Object-aware Contrastive Learning for Debiased Scene Representation

Anonymous Author(s)
Affiliation
Address
email

Abstract

Contrastive self-supervised learning has shown impressive results in learning visual representations from unlabeled images by enforcing invariance against different data augmentations. However, the learned representations are often biased to the spurious scene correlations of different objects (i.e., contextual bias) or object and background (i.e., background bias), which may harm their generalization on the downstream tasks. To tackle the issue, we develop a novel object-aware contrastive learning framework that first (a) localizes objects in a self-supervised manner and then (b) debias scene correlations via appropriate data augmentations considering the inferred object locations. For (a), we propose the contrastive class activation map (ContraCAM), which finds the most discriminative regions (e.g., objects) in the image compared to the other images using the contrastively trained models. We further improve the ContraCAM to detect multiple objects and entire shapes via an iterative refinement procedure. For (b), we introduce two data augmentations based on ContraCAM, object-aware random crop and background mixup, which reduce contextual and background biases during contrastive self-supervised learning, respectively. Our experiments demonstrate the effectiveness of our representation learning framework, particularly when trained under multi-object images or evaluated under the background (and distribution) shifted images.

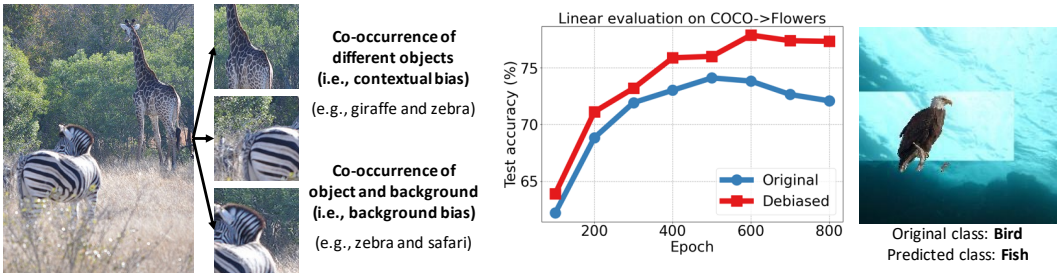
1 Introduction

Self-supervised learning of visual representations from unlabeled images is a fundamental task of machine learning, which establishes various applications including object recognition [1, 2], reinforcement learning [3, 4], out-of-distribution detection [5, 6], and multimodal learning [7, 8]. Recently, contrastive learning [1, 2, 9–15] has shown remarkable advances along this line. The idea is to learn invariant representations by attracting the different views (e.g., augmentations) of the same instance (i.e., positives) while contrasting different instances (i.e., negatives).[1]

Despite the success of contrastive learning on various downstream tasks [16], they still suffer from the generalization issue due to the unique features of the training datasets [17–19] or the choice of data augmentations [19–21]. In particular, the co-occurrence of different objects and background in randomly cropped patches (i.e., positives) leads the model to suffer from the *scene bias*. For example, Figure 1a presents two types of the scene bias: the positive pairs contain different objects (e.g., giraffe and zebra), and the patches contain adjacent object and background (e.g., zebra and safari). Specifically, the co-occurrence of different objects is called contextual bias [22], and that of object and background is called background bias [23]. Attracting the patches in contrastive learning

[1] Some recent works (e.g., [14, 15]) attract the positives without contrasting the negatives. While we mainly focus on contrastive learning with negatives, our method is also applicable to the positive-only methods.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.



(a) Scene bias in random crop (b) Performance drop (c) Biased prediction

Figure 1: Scene bias issue (a) and its negative effects on contrastive learning. (b) Linear evaluation [24] of the original MoCov2 [1] and our debiased method, trained and evaluated on the COCO [25] and Flowers [26] datasets, respectively, using the ResNet-50 architecture [27]. (c) Prediction of the original MoCov2 and our debiased method on an image from the Background Challenge [23].

makes the features of correlated objects and background indistinguishable, which may harm their generalization (Figure 1b) because of being prone to biases (Figure 1c).

Contribution. We develop a novel object-aware contrastive learning framework that mitigates the scene bias and improves the generalization of learned representation. The key to success is the proposed *contrastive class activation map* (ContraCAM), a simple yet effective self-supervised object localization method by contrasting other images to find the most discriminate regions in the image. We leverage the ContraCAM to create new types of positives and negatives. First, we introduce two data augmentations for constructing the *positive* sample-pairs of contrastive learning: *object-aware random crop* and *background mixup* that reduce contextual and background biases, respectively. Second, by equipping ContraCAM with an iterative refinement procedure, we extend it to detect multiple objects and entire shapes, which allows us to generate masked images as effective *negatives*.

We demonstrate that the proposed method can improve two representative contrastive (or positive-only) representation learning schemes, MoCov2 [28] and BYOL [14], by reducing contextual and background biases as well as learning object-centric representation. In particular, we improve:

- The representation learning under multi-object images, evaluated on the COCO [25] dataset, boosting the performance on the downstream tasks, e.g., classification and detection.
- The generalizability of the learned representation on the background shifts, i.e., objects appear in the unusual background (e.g., fish on the ground), evaluated on the Background Challenge [23].
- The generalizability of the learned representation on the distribution shifts, particularly for the shape-biased, e.g., ImageNet-Sketch [29], and corrupted, e.g., ImageNet-C [30] datasets.

Furthermore, ContraCAM shows comparable results with the state-of-the-art unsupervised localization method (and also with the supervised classifier CAM) while being simple.

2 Object-aware Contrastive Learning

We first briefly review contrastive learning in Section 2.1. We then introduce our object localization and debiased contrastive learning methods in Section 2.2 and Section 2.3, respectively.

2.1 Contrastive learning

Contrastive self-supervised learning aims to learn an encoder $f(\cdot)$ that extracts a useful representation from an unlabeled image x by attracting similar sample x^+ (i.e., positives) and dispelling dissimilar samples $\{x_i^-\}$ (i.e., negatives). In particular, instance discrimination [10] defines the same samples of different data augmentations (e.g., random crop) as the positives and different samples as negatives. Formally, contrastive learning maximizes the contrastive score:

$$s_{\texttt{con}}(x; x^+, \{x_n^-\}) := \log \frac{\exp(\mathrm{sim}(z(x), \bar{z}(x^+))/\tau)}{\exp(\mathrm{sim}(z(x), \bar{z}(x^+))/\tau) + \sum_{x_n^-} \exp(\mathrm{sim}(z(x), \bar{z}(x_n^-))/\tau)}, \quad (1)$$

where $z(\cdot)$ and $\bar{z}(\cdot)$ are the output and target functions wrapping the representation $f(x)$ for use, $\mathrm{sim}(\cdot,\cdot)$ denotes the cosine similarity, and τ is a temperature hyperparameter. The specific form of $z(\cdot), \bar{z}(\cdot)$ depends on the method. For example, MoCov2 [28] sets $z(\cdot) = g(f(\cdot)), \bar{z}(\cdot) = g_m(f_m(\cdot))$ where $g(\cdot)$ is a projector network to indirectly match the feature $f(x)$ and $f_m(\cdot), g_m(\cdot)$ are the momentum version of the encoder and projectors. On the other hand, BYOL [14] sets $z(\cdot) = h(g(f(\cdot))), \bar{z}(\cdot) = g_m(f_m(\cdot))$, where $h(\cdot)$ is an additional predictor network to avoid collapse of the features because it only maximizes the similarity score $s_{\texttt{sim}}(x; x^+) := \mathrm{sim}(z(x), \bar{z}(x^+))$ [14, 15].

Scene bias in contrastive learning. Despite the success of contrastive learning, they often suffer from the *scene bias*: entangling representations of co-occurring (but different) objects, i.e., contextual bias [22], or adjacent object and background, i.e., background bias [23], by attracting the randomly cropped patches reflecting the correlations (Figure 1a). The scene bias harms the performance (Figure 1b) and generalization of the learned representations on distribution shifts (Figure 1c). To tackle the issue, we propose object-aware data augmentations for debiased contrastive learning (Section 2.3) utilizing the object locations inferred from the contrastively trained models (Section 2.2).

2.2 ContraCAM: Unsupervised object localization via contrastive learning

We aim to find the most discriminative region in an image, such as objects for scene images, compared to the other images. To this end, we extend the (gradient-based) class activation map (CAM) [31, 32], originally used to find the salient regions for the prediction of classifiers. Our proposed method, *contrastive class activation map* (ContraCAM), has two differences from the classifier CAM. First, we use the contrastive score instead of the softmax probability. Second, we discard the negative signals from the similar objects in the negative batch since they cancel out the positive signals and hinder the localization, which is crucial as shown in Table 1 and Appendix C.1).

Following the classifier CAM, we define the saliency map as the weighted sum of spatial activations (e.g., penultimate feature before pooling), where the weight of each activation is given by the importance, the sum of gradients, of the activation for the score function. Formally, let $\mathbf{A} := [A^k_{ij}]$ be a spatial activation of an image x where $1 \le i \le H, 1 \le j \le W, 1 \le k \le K$ denote the index of row, column, and channel, and H, W, K denote the height, width, and channel size of the activation. Given a batch of samples $\mathcal{B}$, we define the score function of the sample x as the contrastive score $s_{\texttt{con}}$ in Eq. (1) using the sample x itself as a positive[2] and the remaining samples $\mathcal{B} \setminus x$ as negatives. Then, the weight of the k-th activation α_k and the CAM mask $\texttt{CAM} := [\texttt{CAM}_{ij}] \in [0,1]^{H \times W}$ are:

$$\texttt{CAM}_{ij} = \texttt{Normalize}\left(\texttt{ReLU}\left(\sum_k \alpha_k A^k_{ij}\right)\right), \ \alpha_k = \texttt{ReLU}\left(\frac{1}{HW}\sum_{i,j} \frac{\partial s_{\texttt{con}}(x; x, \mathcal{B} \setminus x)}{\partial A^k_{i,j}}\right), \quad (2)$$

where $\texttt{Normalize}(x) := \frac{x - \min x}{\max x - \min x}$ is a normalization function that maps the elements to $[0,1]$. We highlight the differences from the classifier CAM with the red color. Note that the ReLU used to compute α_k in Eq. (2) discards the negative signals. The negative signal removal trick also slightly improves the classifier CAM [33] but much effective for the ContraCAM.

We further improve the ContraCAM to detect multiple objects and entire shapes with an iterative refinement procedure [34]: cover the salient regions of the image with the (reverse of) current CAM, predict new CAM from the masked image, and aggregate them (see Figure 2). It expands the CAM regions since the new CAM from the masked image detects the unmasked regions. Here, we additionally provide the masked images in the batch (parellely computed) as the negatives: they are better negatives by removing the possibly existing similar objects. Also, we use the original image x as the positive to highlight the undetected objects. Formally, let $\texttt{CAM}^t$ be the CAM of iteration t and $\overline{\texttt{CAM}}^t := [\overline{\texttt{CAM}}^t_{ij}] = [\max_{l \le t} \texttt{CAM}^l_{ij}]$ be the aggregated CAM mask. Also, let x^t be the image softly masked by the (reverse of) current aggregated mask, i.e., $x^t := (1 - \overline{\texttt{CAM}}^{t-1}) \odot x$ for $t \ge 2$ and $x^1 = x$ where $\odot$ denotes an element-wise product, and $\mathcal{B}^t := \{x^t_n\}$ be the batch of the masked images. Then, we define the score function for iteration t as:

$$s^t_{\texttt{con}}(x) := s_{\texttt{con}}(x^t; x, \cup_{l \le t}(\mathcal{B}^l \setminus x^l)). \quad (3)$$

We substitute the contrastive score $s_{\texttt{con}}$ in Eq. (2) with the $s^t_{\texttt{con}}$ in Eq. (3) to compute the CAM of iteration t, and use the final aggregated mask after T iterations. We remark that the CAM results are

[2] It does not affect the score but is defined for the notation consistency with the iterative extension.

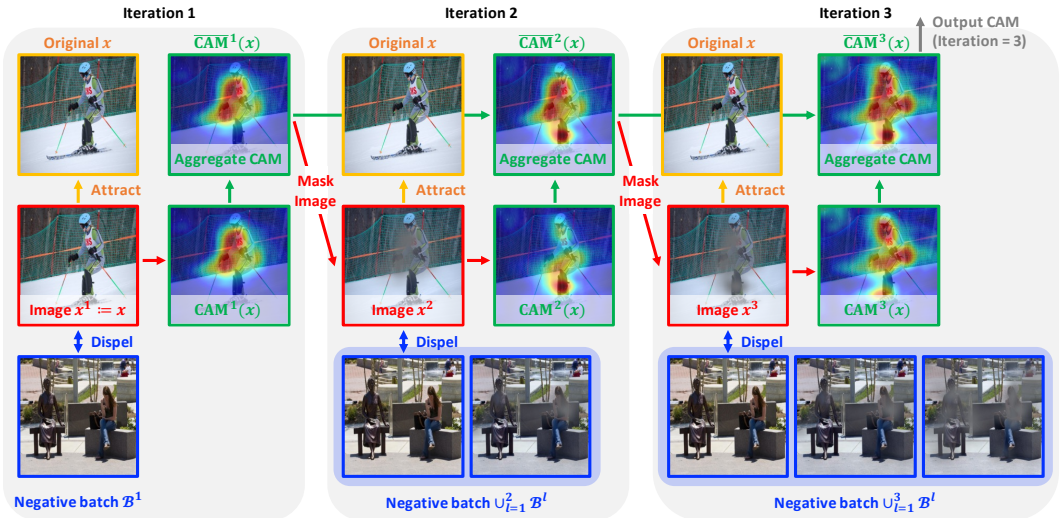


Figure 2: Visual illustration of the Iterative ContraCAM ($T = 3$) procedure.

not sensitive to the number of iterations T if it is large enough; CAM converges to the stationary value since soft masking x^t regularizes the CAM not to be keep expanded (see Appendix C.2). We provide the pseudo-code of the entire Iterative ContraCAM procedure in Appendix A.

Note that contrastive learning was known to be ineffective at localizing objects [35] with standard saliency methods (using a classifier on top of the learned representation) since attracting the randomly cropped patches makes the model look at the entire scene. To our best knowledge, we are the first to extend the CAM for the self-supervised setting, relaxing the assumption of class labels. Selvaraju et al. [36] considered CAM for contrastive learning, but their purpose was to regularize CAM to be similar to the ground-truth masks (or predicted by pre-trained models) and used the similarity of the image and the masked image (by ground-truth masks) as the score function of CAM.

2.3 Object-aware augmentations for debiased contrastive learning

We propose two data augmentations for contrastive learning that reduce contextual and background biases, respectively, utilizing the object locations inferred by ContraCAM. Both augmentations are applied to the *positive* samples before other augmentations; thus, it is applicable for both contrastive learning (e.g., MoCov2 [28]) and positive-only methods (e.g., BYOL [14]).

Reducing contextual bias. We first tackle the contextual bias of contrastive learning, i.e., entangling the features of different objects. To tackle the issue, we propose a data augmentation named *object-aware random crop*, which restricts the random crop around a single object and avoids the attraction of different objects. To this end, we first extract the (possibly multiple or none) bounding boxes of the image from the binarized mask[3] of the ContraCAM. We then crop the image around the box, randomly chosen from the boxes, before applying other augmentations (e.g., random crop). Here, we apply augmentations (to produce positives) to the same cropped box; thus, the patches are restricted in the same box. Technically, it only requires a few line addition of code:

```
if len(boxes) > 0: # can be empty
    box = random.choice(boxes)
    image = image.crop(box)
# apply other augmentations (e.g., random crop)
```

Purushwalkam and Gupta [19] considered a similar approach using ground-truth bounding boxes applied on MoCov2. However, we found that cropping around the ground-truth boxes often harms contrastive learning (see Table 3). This is because some objects (e.g., small ones) in ground-truth boxes are hard to discriminate (as negatives), making contrastive learning hard to optimize. In contrast, the ContraCAM produces more discriminative boxes, often outperforming the ground-truth boxes

[3]Threshold the mask or apply a post-processing method, e.g., conditional random field (CRF) [37].

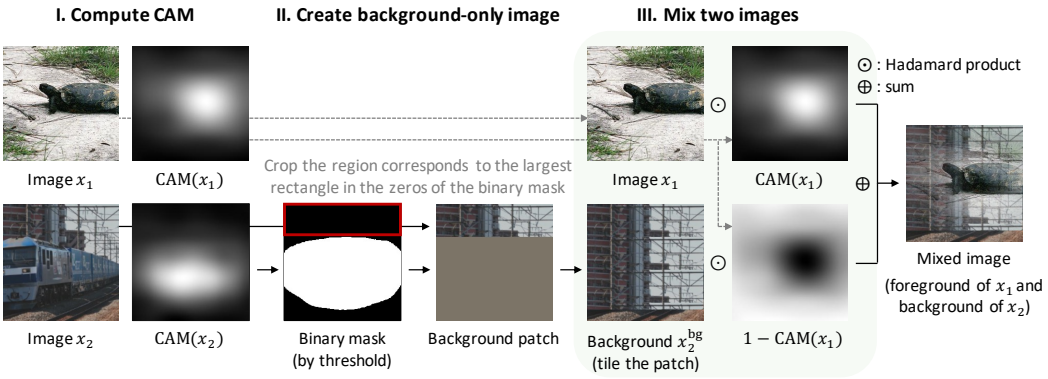


Figure 3: Visual illustration of the background mixup procedure.

(see Appendix D.1). Note that the positive-only methods do not suffer from the issue: both ground-truth and ContraCAM boxes work well. On the other hand, Selvaraju et al. [36] used a pre-trained segmentation model to constrain the patches to contain objects. It partly resolves the false positive issue by avoiding the attraction of background-only patches but does not prevent the patches with different objects; in contrast, the object-aware random crop avoids both cases.

Reducing background bias. We then tackle the background bias of contrastive learning, i.e., entangling the features of adjacent object and background. To this end, we propose a data augmentation named *background mixup*, which substitutes the background of an image with other backgrounds. Intuitively, the positive samples share the objects but have different backgrounds, thus reducing the background bias. Formally, background mixup blends an image x_1 and a background-only image x_2^{bg} (generated from an image x_2) using the ContraCAM of image x_1 as a weight, i.e.,

$$x_1^{\texttt{bg-mix}} := \texttt{CAM}(x_1) \odot x_1 + (1 - \texttt{CAM}(x_1)) \odot x_2^{\text{bg}}, \tag{4}$$

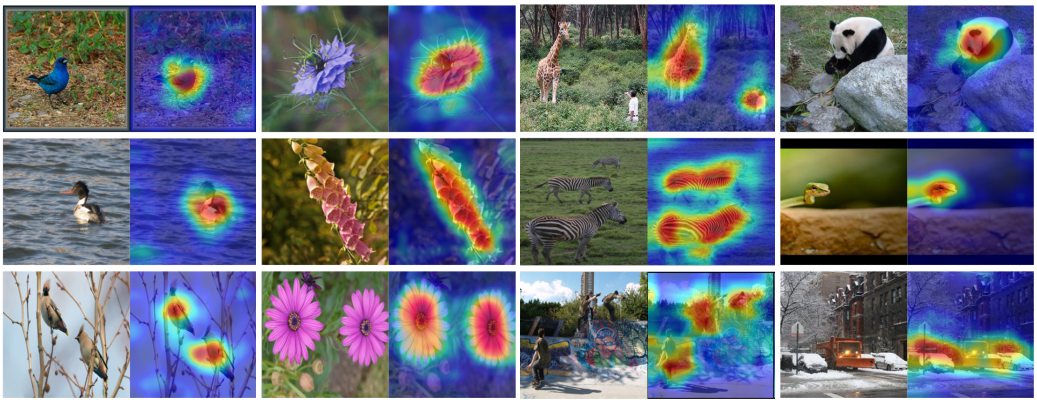
where $\odot$ denotes an element-wise product. Here, the background-only image x_2^{bg} is generated by tiling the background patch of the image x_2 inferred by the ContraCAM. Precisely, we choose the largest rectangle in the zeros of the binarized CAM mask for the region of the background patch. The overall procedure of the background mixup is illustrated in Figure 3.

Prior works considered the background bias for contrastive learning [35, 38] but used a pre-trained segmentation model and copy-and-pasted the objects to the background-only images using binary masks. We also tested the copy-and-paste version with the binarized CAM, but the soft version in Eq. (4) performed better (see Appendix E.1); one should consider the confidence of the soft masks since they are inaccurate. Furthermore, the background mixup improves the generalization on distribution shifts, e.g., shape-biased [29, 39, 40] and corrupted [30] datasets (see Table 6). Remark that the background mixup often outperforms the Mixup [41] and CutMix [42] applied for contrastive learning [43]. Intuitively, the background mixup can be viewed as a saliency-guided extension [44, 45] of mixup but not mixing the targets (positives), since the mixed patch should be only considered as the positive of the patch sharing foreground, not the one sharing background.

3 Experiments

We first verify the localization performance of ContraCAM in Section 3.1. We then demonstrate the efficacy of our debiased contrastive learning: object-aware random crop improves the training under multi-object images by reducing contextual bias in Section 3.2, and background mixup improves generalization on background and distribution shifts by reducing background bias in Section 3.3.

Common setup. We apply our method on two representative contrastive (or positive-only) learning models: MoCov2 [28] and BYOL [14], under the ResNet-18 and ResNet-50 architectures [27]. We train the models for 800 epochs on COCO [25] and ImageNet-9 [23], and 2,000 epochs on CUB [46] and Flowers [26] datasets with batch size 256. We first train the vanilla MoCov2 and BYOL on the training dataset and pre-compute the CAM masks (and the corresponding bounding boxes and background-only images) using the models. Then, we train our method from scratch using



(a) CUB [46] (b) Flowers [26] (c) COCO [25] (d) ImageNet-9 [23]

Figure 4: Visualization of the ContraCAM results on various image datasets.

Table 1: Mask mIoU of unsupervised object localization methods. Bold denote the best results.

Method	CUB	Flowers	COCO	ImageNet-9
ReDO [50]	0.426	0.764	0.286	0.416
ContraCAM w/o negative signal removal	0.287	0.555	0.242	0.361
ContraCAM (ours)	**0.460**	**0.776**	**0.319**	**0.427**

Table 2: MaxBoxAccV2 of the classifier CAM and ContraCAM using the ImageNet-trained classifier and MoCov2. We report the localization results on the trained (ImageNet) and unseen datasets.

Model	Method	ImageNet	CUB	Flowers	VOC	OpenImages
Supervised	Classifier CAM	55.95	55.52	76.87	53.88	48.01
MoCov2	ContraCAM (ours)	55.88	64.07	75.64	59.40	49.89

the fixed pre-computed masks (and others). We also retrained the model using the CAM masks inferred from our debiased model but did not see the gain (see Appendix D.3). We follow the default hyperparameters of MoCov2 and BYOL, except using the smaller minimum random crop scale of 0.08 (instead of the original 0.2) since it performed better, especially for the training on multi-object images. We run a single trial for contextual bias and three trials for background bias experiments.

We use the penultimate spatial activations to compute the CAM results. At inference, we follow the protocol of [47] that doubly expands the resolution of the activations to detect the smaller objects through decreasing the stride of the convolutional layer in the final residual block. Since it produces the smaller masks, we use a larger number of iterations for the (Iterative) ContraCAM; specifically, we update ten iterations for object localization and contextual bias results. Here, we apply the conditional random field (CRF) using the default hyperparameters from the pydensecrf library [48] to produce segmentation masks and use the opencv [49] library to extract bounding boxes. We use a single iteration of the ContraCAM without the expansion trick for background bias results; it is sufficient for single instance images. Here, we binarize the masks with a threshold of 0.2 to produce background-only images. We provide the further implementation details in Appendix B.

Computation time. The training of the baseline models on the COCO ($\sim$100,000 samples) dataset takes $\sim$1.5 days on 4 GPUs and $\sim$3 days on 8 GPUs for ResNet-18 and ResNet-50 architectures, respectively, using a single machine with 8 GeForce RTX 2080 Ti GPUs; proportional to the number of samples and training epochs for other cases. The inference of ContraCAM takes a few minutes for the entire training dataset and generating the boxes and background-only images take a few hours. Using the pre-computed information, our method only increases a negligible amount of training time.

3.1 Unsupervised object localization

We check the performance of our proposed self-supervised object localization method, ContraCAM. Figure 4 shows the examples of the ContraCAM on various image datasets, including CUB, Flowers, COCO, and ImageNet-9 datasets. ContraCAM identifies the shapes of the objects and separates

Table 3: Linear evaluation (%) of MoCov2 and BYOL on various image classification tasks, trained with the original image (-) or object-aware random crop (OA-Crop) using the ContraCAM (CAM) or ground-truth (GT) bounding boxes from the COCO dataset. Gray lines denote the usage of GT boxes, blue and red brackets denote the gain and loss of OA-Crop compared to the original image.

Model	Network	OA-Crop	Test dataset						
			COCO-Crop	CIFAR10	CIFAR100	CUB	Flowers	Food	Pets
MoCov2	ResNet-50	-	74.30	77.58	53.26	22.90	72.09	59.70	59.25
MoCov2	ResNet-50	CAM	76.37 (+2.07)	84.10 (+6.52)	62.72 (+9.46)	25.46 (+2.56)	77.33 (+5.24)	62.01 (+2.31)	60.97 (+1.72)
MoCov2	ResNet-50	GT	76.44 (+2.14)	84.03 (+6.45)	62.81 (+9.55)	22.59 (-0.31)	75.09 (+3.00)	57.47 (-2.23)	57.67 (-1.58)
BYOL	ResNet-50	-	73.36	76.62	51.79	21.95	73.77	59.49	60.72
BYOL	ResNet-50	CAM	74.92 (+1.56)	82.79 (+6.17)	61.13 (+9.34)	24.34 (+2.39)	77.83 (+4.06)	61.83 (+2.34)	61.27 (+0.55)
BYOL	ResNet-50	GT	80.69 (+7.33)	85.92 (+9.30)	65.06 (+13.27)	28.68 (+6.73)	77.95 (+4.18)	64.63 (+5.14)	65.69 (+4.97)
MoCov2	ResNet-18	-	67.38	66.83	41.85	15.36	58.81	45.88	45.37
MoCov2	ResNet-18	CAM	69.92 (+2.54)	76.73 (+9.90)	53.25 (+11.40)	16.26 (+0.90)	64.77 (+5.96)	48.56 (+2.68)	47.37 (+2.00)
MoCov2	ResNet-18	GT	71.60 (+4.22)	77.99 (+11.16)	53.32 (+11.47)	18.19 (+2.83)	65.43 (+6.62)	46.41 (+0.53)	48.68 (+3.31)
BYOL	ResNet-18	-	67.74	67.82	41.96	17.24	64.79	49.58	52.90
BYOL	ResNet-18	CAM	70.85 (+3.11)	77.37 (+9.55)	54.79 (+12.83)	18.24 (+1.00)	70.56 (+5.77)	53.16 (+3.58)	54.27 (+1.37)
BYOL	ResNet-18	GT	76.59 (+8.85)	81.23 (+13.41)	58.11 (+16.15)	22.99 (+5.75)	73.25 (+8.46)	55.33 (+5.75)	59.80 (+6.90)

Table 4: Mean AP (%) of MoCov2 and BYOL fine-tuned on the COCO detection and segmentation tasks, following the setting of the table above, using the ResNet-50 architecture.

	MoCov2			BYOL		
Dataset	Baseline	OA-Crop (CAM)	OA-Crop (GT)	Baseline	OA-Crop (CAM)	OA-Crop (GT)
COCO Detection	36.34	36.60 (+0.26)	35.73 (-0.61)	35.11	35.63 (+0.52)	35.05 (-0.06)
COCO Segmentation	31.95	32.37 (+0.42)	31.47 (-0.48)	31.10	31.39 (+0.29)	31.12 (+0.02)

the different objects. We also provide the quantitative results in Table 1, reporting the mask mean intersection-over-unions (mIoUs). The ContraCAM is comparable with the state-of-the-art unsupervised object localization method, ReDO [50].[4] Also, one can see that the negative signal removal, i.e., `ReLU` in Eq. (1), is a critical to the performance (see Appendix C.1 for the visual examples).

We also compare the localization performance of the ContraCAM and the classifier CAM in Table 2, trained on the ImageNet [51] dataset under the ResNet-50 architecture, and evaluated on the trained (ImageNet) and unseen datasets. We use the publicly available supervised classifier [52] and MoCov2, and follow the MaxBoxAccV2 evaluation protocol [47] that averages the ratios of the bounding boxes whose mean intersection-over-unions (mIoUs) are larger than 30%, 50%, and 70%. The ContraCAM often outperforms the classifier CAM, especially for the unseen images (e.g., CUB). This is because the classifiers project the features unrelated to the target classes, losing their generalizability on the out-of-class samples. See Appendix C.4 for the detailed discussion.

We provide the additional analysis and results in Appendix C. It includes: effects of the number of iterations (Appendix C.2) and the choice of negative batches (Appendix C.3) for the ContraCAM, and comparison with the gradient-based saliency methods [53, 54] when used the same contrastive score (Appendix C.5). ContraCAM detects the background if we construct the batch with similar images, but providing a small amount of randomly selected images resolve the issue.

3.2 Reducing contextual bias: Representation learning from multi-object images

We demonstrate the effectiveness of the object-aware random crop (OA-Crop) for representation learning under multi-object images by reducing contextual bias. To this end, we train MoCov2 and BYOL on the COCO dataset, comparing them with the models that applied the OA-Crop using the ground-truth (GT) bounding boxes or inferred ones from the ContraCAM.

We first compare the linear evaluation [24], test accuracy of a linear classifier trained on top of the learned representation, in Table 3. We report the results on the COCO-Crop, i.e., the objects in the COCO dataset cropped by the GT boxes, CIFAR-10 and CIFAR-100 [55], CUB, Flowers, Food [56], and Pets [57] datasets. OA-Crop significantly improves the linear evaluation of MoCov2 and BYOL for all tested cases. Somewhat interestingly, OA-Crop using the ContraCAM boxes even

[4] We only consider fully self-supervised methods solely utilizing the data of the target domain.

Table 5: Test accuracy (%) of a linear classifier evaluated on the Background Challenge [23], both backbone and classifiers are trained under the ORIGINAL dataset. The backbone is trained from the original image (Baseline), background mixup using ContraCAM (BG-Mixup (CAM)), or hard background mixing using ground-truth masks (BG-HardMix (GT)), under the ResNet-18 architecture. Blue (or red) arrows imply higher (or lower) is better. Subscripts denote standard deviation.

	MoCov2			BYOL		
Dataset	Baseline	BG-Mixup (CAM)	BG-HardMix (GT)	Baseline	BG-Mixup (CAM)	BG-HardMix (GT)
Original $\uparrow$	89.17±0.49	90.73±0.05 (+1.56)	89.69±0.14 (+0.52)	87.30±0.61	89.30±0.02 (+2.00)	90.95±0.33 (+3.65)
Only-BG-B $\downarrow$	31.29±2.46	29.60±0.89 (-1.69)	26.44±1.63 (-4.85)	25.59±0.78	25.70±3.46 (+0.05)	27.28±0.04 (+1.69)
Only-BG-T $\downarrow$	44.91±0.16	41.95±0.38 (-2.96)	40.11±0.58 (-4.80)	42.83±0.51	39.94±0.52 (-2.89)	41.16±0.17 (-1.67)
Only-FG $\uparrow$	63.62±4.71	70.55±1.71 (+6.93)	72.68±0.69 (+9.06)	61.04±0.94	67.53±0.30 (+6.49)	72.63±1.13 (+11.59)
Mixed-Same $\uparrow$	80.98±0.34	84.13±0.33 (+3.15)	84.48±0.17 (+3.50)	79.30±0.31	81.28±0.53 (+1.98)	84.94±0.47 (+5.64)
Mixed-Rand $\uparrow$	60.34±0.66	66.89±0.54 (+6.55)	71.95±0.54 (+11.61)	58.03±0.85	63.83±0.53 (+5.80)	70.51±0.33 (+12.48)
Mixed-Next $\uparrow$	55.50±0.71	63.64±0.41 (+8.14)	70.25±0.14 (+14.75)	53.35±0.36	63.05±3.54 (+9.70)	66.81±0.08 (+13.46)
BG-Gap $\downarrow$	20.64±0.36	17.24±0.31 (-3.40)	12.53±0.69 (-8.11)	21.27±0.64	17.45±0.15 (-3.82)	14.44±0.56 (-6.83)

Table 6: Test accuracy (%) of a linear classifier evaluated on various distribution-shifted datasets, following the training of the table above, additionally comparing with Mixup [41] and CutMix [42].

Model	Augmentation	Test dataset			
		ImageNet-Sketch-9	Stylized-ImageNet-9	ImageNet-R-9	ImageNet-C-9
MoCov2	Original	46.70±0.67	25.66±0.54	37.51±0.80	31.82±0.40
MoCov2	+Mixup [41]	51.18±0.88 (+4.48)	32.36±0.12 (+6.70)	41.00±0.12 (+3.49)	40.15±2.07 (+8.33)
MoCov2	+CutMix [42]	45.92±0.88 (-0.78)	26.46±0.68 (+0.80)	37.07±0.31 (-0.44)	32.29±0.60 (+0.47)
MoCov2	+BG-Mixup (ours)	**52.15**±0.93 (+5.45)	**33.36**±0.61 (+7.70)	**41.50**±0.45 (+3.99)	**44.39**±0.89 (+12.57)
BYOL	Original	45.15±1.12	23.80±0.45	36.21±0.31	28.62±0.06
BYOL	+Mixup [41]	50.12±1.61 (+1.97)	**28.11**±1.15 (+4.31)	37.90±0.44 (+1.69)	32.48±0.55 (+3.86)
BYOL	+CutMix [42]	46.07±0.05 (+1.39)	23.98±0.05 (+0.18)	35.43±0.44 (-0.78)	28.74±1.52 (+0.12)
BYOL	+BG-Mixup (ours)	**52.40**±0.70 (+7.25)	27.01±0.74 (+3.21)	**39.62**±0.21 (+3.41)	**33.83**±0.28 (+5.21)

outperforms the GT boxes for MoCov2 under the ResNet-50 architecture. This is because the GT boxes often contain objects hard to discriminate (e.g., small objects), making contrastive learning hard to optimize; in contrast, ContraCAM finds more distinct objects. Note that BYOL does not suffer from this issue and performs well with both boxes. See Appendix D.1 for the detailed discussion.

We also compare the detection (and segmentation) performance measured by mean average precision (AP), an area under the precision-recall curve of the bounding boxes (or segmentation masks), on the COCO detection and segmentation tasks in Table 4. Here, we fine-tune the MoCov2 and BYOL models using the ResNet-50 architecture. Remark that OA-Crop using the ContraCAM boxes outperforms the baselines, while the GT boxes are on par or worse. This is because the GT boxes solely focus on the objects while ContraCAM also catches the salient scene information.

3.3 Reducing background bias: Generalization on background and distribution shifts

We demonstrate the effectiveness of the background mixup (BG-Mixup) for the generalization of the learned representations on background and distribution shifts by reducing background bias and learning object-centric representation. To this end, we train MoCov2 and BYOL (and BG-Mixup upon them) on the ORIGINAL dataset from the Background Challenge [23], a 9 superclass (370 classes) subset of the ImageNet [51]. We then train a linear classifier on top of the learned representation using the ORIGINAL dataset. Here, we evaluate the classifier on the Background Challenge datasets for the background shift results, and the corresponding 9 superclasses of the ImageNet-Sketch [29], Stylized-ImageNet [39], ImageNet-R [40], and ImageNet-C [30] datasets, denoted by putting '-9' at the suffix of the dataset names, for the distribution shift results (see Appendix B.3 for details).

We additionally compare BG-Mixp with the hard background mixing (i.e., copy-and-paste) using ground-truth masks (BG-HardMix (GT)) for the background shift experiments, and Mixup [41] and CutMix [42] (following the training procedure of [43]) for the distribution shift experiments. We also tested the BG-HardMix using the binarized CAM but did not work well (see Appendix E.1). On the other hand, the BG-Mixup often makes contrastive learning hard to be optimized by producing hard positives; thus, we apply BG-Mix with probability $p_{\texttt{mix}} < 1$, independently applied on the patches. We tested $p_{\texttt{mix}} \in \{0.2, 0.3, 0.4, 0.5\}$ and choose $p_{\texttt{mix}} = 0.4$ for MoCov2 and $p_{\texttt{mix}} = 0.3$ for BYOL.

Note that MoCov2 permits the higher $p_{\texttt{mix}}$, since finding the closest sample from the (finite) batch is easier than clustering infinitely many samples (see Appendix E.2 for details).

Table 5 presents the results on background shifts: BG-Mixup improves the predictions on the object-focused datasets (e.g., MIXED-RAND) while regularizing the background-focused datasets (e.g., ONLY-BG-T). Table 6 presents the results on distribution shifts: BG-Mixup mostly outperforms the Mixup and the CutMix. We also provide the BG-HardMix (GT) results on distribution shifts in Appendix E.3 and the mixup results on background shifts in Appendix E.4. The superiority of BG-Mix on both background and distribution shifts shows that its merits come from both object-centric learning via reducing background and the saliency-guided input interpolation.

4 Related work

Contrastive learning. Contrastive learning (or positive-only method) [1, 2, 14] is the state-of-the-art method for visual representation learning. However, they suffer from an inherent problem of matching false positives. We tackle this scene bias issue and improve the quality of learned representation. Note that prior work considering the scene bias for contrastive learning [19, 35, 36, 38] assumed ground-truth object annotations or pre-trained segmentation models, undermining the motivation of self-supervised learning to reduce such supervision. In contrast, we propose a fully self-supervised framework of object localization and debiased contrastive learning.

Bias in visual representation. The bias (or shortcut) in neural networks [58] have got a significant attention recently, pointing out the unintended over-reliance on, e.g., texture [39], background [23] or adversarial features [59]. Numerous works have thus attempted to remove such biases, particularly in an unsupervised manner, e.g., [29, 60, 61]. Our work also lies on this line: we raise a scene bias of contrastive (and positive-only) representation learning and a self-supervised way to fix it.

Unsupervised object localization. The deep-learning-based unsupervised object localization methods can be categorized as follow. (a) The generative-based [50, 62, 63] approaches train a generative model that disentangles the objects and background by enforcing the object-perturbed image to be considered as real. (b) The noisy-ensemble [64–66] approaches train a model using handcrafted predictions as noisy targets. Despite the training is unsupervised, they initialize the weights with the supervised model. (c) Voynov et al. [67] manually finds the 'salient direction' from the noise (latent) of the ImageNet-trained BigGAN [68]. Besides, scene decomposition (e.g., [69]) aims at a more ambitious goal: fully decompose the objects and background, but currently not scale to the complex images. To our best knowledge, the generative-based approach is the state-of-the-art method for fully unsupervised scenarios. Our proposed ContraCAM could be an alternative in this direction.

Class activation map. Class activation map [31, 32] has been used for the weakly-supervised object localization (WSOL), inferring the pixel- (or object-) level annotations using class labels. ContraCAM further expands its applicability by relaxing the requirements of class labels.

5 Conclusion and Discussion

We proposed the ContraCAM, a simple and effective self-supervised object localization method using the contrastively trained models. We then introduced two data augmentations upon the ContraCAM that reduce scene bias and improve the quality of the learned representations for contrastive learning. We remark that the scene bias is more severe for the uncurated images; our work would be a step towards strong self-supervised learning under real-world scenarios [70, 71].

Limitations. Since the ContraCAM finds the most salient regions, it can differ from the desiderata of the users, e.g., the ContraCAM detects both the birds and branches in the CUB [46] dataset, but one may only want to detect the birds. Also, though the ContraCAM identifies the disjoint objects, it is hard to separate the occluded objects. Incorporating the prior knowledge of the objects and designing a more careful method to disentangle objects would be an interesting future direction.

Potential negative impacts. Our proposed framework enforces the model to focus on the "objects", or the salient regions, to disentangle the relations of the objects and background. However, it may occur a different type of bias: over-relying on the conspicuous objects and ignoring the others, could raise a fairness problem [72]. We remark that the biases in datasets and models cannot be entirely eliminated without carefully designed guidelines. Despite the empirical success of the self-supervised representations, we still need a closer look at the models, interactively correcting them.

References

[1] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[2] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*, 2020.

[3] A. Anand, E. Racah, S. Ozair, Y. Bengio, M.-A. Côté, and R. D. Hjelm. Unsupervised state representation learning in atari. In *Advances in Neural Information Processing Systems*, 2019.

[4] A. Srinivas, M. Laskin, and P. Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. In *International Conference on Machine Learning*, 2020.

[5] J. Tack, S. Mo, J. Jeong, and J. Shin. Csi: Novelty detection via contrastive learning on distributionally shifted instances. In *Advances in Neural Information Processing Systems*, 2020.

[6] K. Sohn, C.-L. Li, J. Yoon, M. Jin, and T. Pfister. Learning and evaluating representations for deep one-class classification. In *International Conference on Learning Representations*, 2021.

[7] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021.

[8] T. Afouras, Y. M. Asano, F. Fagan, A. Vedaldi, and F. Metze. Self-supervised object detection from audio-visual correspondence. *arXiv preprint arXiv:2104.06401*, 2021.

[9] A. v. d. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[10] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin. Unsupervised feature learning via non-parametric instance discrimination. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[11] I. Misra and L. v. d. Maaten. Self-supervised learning of pretext-invariant representations. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[12] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *Advances in Neural Information Processing Systems*, 2020.

[13] Y. Tian, D. Krishnan, and P. Isola. Contrastive multiview coding. In *European Conference on Computer Vision*, 2020.

[14] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. D. Guo, M. G. Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. In *Advances in Neural Information Processing Systems*, 2020.

[15] X. Chen and K. He. Exploring simple siamese representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

[16] N. Zhao, Z. Wu, R. W. Lau, and S. Lin. What makes instance discrimination good for transfer learning? In *International Conference on Learning Representations*, 2021.

[17] K. L. Hermann, T. Chen, and S. Kornblith. The origins and prevalence of texture bias in convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2020.

[18] R. Geirhos, K. Narayanappa, B. Mitzkus, M. Bethge, F. A. Wichmann, and W. Brendel. On the surprising similarities between supervised and self-supervised models. *arXiv preprint arXiv:2010.08377*, 2020.

[19] S. Purushwalkam and A. Gupta. Demystifying contrastive self-supervised learning: Invariances, augmentations and dataset biases. In *Advances in Neural Information Processing Systems*, 2020.

[20] Y. Tian, C. Sun, B. Poole, D. Krishnan, C. Schmid, and P. Isola. What makes for good views for contrastive learning. In *International Conference on Machine Learning*, 2020.

[21] T. Xiao, X. Wang, A. A. Efros, and T. Darrell. What should not be contrastive in contrastive learning. In *International Conference on Learning Representations*, 2021.

[22] K. K. Singh, D. Mahajan, K. Grauman, Y. J. Lee, M. Feiszli, and D. Ghadiyaram. Don't judge an object by its context: Learning to overcome contextual bias. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[23] K. Xiao, L. Engstrom, A. Ilyas, and A. Madry. Noise or signal: The role of image backgrounds in object recognition. In *International Conference on Learning Representations*, 2021.

[24] A. Kolesnikov, X. Zhai, and L. Beyer. Revisiting self-supervised visual representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[25] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, 2014.

[26] M.-E. Nilsback and A. Zisserman. A visual vocabulary for flower classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[27] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[28] X. Chen, H. Fan, R. Girshick, and K. He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

[29] H. Wang, S. Ge, Z. Lipton, and E. P. Xing. Learning robust global representations by penalizing local predictive power. In *Advances in Neural Information Processing Systems*, 2019.

[30] D. Hendrycks and T. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019.

[31] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[32] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *IEEE International Conference on Computer Vision*, 2017.

[33] W. Bae, J. Noh, and G. Kim. Rethinking class activation mapping for weakly supervised object localization. In *European Conference on Computer Vision*, 2020.

[34] Y. Wei, J. Feng, X. Liang, M.-M. Cheng, Y. Zhao, and S. Yan. Object region mining with adversarial erasing: A simple classification to semantic segmentation approach. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[35] N. Zhao, Z. Wu, R. W. Lau, and S. Lin. Distilling localization for self-supervised representation learning. In *AAAI Conference on Artificial Intelligence*, 2021.

[36] R. R. Selvaraju, K. Desai, J. Johnson, and N. Naik. Casting your model: Learning to localize improves self-supervised representations. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

[37] J. Lafferty, A. McCallum, and F. C. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning*, 2001.

[38] C. K. Ryali, D. J. Schwab, and A. S. Morcos. Leveraging background augmentations to encourage semantic focus in self-supervised contrastive learning. *arXiv preprint arXiv:2103.12719*, 2021.

[39] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations*, 2019.

[40] D. Hendrycks, S. Basart, N. Mu, S. Kadavath, F. Wang, E. Dorundo, R. Desai, T. Zhu, S. Parajuli, M. Guo, D. Song, J. Steinhardt, and J. Gilmer. The many faces of robustness: A critical analysis of out-of-distribution generalization. *arXiv preprint arXiv:2006.16241*, 2020.

[41] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

[42] S. Yun, D. Han, S. J. Oh, S. Chun, J. Choe, and Y. Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *IEEE International Conference on Computer Vision*, 2019.

[43] K. Lee, Y. Zhu, K. Sohn, C.-L. Li, J. Shin, and H. Lee. i-mix: A domain-agnostic strategy for contrastive representation learning. In *International Conference on Learning Representations*, 2021.

[44] J.-H. Kim, W. Choo, and H. O. Song. Puzzle mix: Exploiting saliency and local statistics for optimal mixup. In *International Conference on Machine Learning*, 2020.

[45] A. Uddin, M. Monira, W. Shin, T. Chung, S.-H. Bae, et al. Saliencymix: A saliency guided data augmentation strategy for better regularization. In *International Conference on Learning Representations*, 2021.

[46] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

[47] J. Choe, S. J. Oh, S. Lee, S. Chun, Z. Akata, and H. Shim. Evaluating weakly supervised object localization methods right. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[48] P. Krähenbühl and V. Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *Advances in Neural Information Processing Systems*, 2011.

[49] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[50] M. Chen, T. Artières, and L. Denoyer. Unsupervised object segmentation by redrawing. In *Advances in Neural Information Processing Systems*, 2019.

[51] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[52] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*, 2019.

[53] M. Sundararajan, A. Taly, and Q. Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, 2017.

[54] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.

[55] A. Krizhevsky et al. Learning multiple layers of features from tiny images, 2009.

[56] L. Bossard, M. Guillaumin, and L. Van Gool. Food-101–mining discriminative components with random forests. In *European Conference on Computer Vision*, 2014.

[57] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. Jawahar. Cats and dogs. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[58] R. Geirhos, J.-H. Jacobsen, C. Michaelis, R. Zemel, W. Brendel, M. Bethge, and F. A. Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2020.

[59] A. Ilyas, S. Santurkar, D. Tsipras, L. Engstrom, B. Tran, and A. Madry. Adversarial examples are not bugs, they are features. In *Advances in Neural Information Processing Systems*, 2019.

[60] M. Minderer, O. Bachem, N. Houlsby, and M. Tschannen. Automatic shortcut removal for self-supervised representation learning. In *International Conference on Machine Learning*, 2020.

[61] J. Nam, H. Cha, S. Ahn, J. Lee, and J. Shin. Learning from failure: Training debiased classifier from biased classifier. In *Advances in Neural Information Processing Systems*, 2020.

[62] A. Bielski and P. Favaro. Emergence of object segmentation in perturbed generative models. In *Advances in Neural Information Processing Systems*, 2019.

[63] R. Arandjelović and A. Zisserman. Object discovery with a copy-pasting gan. *arXiv preprint arXiv:1905.11369*, 2019.

[64] D. T. Nguyen, M. Dax, C. K. Mummadi, T. P. N. Ngo, T. H. P. Nguyen, Z. Lou, and T. Brox. Deepusps: Deep robust unsupervised saliency prediction with self-supervision. In *Advances in Neural Information Processing Systems*, 2019.

[65] J. Zhang, T. Zhang, Y. Dai, M. Harandi, and R. Hartley. Deep unsupervised saliency detection: A multiple noisy labeling perspective. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[66] D. Zhang, J. Han, and Y. Zhang. Supervision by fusion: Towards unsupervised learning of deep salient object detector. In *IEEE International Conference on Computer Vision*, 2017.

[67] A. Voynov, S. Morozov, and A. Babenko. Big gans are watching you: Towards unsupervised object segmentation with off-the-shelf generative models. *arXiv preprint arXiv:2006.04988*, 2020.

[68] A. Brock, J. Donahue, and K. Simonyan. Large scale gan training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019.

[69] M. Engelcke, O. P. Jones, and I. Posner. Genesis-v2: Inferring unordered object representations without iterative refinement. *arXiv preprint arXiv:2104.09958*, 2021.

[70] P. Goyal, M. Caron, B. Lefaudeux, M. Xu, P. Wang, V. Pai, M. Singh, V. Liptchinsky, I. Misra, A. Joulin, et al. Self-supervised pretraining of visual features in the wild. *arXiv preprint arXiv:2103.01988*, 2021.

[71] Y. Tian, O. J. Henaff, and A. v. d. Oord. Divide and contrast: self-supervised learning from uncurated data. *arXiv preprint arXiv:2105.08054*, 2021.

[72] A. Chouldechova and A. Roth. A snapshot of the frontiers of fairness in machine learning. *Communications of the ACM*, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   See Section 5.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   See Section 5.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   We provide our code in the supplementary materials.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   We state the details in Section 3 and Appendix B.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   We report the error bars for the experiments with larger variances (e.g., Table 5).
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
   See common setup in Section 3.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   We cite the datasets and libraries we use.
   (b) Did you mention the license of the assets? [Yes]
   We only use the public datasets and open source libraries.
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]