2000) algorithm, given a set of image features, we construct a weighted graph. The weight on each edge is computed as the similarity between features, formulating an affinity matrix W. Then, we solve an eigenvalue system $(D - W)x = \lambda Dx$ for a set of eigenvectors x and eigenvalues λ, where D is the diagonal matrix. In Figure 14, 15, 16, 17, we visualize the eigenvectors corresponding to the 2nd, 3rd, and 4th smallest eigenvalues. Specifically, we resize all eigenvectors to be the same size as the source image.

In practice, methods like TokenCut (Wang et al., 2023b) and CuVLER (Arica et al., 2024) directly use the eigenvector corresponding to the 2nd smallest eigenvalue and perform clustering onto it.

From Figure 14, 15, 16, 17, we have observed that segmenting objects via grouping pre-trained self-supervised features: 1) focuses on large objects that dominating the image, while ignoring objects with smaller sizes, 2) tends to capture semantic similarity / background-foreground contrast, instead of objectness. For example, in Figure 14, only the ”bed” object with a large size can be discovered by clustering eigenvectors. In Figure 15, the two ”keyboards”, two ”monitors”, and two ”speakers” are hard to be distinguished into separate clusters. Such behaviors are fundamentally due to the training of self-supervised features only involving image-level contrast, which can hardly lead to fine-grained object understanding.

In contrast, as shown in the last row of Figure 14, 15, 16, 17, the proposed center and boundary representation captures more fine-grained properties that directly reflect objectness, which naturally leads to better object discovery results. It should be noted that the merged center field and merged boundary distance field are derived by combining all proposals with their predicted center field and boundary distance field, instead of predicted in one pass.

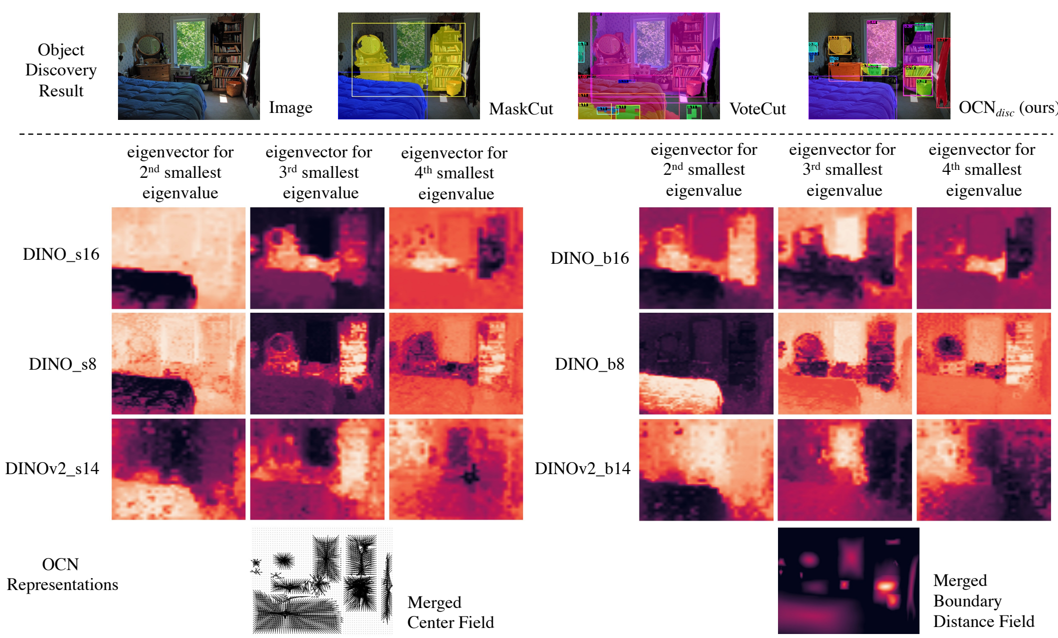


Figure 14: Comparison between DINO/DINOv2 features with proposed boundary-center representations. The eigenvectors are reshaped to be the size of the image. The last row shows the illustrations for the proposed center and boundary distance representations (predicted).

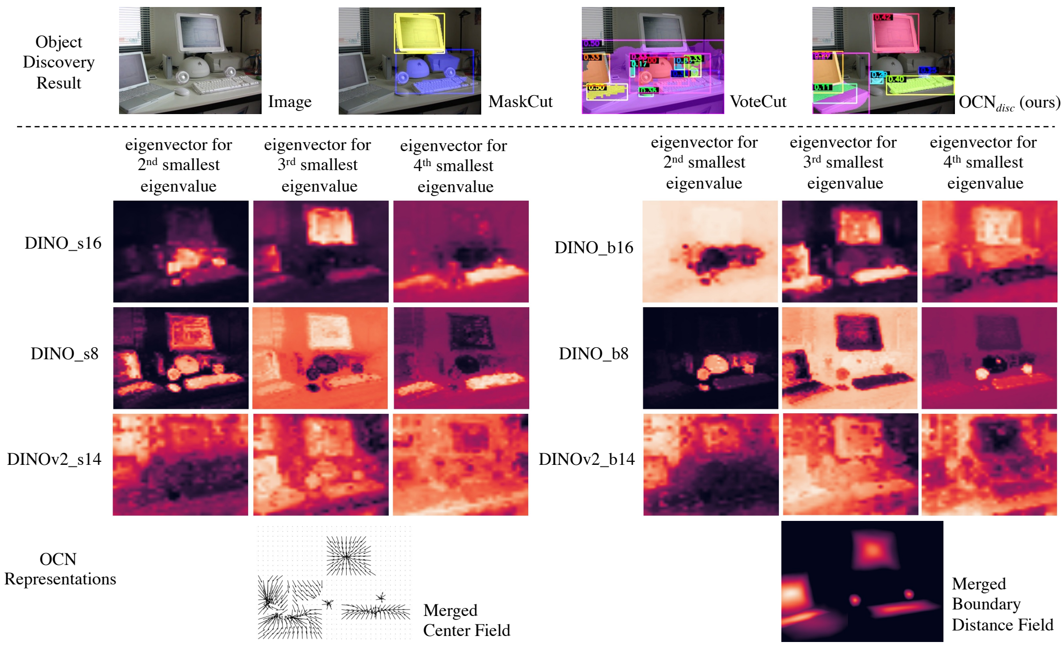


Figure 15: Comparison between DINO/DINOv2 features with proposed boundary-center representations. The eigenvectors are reshaped to be the size of the image. The last row shows the illustrations for the proposed center and boundary distance representations (predicted).

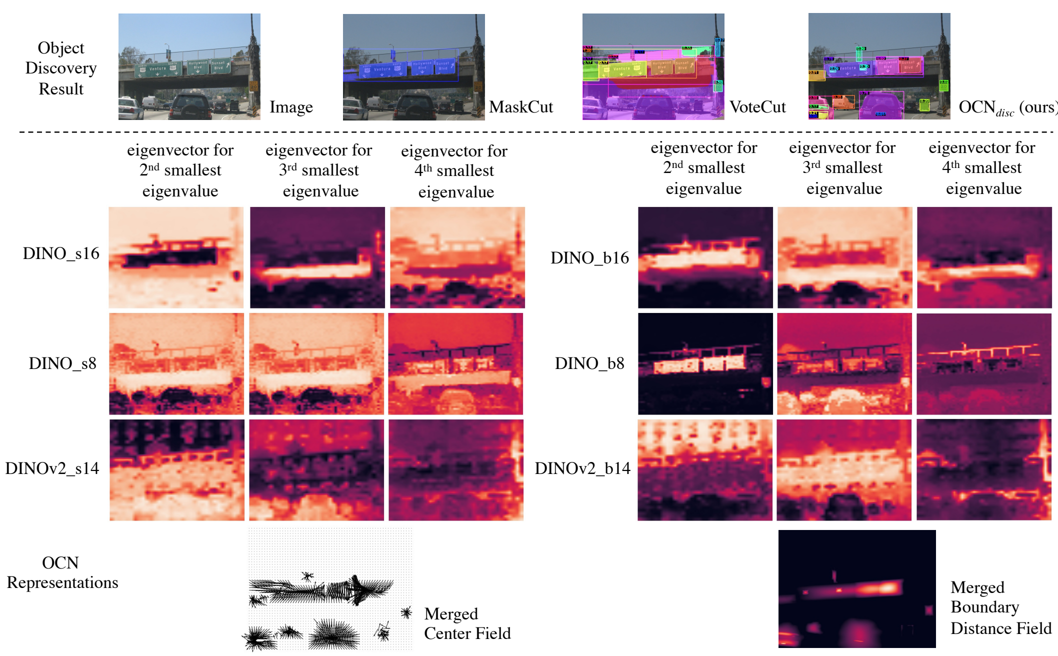


Figure 16: Comparison between DINO/DINOv2 features with proposed boundary-center representations. The eigenvectors are reshaped to be the size of the image. The last row shows the illustrations for the proposed center and boundary distance representations (predicted).

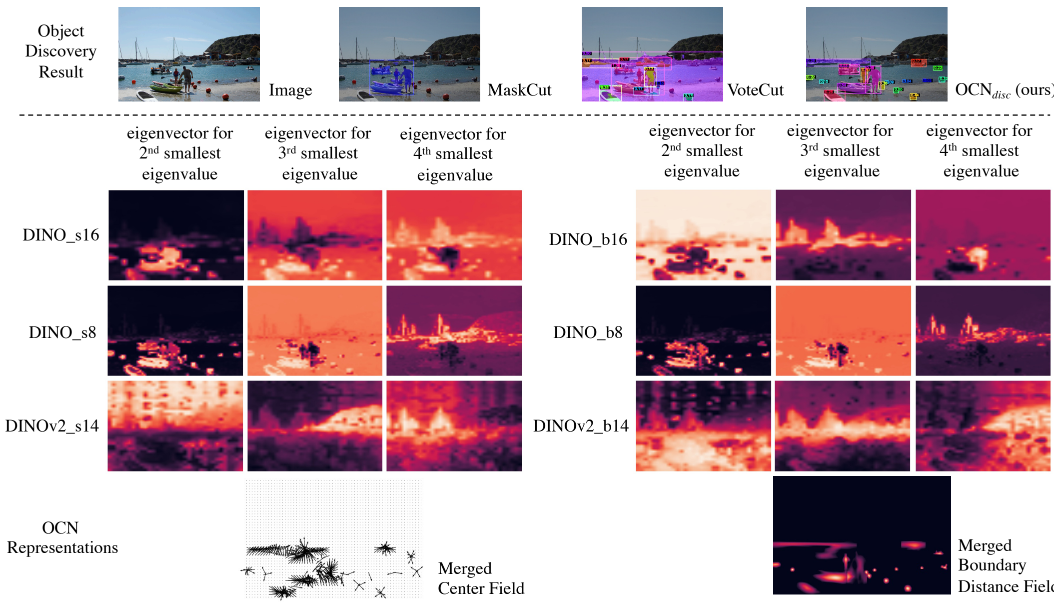


Figure 17: Comparison between DINO/DINOv2 features with proposed boundary-center representations. The eigenvectors are reshaped to be the size of the image. The last row shows the illustrations for the proposed center and boundary distance representations (predicted).

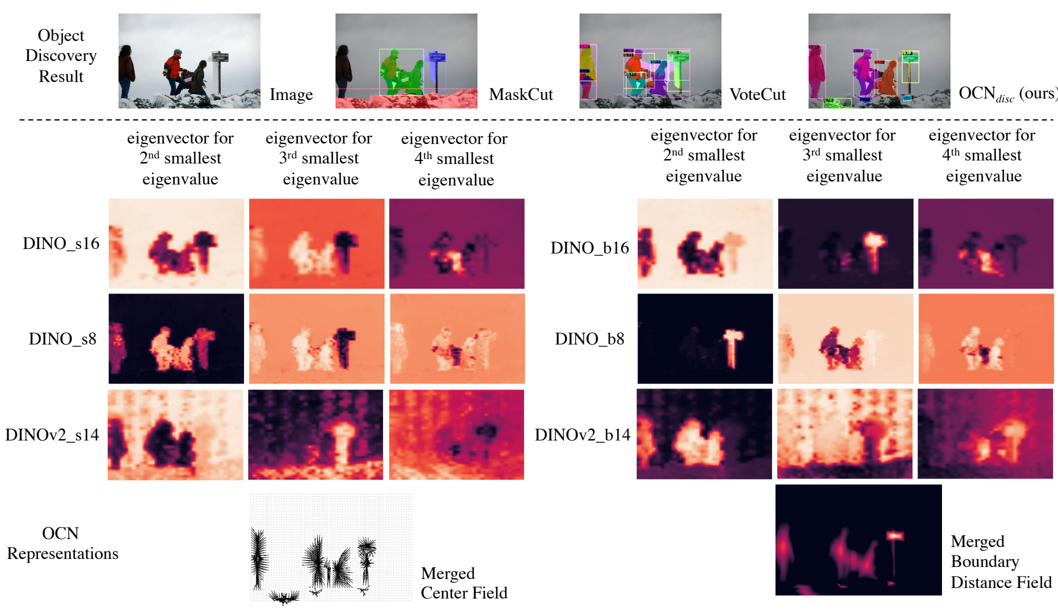


Figure 18: Comparison between DINO/DINOv2 features with proposed boundary-center representations. The eigenvectors are reshaped to be the size of the image. The last row shows the illustrations for the proposed center and boundary distance representations (predicted).

A.14 Efficiency of Direct Object Discovery

For our method of direct object discovery on the COCO* validation set as described in Section 4.1, in implementation, the maximum number of iterations to optimize a proposal is set to be 50. Nevertheless, in practice, as shown in Figure 19 which illustrates the relationship between the average number of pixels to increase or decrease and the number of optimization steps, we observe that all proposals tend to converge after just 10 iterations.

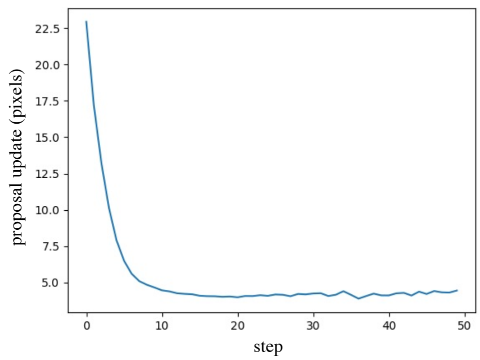


Figure 19: The relationship between the average number of pixels to increase or decrease and the number of optimization steps.

A.15 Additional Results on Medical Images

In order to verify the feasibility of discovering objects on medical images using our OCN$_{disc}$, we evaluate our model on a gland dataset GlaS (Sirinukunwattana et al., 2017). We evaluate on the total 165 images for comparison. Table 14 shows the segmentation results of our OCN$_{disc}$/OCN, MaskCut/CutLER and VoteCut/CuVLER. Under the two settings of direct object discovery and zero-shot detection, our method surpasses CuVLER. Figure 20 shows qualitative results.

Table 14: Gland segmentation results for MaskCut/CutLER, VoteCut/CuVLER, and OCN$_{disc}$/OCN, under direct object discovery and zero-shot detector setting.

		AP_{50}^{mask}	AP_{75}^{mask}	AP^{mask}	AR_{100}^{mask}
MaskCut (K=3)	direct object discovery	0.4	0.1	0.2	0.8
MaskCut (K=10)	direct object discovery	0.4	0.1	0.2	0.9
CutLER	zero-shot detector	8.8	1.0	2.6	21.5
VoteCut	direct object discovery	0.8	0.0	0.2	1.9
CuVLER	zero-shot detector	3.2	0.2	0.7	11.1
OCN$_{disc}$(Ours)	direct object discovery	3.3	1.6	1.7	6.8
OCN (Ours)	zero-shot detector	9.6	1.2	2.9	18.9

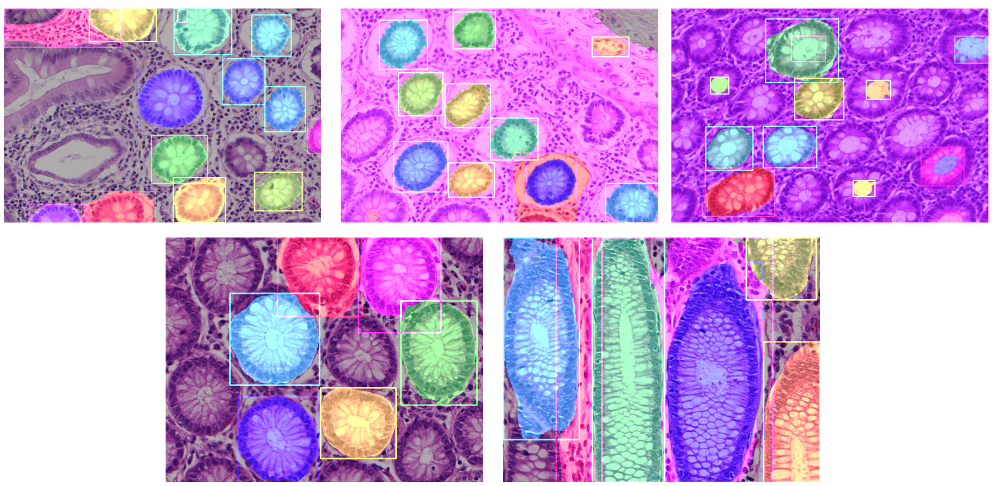


Figure 20: Qualitative results of our OCN_{disc} for direct object discovery on the GlaS dataset (Sirinukunwattana et al., 2017).