Cut and Learn for Unsupervised Object Detection and Instance Segmentation

Xudong Wang[1,2] Rohit Girdhar[1] Stella X. Yu[2,3] Ishan Misra[1]
[1]FAIR, Meta AI [2]UC Berkeley / ICSI [3]University of Michigan

Code: https://github.com/facebookresearch/CutLER

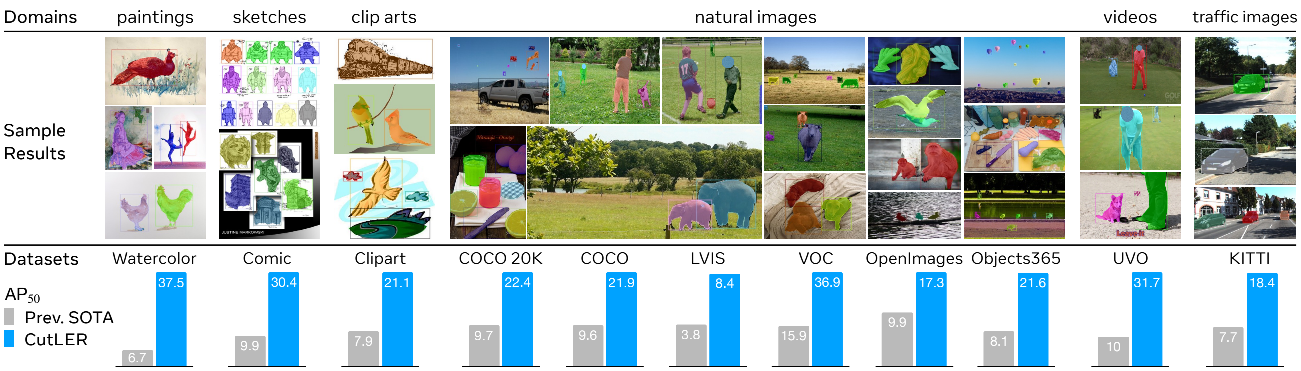

Figure 1. Zero-shot unsupervised object detection and instance segmentation using our CutLER model, which is trained without human supervision. We evaluate the model using the standard detection $\text{AP}_{50}^{\text{box}}$. CutLER gives a strong performance on a variety of benchmarks spanning diverse image domains - video frames, paintings, clip arts, complex scenes, *etc*. Compared to the previous state-of-the-art method, FreeSOLO [47] with a backbone of ResNet101, CutLER with a backbone of ResNet50 provides strong gains on all benchmarks, increasing performance by more than 2× on 10 of the 11 benchmarks. We evaluate [47] with its official code and checkpoint.

Abstract

We propose ***Cut****-and-****LE****a****R****n (CutLER), a simple approach for training unsupervised object detection and segmentation models. We leverage the property of self-supervised models to 'discover' objects without supervision and amplify it to train a state-of-the-art localization model without any human labels. CutLER first uses our proposed MaskCut approach to generate coarse masks for multiple objects in an image, and then learns a detector on these masks using our robust loss function. We further improve performance by self-training the model on its predictions. Compared to prior work, CutLER is simpler, compatible with different detection architectures, and detects multiple objects. CutLER is also a zero-shot unsupervised detector and improves detection performance AP*$_{50}$ *by over 2.7× on 11 benchmarks across domains like video frames, paintings, sketches,* etc. *With finetuning, CutLER serves as a low-shot detector surpassing MoCo-v2 by 7.3% AP*^{box} *and 6.6% AP*^{mask} *on COCO when training with 5% labels.*

1. Introduction

Object localization is a critical task in computer vision that enables AI systems to perceive, reason, plan and act in an object-centric manner. Training models for localization require special annotations like object boxes, masks, localized points, *etc*. which are both difficult and resource intensive to collect. Without accounting for overhead, annotating ∼164K images in the COCO dataset [32] with masks for just 80 classes took more than 28K human hours of annotation time. In this work, we study unsupervised object detection and instance segmentation models that can be trained without any human labels. Our key insight is that simple probing and training mechanisms can amplify the innate localization ability of self-supervised models [7], leading to state-of-the-art unsupervised zero-shot detectors.

Our method **Cut**-and-**LE**a**R**n (CutLER) consists of three simple, architecture- and data-agnostic mechanisms. Consistent with prior self-supervised learning methods [7–9, 26], CutLER is trained exclusively on unlabeled ImageNet data without needing additional training data, but contrary to these methods, CutLER can be directly employed to perform complex segmentation and detection tasks over a wide range of domains. *First*, we propose MaskCut that can automatically produce *multiple* initial coarse masks for each image, using the pretrained self-supervised features. *Second*, we propose a simple loss dropping strategy to train detectors using the coarse masks while being robust to objects missed by MaskCut. *Finally*, we observe that despite learning from these coarse masks, the detectors 'clean' the ground truth and produce masks (and boxes) that are better than the coarse masks used to train them. Therefore, we further show that multiple rounds of self-training on the models' own predictions allow it to evolve from capturing the similarity of local pixels to capturing the global geometry of the object, thus producing finer segmentation masks.

Prior work shows that a self-supervised vision transformer (ViT) [15] can automatically learn patch-wise features that detect a single *salient* object in an image [7,38,43,44,50]. However, unlike CutLER, such salient object detection methods only locate a single, usually the most prominent, object and cannot be used for real world images containing multiple objects. While some recent methods, *e.g.*, FreeSOLO [47] and DETReg [3], also aim at unsupervised multi-object detection (or multi-object discovery), they rely on a particular detection architecture, *e.g.*, SOLO-v2 [48] or DDETR [5,54]. Additionally, apart from self-supervised features trained on ImageNet [12], the current state-of-the-art methods FreeSOLO and MaskDistill [42] also require 'in-domain' unlabeled data for model training.

In contrast, CutLER works with various detection architectures and can be trained solely on ImageNet, without requiring in-domain unlabeled data. Thus, during model training, CutLER does not see any images from any target dataset and yields a zero-shot model capable of detecting and segmenting multiple objects in diverse domains.

Features of CutLER. ***1) Simplicity:*** CutLER is simple to train and agnostic to the choice of detection and backbone architectures. Thus, it can be integrated effortlessly into existing object detection and instance segmentation works. ***2) Zero-shot detector:*** CutLER trained solely on ImageNet shows strong zero-shot performance on 11 different benchmarks where it outperforms prior work trained with additional in-domain data. We **double the $\mathbf{AP}^{\text{box}}_{50}$** performance on 10 of these benchmarks, as shown in Fig. 1, and even outperform supervised detectors on the UVO video instance segmentation benchmark. ***3) Robustness:*** CutLER exhibits strong robustness against domain shifts when tested on images from different domains such as video frames, sketches, paintings, clip arts, *etc*. ***4) Pretraining for supervised detection:*** CutLER can also serve as a pretrained model for training fully supervised object detection and instance segmentation models and improves performance on COCO, including on few-shot object detection benchmarks.

2. Related Work

Self-supervised feature learning involves inferring the patterns within the large-scale unlabeled data without using human-annotated labels. *Contrastive learning based* [8,26,34,52] methods learn such representations that similar samples or various augmentations of the same instance are close to each other, while dissimilar instances are far apart. *Similarity-based self-supervised learning* methods [10,23] learn representations via minimizing the distance between different augmentations of the same instance and use only positive sample pairs. *Clustering-based feature learning* [1,6,46,53,55] automatically discovers the natural grouping of data in the latent representation space. Recently, [2,25] have shown that *masked autoencoders* [2,13,14,25] are scalable self-supervised learners for computer vision [25].

In contrast to these unsupervised representation learning efforts, our work aims to automatically discover natural pixel groupings and locate instances within each image.

	DINO	LOST	TokenCut	FreeSOLO	Ours
detect multiple objects	✗	✓	✗	✓	✓
zero-shot detector	✓	✗	✓	✗	✓
compatible with various detection architectures	-	✓	-	✗	✓
pretrained model for supervised detection	✓	✗	✗	✓	✓

Table 1. We compare previous methods on unsupervised object detection, including DINO [7], LOST [38], TokenCut [50] and FreeSOLO [47], with our CutLER in term of key properties. Our CutLER is the only method with all these desired properties.

Unsupervised object detection and instance segmentation. The main comparisons to previous works are listed in Table 1 and are elaborated as follows:

DINO [7] observes that the underlying semantic segmentation of images can emerge from the self-supervised Vision Transformer (ViT) [15], which does not appear explicitly in either supervised ViT or ConvNets [7,56]. Based on this observation, LOST [38] and TokenCut [50] leverage self-supervised ViT features and propose to segment *one single* salient object [11,38,50] from each image based on a graph that is constructed with DINO's patch features.

These previous works either can not detect more than one object from each image, *e.g.*, DINO and TokenCut, or can not improve the quality of features for better transfer to downstream detection and segmentation tasks, *e.g.*, TokenCut and LOST. Unlike these works, CutLER can locate multiple objects and serve as a pretrained model for label-efficient and fully-supervised learning.

FreeSOLO [47] performs unsupervised instance segmentation by extracting coarse object masks in an unsupervised manner, followed by mask refinement through a self-training procedure. While FreeSOLO's FreeMask stage can generate multiple coarse masks per image, the quality of these masks is often rather low [47]. MaskDistill [42] distills class-agnostic initial masks from the affinity graph produced by a self-supervised DINO [7]. However, it utilizes *one single* mask per image in the distillation stage, which greatly limits the model's ability to detect multiple objects.

By contrast, the initial masks generated by our MaskCut are usually better in quality and quantity than the initial masks used by [42,47]. Therefore, CutLER achieves

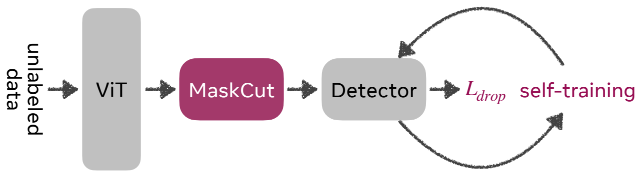

Figure 2. Overview of CutLER. We propose a simple yet effective method to train an object detection and instance segmentation model without using any supervision. We first propose MaskCut to extract initial coarse masks from the features of a self-supervised ViT. We then learn a detector using our loss dropping strategy that is robust to objects missed by MaskCut. We further improve the model using multiple rounds of self-training.

2×~4× higher AP^box and AP^mask than FreeSOLO [47] and MaskDistill [42] on almost all experimented detection and segmentation benchmarks, even when [42, 47] are trained and tested on the same domain.

3. Method

We tackle the problem of unsupervised object detection and segmentation with a simple cut-and-learn pipeline. Our method builds upon insights from recent work [7,50], showing that self-supervised representations can discover objects. While these methods often find a single object per image, we propose a simple approach that can discover multiple objects and significantly improves segmentation and detection performance. The overview of our cut-and-learn pipeline is illustrated in Fig. 2. *First*, we propose MaskCut that generates multiple binary masks per image using self-supervised features from DINO [7] (Sec. 3.2). *Second*, we show a dynamic loss dropping strategy, called DropLoss, that can learn a detector from MaskCut's initial masks while encouraging the model to explore objects missed by MaskCut (Sec. 3.3); *Third*, we further improve the performance of our method through multiple rounds of self-training (Sec. 3.4).

3.1. Preliminaries

Normalized Cuts (NCut) treats the image segmentation problem as a graph partitioning task [37]. We construct a fully connected undirected graph via representing each image as a node. Each pair of nodes is connected by edges with weights W_{ij} that measure the similarity of the connected nodes. NCut minimizes the cost of partitioning the graph into two sub-graphs, *i.e.*, a bipartition, by solving a generalized eigenvalue system

$$(D - W)x = \lambda D x \tag{1}$$

for finding the eigenvector x that corresponds to the second smallest eigenvalue λ, where D is a $N \times N$ diagonal matrix with $d(i) = \sum_j W_{ij}$ and W is a $N \times N$ symmetrical matrix.

DINO and TokenCut. DINO [7] finds that the self-supervised ViT can automatically learn a certain degree of perceptual grouping of image patches. TokenCut [50] leverages the DINO features for NCut and obtaining foreground/background segments in an image. The authors use the similarity of the patches in the DINO feature space as the similarity weight W_{ij} in NCut. Specifically, following multiple recent methods [38, 42, 50], we use the cosine similarity of 'key' features from the last attention layer of DINO-pretrained model, *i.e.*, $W_{ij} = \frac{K_i K_j}{\|K_i\|_2 \|K_j\|_2}$ where K_i is the 'key' feature of patch i, and solve Eq. (1) for finding the second smallest eigenvector x.

A limitation of TokenCut is that it only computes a single binary mask for an image and thus only finds one object per image. Although we can use the other $N-2$ smallest eigenvectors to locate more than one instance, this significantly degrades the performance for multi-object discovery, as demonstrated in Sec. 5.

3.2. MaskCut for Discovering Multiple Objects

As we discussed in Sec. 3.1, vanilla NCut is limited to discovering a single object in an image. We propose MaskCut that extends NCut to discover multiple objects per image by iteratively applying NCut to a *masked* similarity matrix (illustrated in Fig. 3). After getting the bipartition x^t from NCut at stage t, we get two disjoint groups of patches and construct a binary mask M^t, where

$$M^t_{ij} = \begin{cases} 1, & \text{if } M^t_{ij} \geq \text{mean}(x^t) \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

To determine which group corresponds to the foreground, we make use of two criteria: *1)* intuitively, the foreground patches should be more prominent than background patches [7, 43, 50]. Therefore, the foreground mask should contain the patch corresponding to the maximum *absolute* value in the second smallest eigenvector M^t; *2)* we incorporate a simple but empirically effective object-centric prior [33]: the foreground set should contain less than two of the four corners. We reverse the partitioning of the foreground and background, *i.e.*, $M^t_{ij} = 1 - M^t_{ij}$, if the criteria 1 is not satisfied while the current foreground set contains two corners or the criteria 2 is not satisfied. In practice, we also set all $W_{ij} < \tau^{\text{ncut}}$ to $1e^{-5}$ and $W_{ij} \geq \tau^{\text{ncut}}$ to 1.

To get a mask for the $(t+1)^{\text{th}}$ object, we update the node similarity W^{t+1}_{ij} via masking out these nodes corresponding to the foreground in previous stages:

$$W^{t+1}_{ij} = \frac{(K_i \prod_{s=1}^{t} \hat{M}^s_{ij})(K_j \prod_{s=1}^{t} \hat{M}^s_{ij})}{\|K_i\|_2 \|K_j\|_2} \tag{3}$$

where $\hat{M}^s_{ij} = 1 - M^s_{ij}$. Using the updated W^{t+1}_{ij}, we repeat Eqs. (1) and (2) to get a mask M^{t+1}. We repeat this process t times and set $t = 3$ by default.

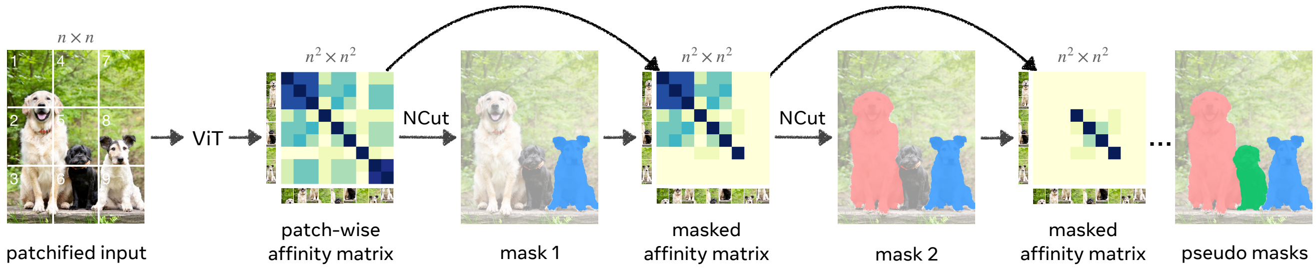

Figure 3. MaskCut can discover multiple object masks in an image without supervision. We build upon [7, 50] and create a patch-wise similarity matrix for the image using a self-supervised DINO [7] model's features. We apply Normalized Cuts [37] to this matrix and obtain a single foreground object mask of the image. We then mask out the affinity matrix values using the foreground mask and repeat the process, which allows MaskCut to discover multiple object masks in a single image. In this pipeline illustration, we set $n = 3$.

3.3. DropLoss for Exploring Image Regions

A standard detection loss penalizes predicted regions r_i that do not overlap with the 'ground-truth'. Since the 'ground-truth' masks given by MaskCut may miss instances, the standard loss does not enable the detector to discover new instances not labeled in the 'ground-truth'. Therefore, we propose to ignore the loss of predicted regions r_i that have a small overlap with the 'ground-truth'. More specifically, during training, we drop the loss for each predicted region r_i that has a maximum overlap of τ^{IoU} with any of the 'ground-truth' instances:

$$\mathcal{L}_{\text{drop}}(r_i) = \mathbb{1}(\text{IoU}_i^{\text{max}} > \tau^{\text{IoU}})\mathcal{L}_{\text{vanilla}}(r_i) \tag{4}$$

where IoU_i^{max} denotes the maximum IoU with 'ground-truth' for r_i and $\mathcal{L}_{\text{vanilla}}$ refers to the vanilla loss function of detectors. $\mathcal{L}_{\text{drop}}$ does not penalize the model for detecting objects missed in the 'ground-truth' and thus encourages the exploration of different image regions. In practice, we use a low threshold $\tau^{\text{IoU}} = 0.01$.

3.4. Multi-Round Self-Training

Empirically, we find that despite learning from the coarse masks obtained by MaskCut, detection models 'clean' the ground truth and produce masks (and boxes) that are better than the initial coarse masks used for training. The detectors refine mask quality, and our DropLoss strategy encourages them to discover new object masks. Thus, we leverage this property and use multiple rounds of self-training to improve the detector's performance.

We use the predicted masks and proposals with a confidence score over $0.75-0.5t$ from the t^{th}-round as the additional pseudo annotations for the $(t+1)^{\text{th}}$-round of self-training. To de-duplicate the predictions and the ground truth from round t, we filter out ground-truth masks with an IoU > 0.5 with the predicted masks. We found that three rounds of self-training are sufficient to obtain good performance. Each round steadily increases the number of 'ground-truth' samples used to train the model.

3.5. Implementation Details

Training data. We only use the images from the ImageNet [12] dataset (1.3 million images) for all parts of the CutLER model and do not use any type of annotations either for training or any supervised pretrained models.

MaskCut. We use MaskCut with three stages on images resized to 480×480 pixels and compute a patch-wise affinity matrix using the ViT-B/8 [15] DINO [7] model. We use Conditional Random Field (CRF) [30] to post-process the masks and compute their bounding boxes.

Detector. While CutLER is agnostic to the underlying detector, we use popular Mask R-CNN [27] and Cascade Mask R-CNN [4] for all experiments, and use Cascade Mask R-CNN by default, unless otherwise noted. We train the detector on ImageNet with initial masks and bounding boxes for 160K iterations with a batch size of 16. When training the detectors with a ResNet-50 backbone [28], we initialize the model with the weights of a self-supervised pretrained DINO [7] model. We explored other pre-trained models, including MoCo-v2 [9], SwAV [6], and CLD [46], and found that they gave similar detection performance. We also leverage the copy-paste augmentation [16, 19] during the model training process. Rather than using the vanilla copy-paste augmentation, to improve the model's ability to segment small objects, we randomly downsample the mask with a scalar uniformly sampled between 0.3 and 1.0. We then optimize the detector for 160K iterations using SGD with a learning rate of 0.005, which is decreased by 5 after 80K iterations, and a batch size of 16. We apply a weight decay of 5×10^{-5} and a momentum of 0.9.

Self-training. We initialize the detection model in each stage using the weights from the previous stage. We optimize the detector using SGD with a learning rate of 0.01 for $80K$ iterations. Since the self-training stage can provide a sufficient number of pseudo-masks for model training, we don't use the DropLoss during the self-training stages.

We provide more details on model implementation and training in Appendix A.1.

Datasets →	Avg.		COCO		COCO20K		VOC		LVIS		UVO		Clipart		Comic		Watercolor		KITTI		Objects365		OpenImages	
Metrics →	AP_{50}	AR	AP_{50}	AR	AP_{50}	AR	AP_{50}	AR	AP_{50}	AR	AP_{50}	AR	AP_{50}	AR	AP_{50}	AR	AP_{50}	AR	AP_{50}	AR	AP_{50}	AR	AP_{50}	AR
Prev. SOTA [47]	9.0	13.4	9.6	12.6	9.7	12.6	15.9	21.3	3.8	6.4	10.0	14.2	7.9	15.1	9.9	16.3	6.7	16.2	7.7	7.1	8.1	10.2	9.9	14.9
CutLER	24.3	35.5	21.9	32.7	22.4	33.1	36.9	44.3	8.4	21.8	31.7	42.8	21.1	41.3	30.4	38.6	37.5	44.6	18.4	27.5	21.6	34.2	17.3	29.6
vs. prev. SOTA	**+15.3**	**+22.1**	**+12.3**	**+20.1**	**+12.7**	**+20.5**	**+21.0**	**+23.0**	**+4.6**	**+15.4**	**+21.7**	**+28.6**	**+13.2**	**+26.2**	**+20.5**	**+22.3**	**+30.8**	**+28.4**	**+10.7**	**+20.4**	**+13.5**	**+24.0**	**+7.4**	**+14.7**

Table 2. State-of-the-art **zero-shot unsupervised object detection** performance on 11 different datasets spanning a variety of domains. We report class-agnostic multi-object detection performance and the averaged results for 11 datasets using AP^{box}_{50} and AR^{box}_{100}. Our CutLER is trained in an unsupervised manner solely on ImageNet. While the previous SOTA method [47] is typically fine-tuned on extra data, *e.g.*, ∼241k unlabeled COCO images, CutLER significantly outperforms it. Results of [47] are produced with official code and checkpoint.

4. Experiments

We evaluate CutLER on various detection and segmentation benchmarks. In Sec. 4.1, we show that CutLER can discover objects without any supervision on completely unseen images. Despite being evaluated in a zero-shot manner on eleven benchmarks, CutLER outperforms prior methods that use in-domain training data. Sec. 4.2 shows that fine-tuning CutLER further improves detection performance, outperforming prior work like MoCo-V2 and FreeSOLO.

4.1. Unsupervised Zero-shot Evaluations

We conduct extensive experiments on eleven different datasets, covering various object categories, image styles, video frames, resolutions, camera angles, *etc*. to verify the effectiveness of CutLER as a universal unsupervised object detection and segmentation method. We describe the different datasets used for zero-shot evaluation in detail in Appendix A.2. CutLER is trained solely using images from ImageNet and evaluated in a zero-shot manner on all downstream datasets without finetuning on any labels or data.

Evaluating unsupervised object detectors poses two unique challenges. *First*, since the model is trained without any notion of semantic classes, it cannot be evaluated using the class-aware detection setup. Thus, like prior work [3, 38, 48] we use the class-agnostic detection evaluation. *Second*, object detection datasets often only annotate a subset of the objects in the images. For example, while COCO and LVIS use the same images, COCO only labels 80 object classes, and LVIS labels 1203 object classes. In this partially labeled setup, Average Recall (AR) is a valuable metric for unsupervised detection as it does not penalize the models for detecting novel objects unlabeled in the dataset. Thus, we additionally report AR for all datasets.

Zero-shot detection on 11 benchmarks. We evaluate CutLER on a variety of datasets and report the detection performance using AP^{box}_{50} and AR^{box}_{100} metrics in Fig. 1 and Table 2. CutLER uses a *smaller* model size and *less* training data than prior work. Compared to the previous SOTA approach, FreeSOLO [47] with a backbone of ResNet101, CutLER, with the smaller ResNet50 backbone, significantly outperforms it in each of these benchmarks spanning various image distributions, more than doubling performance on 10 of them. Also note that, FreeSOLO requires FreeMask pre-

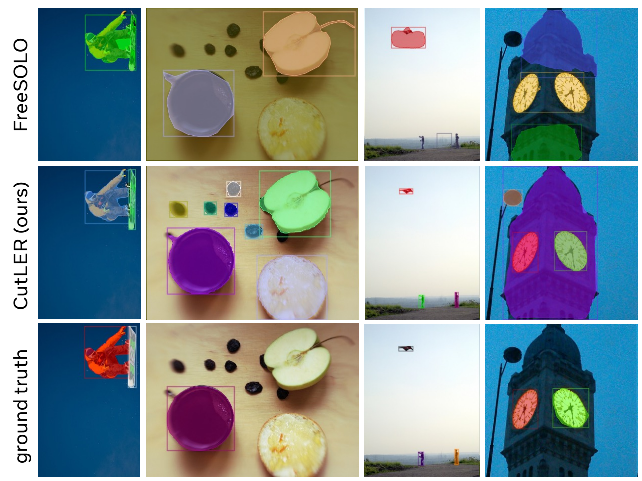

Figure 4. *Compared to the previous state-of-the-art* [47], our CutLER can better discriminate instances (*e.g.* person and skis in col. 1), discover more objects (*e.g.* apple and raisins in col. 2), and produce higher quality segmentation masks even for small objects (*e.g.* kite in col. 3); *compared to human annotations*, CutLER can locate novel instances that are overlooked by human annotators, such as the streetlight and clock tower in col. 4. **Qualitative comparisons** between previous SOTA methods (row 1) and our CutLER (row 2) on COCO, as well as ground truth annotations by human annotators (row 3), are visualized.

training using approximately 1.3M ImageNet images and model fine-tuning using additional data in test benchmarks.

We observe that on different domains, *e.g.* watercolor or frames from videos (UVO dataset), CutLER improves performance by over 4× and 2×, respectively. Fig. 1 shows some qualitative examples of CutLER's predictions.

Detailed comparisons on COCO20K and COCO. Table 3 presents detailed detection and segmentation evaluations (also referred to as 'multi-object' discovery) on two popular benchmarks: COCO `val2017` [32] and COCO `20K`, which contains a subset of 20K images of COCO [38, 47]. CutLER consistently surpasses prior works by a large margin (often gets 2∼3× higher AP) on both the segmentation and detection tasks. Although CutLER is not trained on any images from COCO, it surpasses existing methods trained on COCO by more than 10% in terms of AP^{mask}_{50} and AP^{box}_{50}.

Fig. 4 shows the qualitative comparisons between [47]

Methods	Pretrain	Detector	Init.	COCO 20K						COCO val2017					
				AP_{50}^{box}	AP_{75}^{box}	AP^{box}	AP_{50}^{mask}	AP_{75}^{mask}	AP^{mask}	AP_{50}^{box}	AP_{75}^{box}	AP^{box}	AP_{50}^{mask}	AP_{75}^{mask}	AP^{mask}
non zero-shot methods															
LOST [38]	IN+COCO	FRCNN	DINO	-	-	-	2.4	1.0	1.1	-	-	-	-	-	-
MaskDistill [42]	IN+COCO	MRCNN	MoCo	-	-	-	6.8	2.1	2.9	-	-	-	-	-	-
FreeSOLO* [47]	IN+COCO	SOLOv2	DenseCL	9.7	3.2	4.1	9.7	3.4	4.3	9.6	3.1	4.2	9.4	3.3	4.3
zero-shot methods															
DETReg [3]	IN	DDETR	SwAV	-	-	-	-	-	-	3.1	0.6	1.0	8.8	1.9	3.3
DINO [7]	IN	-	DINO	1.7	0.1	0.3	-	-	-	-	-	-	-	-	-
TokenCut [50]	IN	-	DINO	-	-	-	-	-	-	5.8	2.8	3.0	4.8	1.9	2.4
CutLER (ours)	IN	MRCNN	DINO	21.8	11.1	10.1	18.6	9.0	8.0	21.3	11.1	10.2	18.0	8.9	7.9
CutLER (ours)	IN	Cascade	DINO	22.4	12.5	11.9	19.6	10.0	9.2	21.9	11.8	12.3	18.9	9.7	9.2
vs. prev. SOTA				**+12.7**	**+9.3**	**+7.8**	**+9.9**	**+6.6**	**+4.9**	**+12.3**	**+8.7**	**+8.1**	**+9.5**	**+6.4**	**+4.9**

Table 3. Unsupervised object detection and instance segmentation on COCO `20K` and COCO `val2017`. We report the detection and segmentation metrics and note the pretraining data (Pretrain), detectors, and backbone initialization (Init.). Methods in the top half of the table train on extra unlabeled images from the downstream datasets, while zero-shot methods in the bottom half only train on ImageNet. Despite using an older detector, CutLER outperforms all prior works on all evaluation metrics. *: results obtained with the official code and checkpoint. IN, Cascade, MRCNN, and FRCNN denote ImageNet, Cascade Mask R-CNN, Mask R-CNN, and Faster R-CNN, respectively.

Methods	AP_{50}	AP_{75}	AP	AP_S	AP_M	AP_L
rOSD [43]	13.1	-	4.3	-	-	-
LOD [44]	13.9	-	4.5	-	-	-
LOST [38]	19.8	-	6.7	-	-	-
FreeSOLO* [47]	15.9	3.6	5.9	0.0	2.0	9.3
CutLER (ours)	36.9	19.2	20.2	1.3	6.5	32.2
vs. prev. SOTA	**+17.1**	**+15.6**	**+13.5**	**+1.3**	**+4.5**	**+22.9**

Table 4. Zero-shot unsupervised object detection on **VOC**. *: reproduced results with official code and checkpoint.

Methods	AP_{50}^{box}	AP_{75}^{box}	AP^{box}	AP_{50}^{mask}	AP_{75}^{mask}	AP^{mask}
fully-supervised methods:						
SOLO-v2 (w/ COCO) [48]	-	-	-	38.0	20.9	21.4
Mask R-CNN (w/ COCO) [27]	-	-	-	31.0	14.2	15.9
SOLO-v2 (w/ LVIS) [48]	-	-	-	14.8	5.9	7.1
unsupervised methods:						
FreeSOLO* [47]	10.0	1.8	3.2	9.5	2.0	3.3
CutLER (ours)	31.7	14.1	16.1	22.8	8.0	10.1
vs. prev. SOTA	**+21.7**	**+12.3**	**+12.9**	**+13.3**	**+6.0**	**+6.8**

Table 5. Zero-shot unsupervised object detection and instance segmentation on the **UVO `val` video benchmark**. CutLER outperforms prior unsupervised methods and achieves better performance than the supervised SOLO-v2 model trained on the LVIS dataset. *: reproduced results with official code and checkpoint.

and our CutLER on COCO `val2017`, along with human annotations. Surprisingly, *CutLER can often detect novel instances that human annotators miss*. We present detailed comparisons on COCO `20K`, COCO `val2017` and LVIS [24] benchmarks in Appendix A.3.

Detailed comparisons on UVO and VOC. For a comprehensive comparison with existing unsupervised multi-object detection methods, we report the results for UVO `val` [45] and VOC `trainval07` [17]. Table 4 shows that CutLER yields significant performance gains over previous SOTA, obtaining over 3× higher AP, with the most considerable improvement coming from AP_L. On UVO, Table 5 shows that CutLER more than quadruples the AP of previous SOTA and almost triples the AP_{50}^{box}. *Our* AP_{50}^{mask} *is even 4.8% higher than the fully-supervised SOLOv2 [48] trained on LVIS with 100% annotations*, significantly narrowing the gap between supervised and unsupervised learning.

4.2. Label-Efficient and Fully-Supervised Learning

We now evaluate CutLER as a pretraining method for training object detection and instance segmentation models. While CutLER can discover objects without any supervision, finetuning it on a target dataset aligns the model output to the same set of objects labeled in the dataset.

Setup. We use CutLER to initialize a standard Cascade Mask R-CNN [4] detector with a ResNet50 [28]. Prior work uses more advanced detectors, SOLOv2 [48] used in [47] and DDETR [54] used in [3], that perform better. However, we choose Cascade Mask R-CNN for its simplicity and show in Sec. 5 that CutLER's performance improves with stronger detectors. We train the detector on the COCO [32] dataset using the bounding box and instance mask labels. To evaluate label efficiency, we subsample the training set to create subsets with varying proportions of labeled images. We train the detector, initialized with CutLER, on each of these subsets. As a baseline, we follow the settings from MoCo-v2 [9] and train the same detection architecture initialized with a MoCo-v2 ResNet50 model, given its strong performance on object detection tasks. Both MoCo-v2 and our models are trained for the 1× schedule using Detectron2 [51], except for extremely low-shot settings with 1% or 2% labels. Following previous works [47], when training with 1% or 2% labels, we train both MoCo-v2 and our model for 3,600 iterations with a batch size of 16.

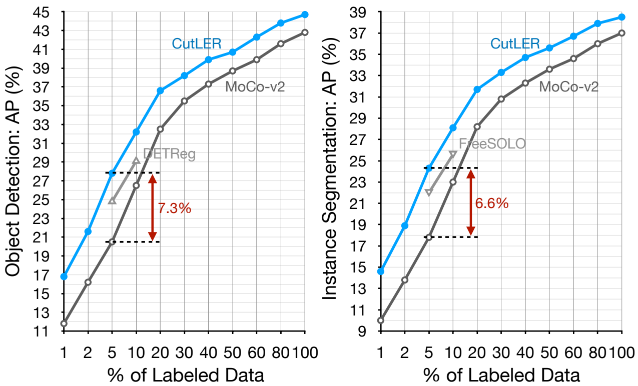

Figure 5. Finetuning CutLER for low-shot and fully supervised detection and instance segmentation. We fine-tune a Cascade Mask R-CNN model initialized with CutLER or MoCo-v2 on varying amounts of labeled data on the COCO dataset. CutLER consistently outperforms the MoCo-v2 baseline: in the low-shot setting with 1% labels and the fully supervised setting using 100% labels. CutLER also outperforms FreeSOLO [47] and DETReg [3] on this benchmark despite using an older detection architecture. Results with Mask R-CNN are in the appendix.

Methods	UVO		COCO	
	AP_{50}^{mask}	AP^{mask}	AP_{50}^{mask}	AP^{mask}
TokenCut [50]	-	-	4.9	2.0
Base	14.6	5.4	13.5	5.7
+ MaskCut	19.3	8.1	15.8	7.7
+ DropLoss	20.9	9.0	16.6	8.2
+ copy-paste [16, 19]	21.5	9.9	17.7	8.8
+ self-train (CutLER)	22.8	10.1	18.9	9.7

Table 6. Ablation study on the contribution of each component. Results reported on COCO and video segmentation dataset UVO.

Results. Fig. 5 shows the results of fine-tuning the detector on different subsets of COCO. When tested with low-shot settings, *e.g.*, 2% and 5% labeled data, our approach achieves 5.4% and 7.3% higher AP^{box} than the MoCo-v2 baseline, respectively. When training with full annotations, CutLER still consistently gives more than 2% improvements. More impressively, CutLER outperforms prior SOTA methods - FreeSOLO [47] and DETReg [3] despite using an older detection architecture.

5. Ablations

We analyze the design decisions in CutLER. We use similar settings to Sec. 4 and train CutLER only on ImageNet. We use the Cascade Mask R-CNN detection architecture and evaluate our model primarily on the COCO and UVO unsupervised detection benchmarks. All ablation studies are conducted without self-training unless otherwise noted.

Importance of each component. Table 6 analyzes the main components of CutLER and report their relative contribution. We report results on the COCO [32] dataset and

Methods	AP_{50}^{box}	AP^{box}	AR_{100}^{box}	AP_{50}^{mask}	AP^{mask}	AR_{100}^{mask}
TokenCut (1 eigenvec.)	5.2	2.6	5.0	4.9	2.0	4.4
TokenCut (3 eigenvec.)	4.7	1.7	8.1	3.6	1.2	6.9
MaskCut ($t = 3$)	6.0	2.9	8.1	4.9	2.2	6.9
CutLER	21.9	12.3	32.7	18.9	9.7	27.1

Table 7. CutLER achieves much higher results even when compared to a modified TokenCut that can produce more than one mask per image. Compared to TokenCut, MaskCut gets a higher recall without reducing precision. We report results on COCO.

a densely annotated video instance segmentation dataset UVO [45]. We also report the performance of running TokenCut [50] on the COCO dataset. Next, we use TokenCut to generate masks on ImageNet and use them for training a Cascade Mask R-CNN. This base model provides substantial gains over just using TokenCut on COCO. We add each of our proposed components to this strong base model. Using MaskCut increases AP_{50}^{mask} and AP^{mask} by 4.7% and 2.7%, respectively. Also, the improvements to AP_{50}^{mask} is larger for densely annotated dataset UVO, *i.e.* 4.7% vs. 2.7%. These results prove that MaskCut's ability to segment multiple instances per image is vital for densely annotated datasets. DropLoss brings another 1.6% and 0.9% improvements to AP_{50}^{mask} for UVO and COCO, respectively. Multi-round of self-training increases the quantity and quality of pseudo-masks, leading to 1.3% improvements.

Comparison with TokenCut. TokenCut [50] is also a zero-shot segmentation method. However, it only segments a single instance per image (Sec. 3.1). To obtain more than one segmentation mask per image, we use a modified TokenCut by using more of the smaller eigenvectors and combining all produced masks. Table 7 shows the object detection performance on COCO's validation set for vanilla TokenCut, our modified TokenCut and CutLER. Although using more eigenvectors increases the recall AR_{10}^{box}, it significantly reduces the precision AP^{box}. CutLER not only improves the average recall AR_{100}^{box} by 4× but also surpasses TokenCut's average precision AP^{box} by 4.8×.

Design choices in MaskCut and DropLoss and their impact on the final localization performance is presented in Table 8. We first study the effect of the image size used by MaskCut for generating the initial masks. As expected, Table 8a shows that MaskCut benefits from using higher resolution images presumably as it provides a higher resolution similarity between pixels. We pick a resolution of 480px for a better trade-off between the speed of MaskCut and its performance. In Table 8b, we study the effect of the threshold used in MaskCut for producing a binary W matrix (Sec. 3.2). Overall, CutLER seems to be robust to the threshold values. We understand the impact of the number of masks per image generated by MaskCut in Table 8c. Increasing the number improves the performance of the re-

Size $\rightarrow$	240	360	480	640
AP^{mask}_{50}	15.1	16.6	17.7	17.9

(a) Image size.

$\tau^{ncut} \rightarrow$	0	0.1	0.15	0.2	0.3
AP^{mask}_{50}	17.1	17.5	17.7	17.6	17.5

(b) τ^{ncut} for MaskCut.

$N \rightarrow$	2	3	4
AP^{mask}_{50}	16.9	17.7	17.7

(c) # masks per image.

$\tau^{IoU} \rightarrow$	0	0.01	0.1	0.2
AP^{mask}_{50}	17.4	17.7	14.4	12.7

(d) τ^{IoU} for DropLoss.

Table 8. Ablations for MaskCut and DropLoss used for training CutLER. We report CutLER's detection and instance segmentation performance on COCO `val2017`, without adding the self-training stage. **(a)** We vary the size of the image used for MaskCut. **(b)** We vary the threshold τ^{ncut} in MaskCut, which controls the sparsity of the affinity matrix used for Normalized Cuts. **(c)** We vary the number of masks extracted using MaskCut and train different CutLER models. **(d)** We vary τ^{IoU} in DropLoss, *i.e.*, the maximum overlap between the predicted regions and the ground truth beyond which the loss for the predicted regions is ignored. Default settings are highlighted in gray.

	UVO			COCO		
	AP^{mask}_{50}	AP^{mask}	AP^{mask}_{75}	AP^{mask}_{50}	AP^{mask}	AP^{mask}_{75}
1 round	20.6	9.0	7.0	17.7	8.8	8.0
2 rounds	22.2	9.6	7.5	18.5	9.5	8.8
3 rounds	22.8	10.1	8.0	18.9	9.7	9.2
4 rounds	22.8	10.4	8.6	18.9	9.9	9.4

Table 9. Number of self-training rounds used in CutLER. We find that 3 rounds of self-training are sufficient. Self-training provides larger gains for the densely labeled UVO dataset.

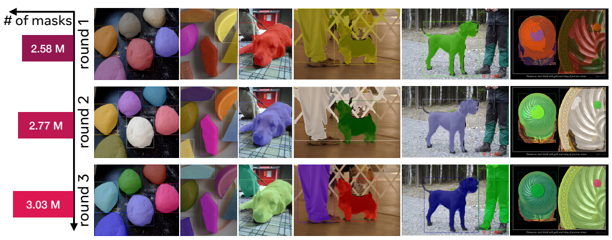

Figure 6. Multiple rounds of self-training can improve the pseudo-masks in terms of quality and quantity. We show **qualitative visualizations and number of pseudo-masks** for all three rounds.

	Mask R-CNN	Cascade Mask R-CNN	ViTDet
AP^{box}_{50} / AP^{box}	20.3 / 10.6	20.8 / 11.5	21.5 / 11.8
AP^{mask}_{50} / AP^{mask}	17.2 / 8.5	17.7 / 8.8	18.0 / 9.0

Table 10. CutLER with different detection architectures. We report results on COCO and observe that CutLER is agnostic to the detection architecture and improves performance using stronger detection architectures such as ViTDet with a backbone of ViT-B.

sulting CutLER models. This shows that MaskCut generates high-quality masks that directly impact the overall performance. Finally, in Table 8d, we vary the IOU threshold used for DropLoss. With a high threshold, we ignore the loss for a higher number of predicted regions while encouraging the model to explore. 0.01 works best for the trade-off between exploration and detection performance.

Self-training and its impact on the final performance is analyzed in Table 9. Self-training consistently improves performance across the UVO and COCO benchmarks and all metrics. UVO, which has dense object annotations, benefits more from multi-round of self-training. By default, CutLER uses 3 rounds of self-training. Fig. 6 shows qualitative examples of how self-training improves both the quality of predictions and the number of objects predicted.

Pre-train	CutLER	AP^{box}_{50}	AP^{box}_{75}	AP^{box}	AP^{mask}_{50}	AP^{mask}_{75}	AP^{mask}
IN1K	IN1K	20.8	10.8	11.5	17.7	8.0	8.8
YFCC1M	YFCC1M	19.4	10.4	10.9	16.3	7.4	8.1
IN1K	YFCC1M	14.9	7.6	8.2	12.1	5.4	5.9
YFCC1M	IN1K	14.8	7.2	8.0	11.8	5.2	5.8

Table 11. Impact of datasets used to pre-train DINO and train CutLER. CutLER's detection performance is similar when pre-training both DINO and CutLER with the same dataset: the object-centric ImageNet dataset or the non-object-centric YFCC dataset.

Generalization to different detection architectures. We use different detector architectures for training CutLER and measure their performance in Table 10. We observe that CutLER works with various architectures, and its performance is improved with stronger architectures.

Impact of the pretraining dataset. We now study the impact of the dataset used for 1) pretraining the self-supervised DINO model and 2) training the CutLER model. The commonly used ImageNet dataset has a well-known object-centric bias [12] which may affect the unsupervised detection performance. Thus, we also use YFCC [40], a non-object-centric dataset. We control for the number of images in both ImageNet and YFCC for a fair comparison and use them for training DINO and CutLER. As Table 11 shows, CutLER's performance on COCO is robust to the choice of object-centric or non-object-centric datasets as long as the same dataset is used to train DINO and CutLER. This shows the generalization of CutLER to different data distributions. However, training DINO and CutLER with different data lead to worse performance suggesting the importance of using the same image distribution for learning both models.

6. Summary

Object localization is a fundamental task in computer vision. In this paper, we have shown that a simple yet effective cut-and-learn approach can achieve extraordinary performance on challenging object detection and instance segmentation tasks without needing to train with human annotations. As a zero-shot unsupervised detector, CutLER, trained solely on ImageNet, outperforms the detection performance of previous works by over 2.7× on 11 benchmarks across various domains.

References

[1] YM Asano, C Rupprecht, and A Vedaldi. Self-labelling via simultaneous clustering and representation learning. In *International Conference on Learning Representations*, 2019. 2

[2] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. Beit: Bert pre-training of image transformers. In *International Conference on Learning Representations*, 2021. 2

[3] Amir Bar, Xin Wang, Vadim Kantorov, Colorado J Reed, Roei Herzig, Gal Chechik, Anna Rohrbach, Trevor Darrell, and Amir Globerson. Detreg: Unsupervised pretraining with region priors for object detection. In *CVPR*, 2022. 2, 5, 6, 7

[4] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6154–6162, 2018. 4, 6, 12

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 2

[6] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in Neural Information Processing Systems*, 33, 2020. 2, 4, 12

[7] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9650–9660, 2021. 1, 2, 3, 4, 6, 12

[8] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020. 1, 2

[9] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 1, 4, 6, 12, 13

[10] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758, 2021. 2

[11] Minsu Cho, Suha Kwak, Cordelia Schmid, and Jean Ponce. Unsupervised object discovery and localization in the wild: Part-based matching with bottom-up region proposals. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1201–1210, 2015. 2

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 2, 4, 8

[13] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Context as supervisory signal: Discovering objects with predictable context. In *European Conference on Computer Vision*, pages 362–377. Springer, 2014. 2

[14] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision*, pages 1422–1430, 2015. 2

[15] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2, 4

[16] Debidatta Dwibedi, Ishan Misra, and Martial Hebert. Cut, paste and learn: Surprisingly easy synthesis for instance detection. In *Proceedings of the IEEE international conference on computer vision*, pages 1301–1310, 2017. 4, 7, 12

[17] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 6, 12, 13

[18] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3354–3361. IEEE, 2012. 12, 13

[19] Golnaz Ghiasi, Yin Cui, Aravind Srinivas, Rui Qian, Tsung-Yi Lin, Ekin D Cubuk, Quoc V Le, and Barret Zoph. Simple copy-paste is a strong data augmentation method for instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2918–2928, 2021. 4, 7, 12

[20] Spyros Gidaris, Andrei Bursuc, Gilles Puy, Nikos Komodakis, Matthieu Cord, and Patrick Perez. Obow: Online bag-of-visual-words generation for self-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6830–6840, 2021.

[21] Ross Girshick. Fast r-cnn. In *ICCV*, 2015. 12

[22] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014. 12

[23] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020. 2

[24] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5356–5364, 2019. 6, 12, 13

[25] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022. 2

[26] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020. 1, 2

[27] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international*

conference on computer vision, pages 2961–2969, 2017. 4, 6, 12

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 4, 6, 12

[29] Naoto Inoue, Ryosuke Furuta, Toshihiko Yamasaki, and Kiyoharu Aizawa. Cross-domain weakly-supervised object detection through progressive domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5001–5009, 2018. 12, 13

[30] Philipp Krähenbühl and Vladlen Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. *NeurIPS*, 2011. 4

[31] 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. *International Journal of Computer Vision*, 128(7):1956–1981, 2020. 12, 13

[32] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 1, 5, 6, 7, 12, 13

[33] S Maji, NK Vishnoi, and J Malik. Biased normalized cuts. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2057–2064, 2011. 3

[34] Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6707–6717, 2020. 2

[35] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[36] 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 *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8430–8439, 2019. 12, 13

[37] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000. 3, 4

[38] 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. *arXiv preprint arXiv:2109.14279*, 2021. 2, 3, 5, 6, 12

[39] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[40] Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data in multimedia research. *Communications of the ACM*, 59(2):64–73, 2016. 8

[41] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013. 12

[42] Wouter Van Gansbeke, Simon Vandenhende, and Luc Van Gool. Discovering object masks with transformers for unsupervised semantic segmentation. *arXiv preprint arXiv:2206.06363*, 2022. 2, 3, 6

[43] Huy V Vo, Patrick Pérez, and Jean Ponce. Toward unsupervised, multi-object discovery in large-scale image collections. In *European Conference on Computer Vision*, pages 779–795. Springer, 2020. 2, 3, 6, 12

[44] Van Huy Vo, Elena Sizikova, Cordelia Schmid, Patrick Pérez, and Jean Ponce. Large-scale unsupervised object discovery. *Advances in Neural Information Processing Systems*, 34:16764–16778, 2021. 2, 6

[45] Weiyao Wang, Matt Feiszli, Heng Wang, and Du Tran. Unidentified video objects: A benchmark for dense, open-world segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10776–10785, 2021. 6, 7, 12, 13

[46] Xudong Wang, Ziwei Liu, and Stella X Yu. Unsupervised feature learning by cross-level instance-group discrimination. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12586–12595, 2021. 2, 4, 12

[47] Xinlong Wang, Zhiding Yu, Shalini De Mello, Jan Kautz, Anima Anandkumar, Chunhua Shen, and Jose M Alvarez. Freesolo: Learning to segment objects without annotations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14176–14186, 2022. 1, 2, 3, 5, 6, 7, 13

[48] Xinlong Wang, Rufeng Zhang, Tao Kong, Lei Li, and Chunhua Shen. Solov2: Dynamic and fast instance segmentation. *Advances in Neural information processing systems*, 33:17721–17732, 2020. 2, 5, 6

[49] Xinlong Wang, Rufeng Zhang, Chunhua Shen, Tao Kong, and Lei Li. Dense contrastive learning for self-supervised visual pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3024–3033, 2021.

[50] 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. *arXiv preprint arXiv:2209.00383*, 2022. 2, 3, 4, 6, 7, 12

[51] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019. 6, 13

[52] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3733–3742, 2018. 2

[53] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487. PMLR, 2016. 2

[54] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers

for end-to-end object detection. In *International Conference on Learning Representations*, 2020. 2, 6

[55] Chengxu Zhuang, Alex Lin Zhai, Daniel Yamins, , et al. Local aggregation for unsupervised learning of visual embeddings. In *ICCV*, 2019. 2

[56] Adrian Ziegler and Yuki M Asano. Self-supervised learning of object parts for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14502–14511, 2022. 2