Unsupervised Object Localization with Representer Point Selection

Yeonghwan Song\(^1\)  Seokwoo Jang\(^1\)  Dina Katabi\(^2\)  Jeany Son\(^1\)
\(^1\)AI Graduate School, GIST  \(^2\)MIT CSAIL
{yeonghwan.song, jangseokwoo}@gm.gist.ac.kr  dina@csail.mit.edu  jeany@gist.ac.kr

Abstract

We propose a novel unsupervised object localization method that allows us to explain the predictions of the model by utilizing self-supervised pre-trained models without additional finetuning. Existing unsupervised and self-supervised object localization methods often utilize class-agnostic activation maps or self-similarity maps of a pre-trained model. Although these maps can offer valuable information for localization, their limited ability to explain how the model makes predictions remains challenging. In this paper, we propose a simple yet effective unsupervised object localization method based on representer point selection, where the predictions of the model can be represented as a linear combination of representer values of training points. By selecting representer points, which are the most important examples for the model predictions, our model can provide insights into how the model predicts the foreground object by providing relevant examples as well as their importance. Our method outperforms the state-of-the-art unsupervised and self-supervised object localization methods on various datasets with significant margins and even outperforms recent weakly supervised and few-shot methods. Our code is available at: https://github.com/yeonghwansong/UOLwRPS

1. Introduction

Object localization is one of the most fundamental problems in computer vision which aims to find a bounding box of a particular object category in a given image. Recent object localization methods achieve impressive performances thanks to advances in deep neural networks (DNNs) [19, 13] and large-scale datasets [33, 37]. Despite their successes, the biggest concern in this task is that collecting datasets with precise box-level annotations is labor-intensive and time-consuming. Recently, object localization methods with less supervision, i.e. weakly-supervised methods [25, 52, 48, 12, 53, 46, 31, 44, 3], have been well-studied to mitigate those problems, nevertheless, they require image-level class labels.

Beyond learning with less supervision, most recent works [42, 41, 51, 1, 21, 27, 36, 45, 35, 39] focus on the object localization task using self-supervised or unsupervised learning that does not require any human annotated labels. These works address the problem of identifying which regions are more likely to contain the foreground object, which is a salient object in an image. In order to discover foreground object regions, several methods [1, 21, 27, 36] attempt to use the magnitude of feature vectors as a clue for class-agnostic activation maps (CAAM) [1]. Most of these methods rely heavily on pre-trained models designed for the image classification task. However transferring these models to the object localization task is challenging, since foreground features are not easily distinguished from the background. To address this problem, these works have been focusing on learning to discriminate foreground and background representations from the pre-trained model.

Recent approaches for unsupervised object localization face a limitation in terms of the explainability of the predictions. DNNs are powerful predictors across various domains, but their complicated structure often makes them black-box functions. Class activation maps (CAM) [56] are frequently used to provide visually interpretable information about a specific class the model has learned, but a detailed explanation of how the model makes its predictions...
remains unclear. This limitation becomes even more severe for CAAM, which is commonly used in self-supervised object localization methods [1, 21, 27, 36].

In this paper, we propose a simple yet effective unsupervised object localization method based on the representer theorem using self-supervised representation learning models. The proposed method formulates the solution to the object localization task based on representer point selection [47]. This strategy is derived from the representer theorem, where the predictions of a neural network for a given test point find expression through a linear combination of the activations originating from training examples. The importance of each feature, which measures the significance of training examples in contributing to the final prediction, is computed by its norm. By selecting representer points that have the most influence on the model’s predictions, our model gains valuable insights into how the model identifies foreground regions. In addition, we introduce a simple extension to weakly supervised object localization, demonstrating zero-shot transferability across different datasets without the need for additional training.

Our experiments are conducted on various datasets for the object localization task to prove the effectiveness of our method and show outstanding performances compared to the state-of-the-art methods with substantial margins. Furthermore, as shown in Figure 1, our method even outperforms some recent weakly supervised and few-shot object localization methods without any fine-tuning or post-processing using additional refinement networks.

We summarize our contributions as follows:

- We propose a novel unsupervised object localization method that leverages the representer theorem on self-supervised representation learning models. The proposed method helps to enhance the interpretability of predictions by providing relevant examples as well as their importance.

- We show that the representer point selection, which has not been extensively explored in previous research on deep neural networks, can be effectively applied to object localization.

- Our method is simple to implement and works well with a variety of self-supervised visual representation learning models. It is also an effective solution due to its good transferability across datasets and easy scalability to weakly supervised object localization.

- Our method shows outstanding results on various datasets for the object localization task and also achieves comparable performances to the state-of-the-art weakly-supervised and few-shot methods.

2. Related Works

Weakly Supervised Object Localization Weakly supervised object localization (WSOL) is a challenging task, aiming to localize an object with only image-level annotations. CAM [56] is the first work to introduce a method of generating a class activation map by projecting the weights of the classifier on the feature maps. However, CAM has a limitation in finding parts of the objects that are not relevant for classification. To address this problem, various CAM-based WSOL methods have been proposed [25, 52, 12, 53, 46, 31, 45, 44, 3]. HaS [25] used the random masking of image patches to encourage the model to observe more object regions. ACoL [52] adopted two adversarial classifiers to catch the relevant parts. ADL [12] and AE [43] progressively erased the discriminative parts on the feature map of each convolutional layer, ensuring that the focus was not solely on these discriminative areas. SPG [53] and i2C [54] utilized shallow convolutional feature maps and pixel-level correlations between two different images to adjust an activation map respectively. PSOL [50] proposed a new task of pseudo-supervised object localization which consists of two independent tasks: class-agnostic object localization and object classification. PSOL [50] and SEM [55] trained a localizer with an object bounding box generated by DDT [42] and edges of an input image extracted from the canny edge detector as pseudo labels, respectively. Recently, most CAM-based methods [46, 40, 31, 45, 44, 3] have focused on learning discriminative features for the foreground and background, while vision transformer-based methods [14, 17, 10, 2] have emerged to exploit long-range dependency.

Unsupervised Object Localization Unsupervised object discovery is strongly related to co-localization [42, 1, 21] and co-segmentation [49, 28] tasks. Early work for unsupervised object discovery [11] proposed a part-based region matching using off-the-shelf region proposals. With the advance of deep learning, DDT [42] and SCDA [41] introduced methods to mine the statistical properties of object regions in an image using features from pre-trained CNNs. MO [51] mined a frequently activated pattern from multi-scale feature maps. These methods achieved impressive results not only in object discovery but also in co-localization, which aims to localize objects in a dataset that consists of only one coarse category. PsyNet [1] introduced a simple and practical framework for the co-localization task using self-supervised learning and class-agnostic activation mapping. Ki et al. [21] employed contrastive learning with the assumption of consistency between image-level and activation-level transformations. Su et al. [36] proposed a joint graph partition method splitting foreground and background regions in a paired image feature graph. C²AM [45] proposed to apply contrastive learning to disentangle the
feature map into foreground and background with an additional trainable layer generating a class-agnostic activation map. LOST [35] and TokenCut [39] utilize a pre-trained vision transformer model [13] by DINO [5], which has the ability to produce fine segmentation masks using attention maps with [CLS] token. LOST [35] suggested a heuristic seed selection and expansion method, and TokenCut [39] employed a normalized cut algorithm on the graph using the similarity between patches.

Self-supervised Visual Representation Learning  Self-supervised visual representation learning (SSL) methods have been proposed to learn meaningful visual representation without manual annotations. The network learns visual representation by training pretext tasks rather than the task intended to be solved. For example, Noroozi et al. [32] trained CNN to solve a jigsaw puzzle, Larsson et al. [26] reconstructed color images from grayscale images, and Gidaris et al. [15] predicted discrete rotation angles of rotated images. Recently, contrastive learning-based SSL methods [18, 6, 8, 16] have emerged with comparable performances to the supervised learning methods. They focus on modeling image similarity and dissimilarity between pairs of images [6, 18] or consider image similarity [16, 8] based on data augmentation. DINO [5] proposed a knowledge distillation strategy to train the vision transformer model. These pre-trained SSL models are utilized in various downstream tasks of computer vision with fine-tuning. On the contrary, in this work, we focus on leveraging the pre-trained SSL models for the unsupervised object localization task without any fine-tuning.

3. Background

Representer Theorem. Representer theorem [34] provides a mathematical basis for traditional machine learning methods. Let $L(x_n, y_n, w)$ be the loss function, and $\frac{1}{N} \sum_{n=1}^{N} L(x_n, y_n, w)$ be the empirical risk over a reproducing kernel Hilbert space (RKHS) $H_k$. Then the representer theorem states that an optimal solution of a regularized empirical risk minimization (ERM) problem can be represented as a linear combination of a positive definite kernel $k$ on the input set $\mathcal{X}$ over $H_k$. The optimal parameter $w^*$ can be expressed through ERM as follows:

$$w^* = \arg\min_{w \in H_k} \left\{ \frac{1}{N} \sum_{n=1}^{N} L(x_n, y_n, w) + \lambda ||w||^2 \right\}. \quad (1)$$

This can be rewritten by the representer theorem

$$w^*(\cdot) = \sum_{i=1}^{N} \alpha_i k(\cdot, x_i) = \sum_{i=1}^{N} \alpha_i \langle \varphi(\cdot), \varphi(x_i) \rangle, \quad (2)$$

where $\alpha_i \in \mathbb{R}$, and $\varphi$ denotes a mapping function into $H_k$. For further details for the Representer theorem, please refer to the work by [34, 47].

Representer Point Selection for DNNs. The representer theorem is originally developed for non-parametric predictors, where the parameters lie in a reproducing kernel Hilbert space. In deep neural networks, however, it is difficult to find a global solution for empirical risk minimization. To address this issue, [47] proposes a technique to decompose the pre-activation predictions into a linear combination of the activation values of the training points. The neural network can be represented as

$$\hat{y}_t = \sigma(\phi(x_t, W_{\Phi}, w_{\phi})) = \sigma(w_{\phi}^T f_t), \quad (3)$$

where $\sigma$ is the activation function, $f_t = \Phi(x_t, W_{\Phi})$ is the last intermediate layer feature of a training example $x_n$ in DNNs, $\phi$ denotes the last prediction layer for a specific task, and $W_{\phi}$ and $w_{\phi}$ denote the parameters of $\Phi$ and $\phi$, respectively. Let $W^*$ be a stationary point of the objective (1). Then we have decomposition as follows:

$$w_{\phi}^T f_t = \sum_{n=1}^{N} \alpha_n k(x_t, x_n) = \sum_{n=1}^{N} \alpha_n f_n^{\top} f_t, \quad (4)$$

where $\alpha_n = \frac{1}{-2\lambda N} \frac{\partial L(x_n, y_n, W)}{\partial \phi(f_n, w_{\phi})}$.

and $\alpha_n f_n^{\top} f_t$ is a representer value of a training example $x_n$ given a test sample $x_t$. This representer value is assigned to each training point reflect their importance on the learned parameters and $\alpha_n$ is the global sample importance which is used to evaluate the importance of the training data $x_n$. Consequently, this approach enables the selection of representer points – training instances with either large or small representer values – to enhance comprehension of the model’s predictions.

4. Method

In this section, we discuss how we leverage the representer point selection [47] to solve unsupervised object localization (UOL) task, and our extension to weakly supervised object localization (WSOL) and zero-shot transferring.

4.1. Representer Point Selection for UOL

Inspired by the representer point selection, we formulate the object localization task as identifying the important regions in training examples, called representer points, using a pre-trained self-supervised model. Unlike other tasks such as classification, regression, and retrieval that are covered in [47], the object localization task requires dense predictions within a single image. To address this, we address all elements within image feature maps rather than relying solely on a single image-level feature vector.
As our focus lies in unsupervised learning, the empirical risk or the loss function can be defined as
\[
\frac{1}{N_D} \sum_{n=1}^{N_D} L(x_n, W_\Phi) \text{ without a label } y_n \text{ given pre-trained encoder } \Phi \text{ and training set } D.
\]
Then the activation maps \( A_i \) for foreground regions can be computed by aggregating feature maps using the global sample importance, which indicate how much each training example contributes to the prediction of being foreground:
\[
vec(A_t)_j = \frac{1}{N_D} \sum_{n=1}^{N_D} \sum_{i=1}^{|W|} \alpha_{n,i} f_{n,i}^T \hat{f}_{t,j} = w^* \hat{f}_{t,j},
\]
where \( f_{n,i} = \Phi(x_n, W_\Phi) \), denotes \( i \)-th feature vector in the feature map of an image \( x_n \), which has been encoded using a self-supervised pre-trained encoder \( \Phi \), \( \hat{f} = \frac{f}{\|f\|} \) indicates the normalized feature vector, \( \alpha \) is the global sample importance, and \( N_D, |W|, H \) denote the number of images in the training dataset, the width of the feature map, and the height of the feature map, respectively. Here, we refer to \( \alpha_{n,i} \hat{f}_{n,i}^T \hat{f}_{t,j} \) as the representer value for each training image patch \( x_{n,i} \) given a testing image patch \( x_{t,j} \), corresponding to receptive fields of \( f_{n,i} \) and \( \hat{f}_{t,j} \), respectively. In Figure 2, we illustrate the entire structure of the proposed representer point selection method to compute a foreground predictor \( w^* \) and inference pipelines using it.

**Global Sample Importance.** \( \alpha_n \) is referred to as the global sample importance since it is independent of the testing examples but reliant on the empirical risk of the training set. In order to tackle the unsupervised localization task, we first design the loss function which is suitable for an unsupervised setting to compute \( \alpha \). Let \( w \) be the parameters of a binary classifier for foreground predictions given feature vectors, then the loss function can be written as follows:
\[
L(x_n,i, y_n,i, W_\Phi) = -y_n,i w^T \hat{f}_{n,i},
\]
where \( y_n,i \in \{-1, 1\} \) denotes a label of \( x_{n,i} \), which is an image patch corresponding to a receptive field of \( f_{n,i} \). In the unsupervised object localization task, however, the goal is to find foreground regions in images without the use of labels. Thus, instead of using \( y_n,i \), we use the norm of the feature vector to determine a soft pseudo label of the example for the objective function as follows:
\[
L(x_n,i, W_\Phi) = -\left(\|f_{n,i}\| - \tau\right)w^T \hat{f}_{n,i},
\]
where \( \tau \) denotes a foreground threshold.

We estimate soft labels of inputs based on the norm of feature vectors because the norm can serve as an important indicator of the features that the network emphasized during training with the self-supervised contrastive loss. In other words, a higher norm of a feature vector indicates that the corresponding feature had more weight in the model training. These features are extracted from regions of the image that contain rich information to distinguish them from other instances, and these regions are more likely to be associated with significant objects. Likewise, prior self-supervised co-localization studies [1, 21, 27, 36] also use the magnitude of feature vectors as a clue to discover object regions.

Hence, the global sample importance is computed as:
\[
\alpha_{n,i} = \frac{1}{2\lambda N_D HW} \frac{\partial L(x_n,i, W_\Phi)}{\partial w^T \hat{f}_{n,i}}
\]
\[
= \frac{1}{2\lambda N_D HW} (\|f_{n,i}\| - \tau) = \frac{1}{C} (\|f_{n,i}\| - \tau).
\]

Here, the sign of the global sample importance is related to whether to representer points – training examples that have large or small representer values – contribute to the foreground or background. In order to have large representer values, both \( \alpha \) and similarity between two features, expressed as \( f_{n,i}^T f_t \), should have large values. By analyzing the representer values, we can gain insight into how the model predicts foreground regions by providing relevant examples.
as well as their importance. Figure 3 shows an example of global sample importance, the similarity between features, and representer values for given points in a given test image.

**Straightforward Computation of $w^*$** We then compute a foreground predictor $w^*$ using $\alpha_{n,i}$ in Eq (10) as follows:

$$w^* = \frac{1}{C} \sum_{n=1}^{N_D} \sum_{i=1}^{H,W} f_{n,i} - \tau \sum_{n=1}^{N_D} \sum_{i=1}^{H,W} \hat{f}_{n,i},$$

The optimal $w^*$ can be computed by taking the difference between the mean feature vector of all images in the dataset and the mean of the normalized feature vectors of all images in the dataset, multiplied by two constant values. Algorithm 1 illustrates the Python-style pseudo code of the proposed representer point selection method to compute the foreground predictor $w^*$ and it clearly shows the simplicity and reproducibility of our method.

**4.2. Towards WSOL and Zero-shot Transferring**

Our method can be easily extended to weakly supervised and zero-shot transferring settings with a simple modification, since activation maps are computed by the linear combination of activations of training points directly without learning. Given a class label $c$, the activation map $A^c$ is computed as follows:

$$\text{vec}(A^c_j) = \sum_{n \in D_c} \sum_{i=1}^{H,W} \alpha_{n,i} \hat{f}_{n,i}^T \hat{f}_{t,j} = w^c_\tau \hat{f}_{t,j},$$

where $D_c$ denotes the subset of the training dataset $D$ that contains only the images assigned to class $c$. Note that, although WSOL allows for the use of image-level class labels, we utilize an off-the-shelf classification model to determine the class of each image in $D$ during our experiments in order to ensure a fair comparison with other methods [45, 50]. Extending our method to zero-shot transferring scenarios can be simply done by computing the weight vector $w^*$ on the different datasets in which the classes do not overlap with the test dataset.

**5. Experiments**

**5.1. Implementation Details**

We adopt ResNet50 [19] and ViT-S [13] as backbones for SSL pre-trained model, utilizing publicly available pre-trained weights. We use the input image size of 224×224 for the representer point selection. To evaluate object localization performances on fine-grained classification datasets, the input image is resized to 256×256 and then center-cropped to a size of 224×224. We use $\|v\|/\|u\|$ for $\tau$, and min-max normalization to estimate the bounding box from the generated activation map. We also set $\lambda = 0.001$ as in [47], but changing this value does not have an impact on the final results due to the use of min-max normalization.

**5.2. Datasets and Evaluation Metrics**

We conduct experiments on the four fine-grained datasets for unsupervised object localization to evaluate our
The following PSOL [50].

Table 1. Comparison between our method and the state-of-the-art self-supervised/unsupervised methods in terms of GT-Known Loc performance on ImageNet-1K and four fine-grained datasets: CUB200-2011, Stanford Cars, FGVC-Aircraft, and Stanford Dogs. * denotes our reproduced results from their official code. † and ‡ indicate a backbone initialized by MoCo v2 and DINO pre-trained weights, respectively.

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>S</th>
<th>T</th>
<th>CUB-200-11</th>
<th>Stanford Cars</th>
<th>FGVC-Aircraft</th>
<th>Stanford Dogs</th>
<th>ImageNet-1K</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ours ViT-S</td>
<td>✔</td>
<td>✓</td>
<td>✔</td>
<td>87.72</td>
<td>97.43</td>
<td>97.86</td>
<td>85.45</td>
<td>-</td>
</tr>
<tr>
<td>Ours ResNet50</td>
<td>✔</td>
<td>✓</td>
<td>✔</td>
<td>88.83</td>
<td>97.73</td>
<td>96.72</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PsyNet</td>
<td>✔</td>
<td></td>
<td></td>
<td>85.10</td>
<td>98.81</td>
<td>97.81</td>
<td>77.84</td>
<td>-</td>
</tr>
<tr>
<td>Ki et al.</td>
<td>✔</td>
<td></td>
<td></td>
<td>85.93</td>
<td>98.95</td>
<td>98.75</td>
<td>80.32</td>
<td>-</td>
</tr>
<tr>
<td>PSOL</td>
<td>✔</td>
<td></td>
<td></td>
<td>90.00</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>65.44</td>
</tr>
<tr>
<td>C2AM</td>
<td>✔</td>
<td></td>
<td>✔</td>
<td>89.90</td>
<td>99.44*</td>
<td>98.65*</td>
<td>89.91*</td>
<td>66.51</td>
</tr>
<tr>
<td>C2AM (+PSOL)</td>
<td>✔</td>
<td></td>
<td>✔</td>
<td>91.54</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>68.07</td>
</tr>
</tbody>
</table>

Table 2. Comparison of localization performances in terms of Top-1, Top-5 and GT-known Loc on CUB-200-2011 test set and ImageNet-1K validation set. Self-supervised and unsupervised methods including ours employ an additional classifier to evaluate in WSOL setup following PSOL [50]. T and S indicate whether each method uses training and image-level supervision (B for bounding box annotations), respectively. †, ‡, and ◦ indicate the use of MoCo v2, BYOL, and DINO pre-trained weights, respectively.

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>S</th>
<th>T</th>
<th>CUB-200-11</th>
<th>Top-1 Loc</th>
<th>Top-5 Loc</th>
<th>GT-Known</th>
<th>ImageNet-1K</th>
<th>Top-1 Loc</th>
<th>Top-5 Loc</th>
<th>GT-Known</th>
</tr>
</thead>
<tbody>
<tr>
<td>Few-shot method</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPNet</td>
<td>✔</td>
<td>(B)</td>
<td>✔</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Weakly supervised methods</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FAM</td>
<td>✔</td>
<td>✓</td>
<td>✔</td>
<td>73.74</td>
<td>-</td>
<td>85.73</td>
<td>-</td>
<td>64.56</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CREAM</td>
<td>✔</td>
<td>✓</td>
<td>✔</td>
<td>76.03</td>
<td>-</td>
<td>89.88</td>
<td>-</td>
<td>69.31</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BGC</td>
<td>✔</td>
<td>✓</td>
<td>✔</td>
<td>73.16</td>
<td>86.68</td>
<td>91.60</td>
<td>-</td>
<td>69.89</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DAOL</td>
<td>✔</td>
<td>✓</td>
<td>✔</td>
<td>66.65</td>
<td>-</td>
<td>81.83</td>
<td>-</td>
<td>70.27</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BAS</td>
<td>✔</td>
<td>✓</td>
<td>✔</td>
<td>77.25</td>
<td>90.08</td>
<td>95.13</td>
<td>-</td>
<td>71.77</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BagCAM</td>
<td>✔</td>
<td>✓</td>
<td>✔</td>
<td>69.67</td>
<td>-</td>
<td>94.01</td>
<td>-</td>
<td>72.08</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TS-CAM</td>
<td>✔</td>
<td>✓</td>
<td>✔</td>
<td>71.30</td>
<td>83.80</td>
<td>87.70</td>
<td>-</td>
<td>67.60</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LCTR</td>
<td>✔</td>
<td>✓</td>
<td>✔</td>
<td>79.20</td>
<td>89.90</td>
<td>92.40</td>
<td>-</td>
<td>68.70</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VTOL</td>
<td>✔</td>
<td>✓</td>
<td>✔</td>
<td>-</td>
<td>-</td>
<td>80.90</td>
<td>-</td>
<td>72.51</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SCM</td>
<td>✔</td>
<td>✓</td>
<td>✔</td>
<td>76.40</td>
<td>91.60</td>
<td>96.60</td>
<td>-</td>
<td>68.80</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Self-supervised methods</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSOL</td>
<td>✔</td>
<td></td>
<td>✔</td>
<td>70.68</td>
<td>86.64</td>
<td>90.00</td>
<td>-</td>
<td>65.44</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C2AM</td>
<td>✔</td>
<td></td>
<td>✔</td>
<td>74.76</td>
<td>87.37</td>
<td>91.54</td>
<td>-</td>
<td>68.07</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C2AM (+PSOL)</td>
<td>✔</td>
<td></td>
<td>✔</td>
<td>76.40</td>
<td>91.60</td>
<td>96.60</td>
<td>-</td>
<td>68.80</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>w/o finetuning</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LOST</td>
<td>✔</td>
<td></td>
<td>✔</td>
<td>71.30</td>
<td>-</td>
<td>89.70</td>
<td>-</td>
<td>60.00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TokenCut</td>
<td>✔</td>
<td></td>
<td>✔</td>
<td>72.90</td>
<td>-</td>
<td>91.80</td>
<td>-</td>
<td>65.40</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ours</td>
<td>✔</td>
<td></td>
<td>✔</td>
<td>79.57</td>
<td>92.60</td>
<td>96.62</td>
<td>-</td>
<td>69.10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ours</td>
<td>✔</td>
<td></td>
<td>✔</td>
<td>79.55</td>
<td>92.58</td>
<td>96.72</td>
<td>-</td>
<td>69.93</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ours</td>
<td>✔</td>
<td></td>
<td>✔</td>
<td>74.97</td>
<td>84.24</td>
<td>91.61</td>
<td>-</td>
<td>71.96</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Method: CUB-200-2011 [37], Stanford Cars [23], FGVC-Aircraft [30], and Stanford Dogs [20]. We also evaluate our method on ImageNet-1K to compare with other state-of-the-art methods in unsupervised and WSOL setups. In addition, we evaluate the segmentation quality of our method by comparing it with the WSOL state-of-the-art methods on CUB-200-2011 and OpenImages-30K [4] datasets.

Following [56], we use Top-1 (Top-1 Loc), Top-5 (Top-5 Loc) and GT-known localization (GT-Known Loc) accuracy for evaluation. GT-known Loc computes the ratio of the images where the Intersection over Union (IoU) between the estimated bounding box and the known ground-truth boxes is greater than 50%. Top-1 Loc and Top-5 Loc compute the ratio of the images correctly classified with Top-1 and Top-5.
Table 3. Comparison between the proposed method and the state-of-the-art weakly supervised object localization methods in terms of \(\text{PxAP}\) and \(\text{PloU}\) on CUB-200-2011 and OpenImages-30K. * denotes that the results are obtained from their official code.

<table>
<thead>
<tr>
<th>Method</th>
<th>CUB-200-2011</th>
<th>OpenImages-30K</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PloU</td>
<td>PxAP</td>
</tr>
<tr>
<td>wsOL methods</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BGC [22]</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CREAM [46]</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>DAOL [58]</td>
<td>56.18</td>
<td>74.70</td>
</tr>
<tr>
<td>BagCAM [22]</td>
<td>74.51</td>
<td>90.38</td>
</tr>
<tr>
<td>Self-supervised methods</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C(^2)AM [45]</td>
<td>69.65*</td>
<td>88.74*</td>
</tr>
<tr>
<td>Ours with SSL/WSOL pre-trained backbone</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MoCo v2 [7] + Ours</td>
<td>71.28</td>
<td>87.03</td>
</tr>
<tr>
<td>DenseCL [38] + Ours</td>
<td>70.11</td>
<td>87.74</td>
</tr>
<tr>
<td>DINO ViT-S [5] + Ours</td>
<td>78.97</td>
<td>94.37</td>
</tr>
<tr>
<td>C(^2)AM [45] + Ours</td>
<td>71.54</td>
<td>90.22</td>
</tr>
<tr>
<td>DAOL [58] + Ours</td>
<td>67.37</td>
<td>85.99</td>
</tr>
</tbody>
</table>

5 predictions, respectively, with the condition of \(\text{GT-Known Loc}\). For segmentation evaluation, \(\text{Pixel-wise average precision (PxAP)}\) computes the area under the pixel precision-recall curve and \(\text{Peak-IoU (PloU)}\) is the best IoU score with various thresholds.

5.3. Results

Unsupervised Object Localization. Our method is primarily designed for the unsupervised object localization (UOL) task which does not focus on localizing objects for specific categories. In this task, \(w^*\) is computed using the entire dataset, regardless of the number of categories in the dataset. We report the performances of our method and the state-of-the-art methods in terms of \(\text{GT-known Loc}\) on five commonly used datasets in Table 1. We compare our method with recent self-supervised object localization methods including ORE [27], PsyNet [1], Ki et al. [21], JGP [36] and \(C^2\)AM [45], as well as object localization methods without finetuning including DDT [42], MO [51], LOST [35], and TokenCut [39]. As shown in Table 1, our method significantly surpasses other methods on all benchmark datasets. Our method outperforms not only unsupervised methods but also the self-supervised methods where the models are finetuned on the training split of the target datasets. Even with a shallower architecture, such as VGG-16, our method shows better performance compared to the methods using ResNet50 on CUB-200-2011. For the VGG-16 backbone, we employ ImageNet pre-trained weights.

Weakly Supervised Object Localization. In addition, as mentioned in Section 4.2, our method can be extended to the weakly supervised object localization (WSOL) setting by computing \(w^*_c\) for each class \(c\) using a pre-trained classifier. To evaluate on \(\text{Top-1/5 Loc metrics}\), we follow [50] which divides the WSOL task into two sub-tasks: object localization and classification. We first localize objects, and then we determine classes for localized objects using the pre-trained classifier, i.e., ResNet50 pre-trained on ImageNet.

Table 2 shows the performance of our method and the state-of-the-art weakly supervised object localization methods, including FAM [31], CREAM [46], BGC [22], BAS [44], DAOL [58], BagCAM [57], TS-CAM [14], LCTR [10], ViTOL [17], and SCM [2] on CUB-200-2011 and ImageNet-1K datasets. We also compare our method with PSOL [50] and \(C^2\)AM [45] which are self-supervised methods. Our method shows outstanding results on CUB-200-2011 and comparable results on ImageNet-1K, compared to the advanced weakly supervised methods without any supervision and training. As shown in Table 2, our method outperforms all self-supervised methods with the additional classifier using the same architecture with localization backbone [19, 13]. For a fair comparison, we report \(C^2\)AM initialized with MoCo v2 pre-trained weights as well as its refinements with PSOL. Note that, without any training and post-processing as PSOL, our method significantly surpasses \(C^2\)AM (+PSOL) and PSOL on CUB-200-2011. For a fine-grained dataset such as CUB-200-2011, we utilize a class-agnostic foreground predictor \(w^*\) for comprehensive object localization rather than a class-specific one.

Our method shows outstanding performance on not only localization but also segmentation, as shown in Table 3, particularly with the model trained by self-supervised pixel-wise contrastive learning (i.e. DenseCL [38]) on the OpenImages-30K dataset. All the methods compared in this table employ ResNet50 as their backbone, with the exception of DINO. Upon utilizing the DINO ViT model as our backbone, we observe a noteworthy and substantial enhancement in performance. Exploiting our method as an inference method for feature extractors trained by the WSOL methods [45, 58] further enhances performance, yielding

Table 4. \(\text{GT-Known Loc}\) results of our method on various SSL models with ResNet50 backbones except DINO ViT.

<table>
<thead>
<tr>
<th>Model</th>
<th>CUB</th>
<th>ImageNet</th>
</tr>
</thead>
<tbody>
<tr>
<td>MoCo v2 + CAAM</td>
<td>86.93</td>
<td>61.19</td>
</tr>
<tr>
<td>MoCo v2 + Ours</td>
<td>96.67 (+9.74)</td>
<td>69.10 (+7.91)</td>
</tr>
<tr>
<td>(C^2)AM [45]</td>
<td>89.90</td>
<td>66.81</td>
</tr>
<tr>
<td>(C^2)AM [45] + PSOL</td>
<td>91.54 (+1.64)</td>
<td>68.07 (+1.56)</td>
</tr>
<tr>
<td>(C^2)AM [45] + Ours</td>
<td>96.86 (+6.96)</td>
<td>69.80 (+3.29)</td>
</tr>
<tr>
<td>DAOL [58]</td>
<td>81.33</td>
<td>70.27</td>
</tr>
<tr>
<td>DAOL [58] + Ours</td>
<td>89.44 (+7.61)</td>
<td>70.97 (+0.70)</td>
</tr>
</tbody>
</table>

Table 5. \(\text{GT-Known Loc}\) results of models with and without our inference using the representer point selection.

<table>
<thead>
<tr>
<th>Method</th>
<th>CUB</th>
<th>ImageNet</th>
</tr>
</thead>
<tbody>
<tr>
<td>MoCo v2 + CAAM</td>
<td>86.93</td>
<td>61.19</td>
</tr>
<tr>
<td>MoCo v2 + Ours</td>
<td>96.67 (+9.74)</td>
<td>69.10 (+7.91)</td>
</tr>
<tr>
<td>(C^2)AM [45]</td>
<td>89.90</td>
<td>66.81</td>
</tr>
<tr>
<td>(C^2)AM [45] + PSOL</td>
<td>91.54 (+1.64)</td>
<td>68.07 (+1.56)</td>
</tr>
<tr>
<td>(C^2)AM [45] + Ours</td>
<td>96.86 (+6.96)</td>
<td>69.80 (+3.29)</td>
</tr>
<tr>
<td>DAOL [58]</td>
<td>81.33</td>
<td>70.27</td>
</tr>
<tr>
<td>DAOL [58] + Ours</td>
<td>89.44 (+7.61)</td>
<td>70.97 (+0.70)</td>
</tr>
</tbody>
</table>
We adopt the GT-Known Loc effectiveness of our method. We report Robustness across various SSL pre-trained models. We conduct comprehensive ablation studies to verify the effectiveness of our method. We adopt the GT-Known Loc metric as a performance evaluation for all experiments.

Figure 4. GT-Known scores with respect to different thresholds using various pre-trained models on the CUB-200-2011 dataset. The proposed threshold τ consistently achieves near-optimal scores.

significant improvements. It clearly shows that our method demonstrates consistently good performances on various datasets and diverse metrics.

5.4. Ablation Study

We conduct comprehensive ablation studies to verify the effectiveness of our method. We adopt the GT-Known Loc metric as a performance evaluation for all experiments.

Robustness across various SSL pre-trained models. We report GT-known Loc scores of our method using popular self-supervised pre-trained models including BYOL [16], MoCo v2 [7], MoCo v3 [9], DINO [5] and SimSiam [8] to support compatibility of our method in Table 4. Our method consistently shows good performance with various pre-trained models. We believe that our method generates more precise activation map of the object, as long as good representations are provided by pre-trained models no matter how they are trained.

Effectiveness of representer point selection. Table 5 reports the results of our inference method using representer point selection on several models including C2AM [45] and DAOL [58] as well as class-agnostic activation map (CAAM) proposed in [1]. The backbones for our model and C2AM are initialized by MoCo v2 [7] pre-trained weights. Even with the state-of-the-art models trained for the object localization task, applying our inference improves their performance with significant margins on both CUB-200-2011 and ImageNet-1K datasets. In the case of C2AM, our inference significantly improves its performance more than using post-processing by PSOL [50] which requires additional training using a ResNet50 network.

Zero-shot transferability across datasets. We also examine the transferability of our model across different datasets in Table 6. We compare our method’s performance with C2AM [45], using our foreground predictor and C2AM’s trained model from CIFAR10, CIFAR100 [24] and CUB-200-2011 datasets, and then testing on other datasets without additional training. In this comparison, we ensure fairness by using class-agnostic weights and initializing the ResNet50 model with pre-trained weights from MoCo v2 [7]. The results in Table 6 highlight that our model, leveraging representer points from these datasets, can be transferred to various datasets with strong performance, while C2AM’s performance notably drops.

Impact of sampling rate on a dataset D. In order to show the impact of the number of samples for selecting representer points, we randomly sample images on the training set. In Table 7, to illustrate the effect of sampling rate, we report the results of our method with DINO ViTs using 100%, 10%, 1%, 0.1% and 0% of the dataset, where 0% uses CIFAR10 dataset to establish representer points. For 10%, 1%, and 0.1% sampling rates, we compute mean and standard deviation with 10 random trials. As shown in Table 7, the number of samples to compute the foreground predictor has no significant effect on performance drops. In fact, in certain instances, it even produces better results compared to using the entire dataset. We also report results of ViT-s based models, such as TokenCut [39], ViTOL [17], and a few-shot method, SPNet [29]. SPNet uses 5-shots for 200 fine-grained classes on CUB-200-2011 and 10-shots for 1K classes on ImageNet-1K, with object localization annotations, while ViTOL and TokenCut uses a full dataset to train the model. Even with an extremely small amount of samples in the dataset, our method surpass SPNet, ViTOL,
and TokenCut on ImageNet.

**Impact of threshold** $\tau$. We conduct an empirical analysis to show the impact of a threshold $\tau = \|v\|/\|u\|$. As demonstrated by the results in Figure 4, we found that the proposed threshold $\tau$ consistently achieved nearly optimal scores, even when tested with various thresholds. Our defined threshold $\tau$ also consistently performs well across diverse models. Further detailed theoretical analysis for $\tau$ is provided in our supplementary material.

5.5. Qualitative Results and Failure Cases

We illustrate the qualitative results of our method and C$^2$AM in Figure 5. Both methods use the MoCo v2 model with the ResNet50 backbone. Without any training on the dataset unlike C$^2$AM, our method successfully localizes objects on various datasets. Compared with the other method, our activation map shows the more precise area of the object, and activations in the background are suppressed.

We also demonstrate some failure cases in Figure 6. Since our method only relies on features from the pre-trained networks without additional training, it has limited ability to find the small target object particularly in scenarios where multiple objects coexist within the image. We observe that our method tends to find objects with a dominant size in the image. Moreover localizing the whole object is challenging in cases where the object is either occluded or fragmented. These failure cases, however, are not only our limitations but also the limitations of current unsupervised and weakly supervised object localization methods.

6. Conclusion

In this paper, we propose a simple and effective unsupervised object localization method without using additional training and supervision. We leverage the representer point selection in the object localization task, which allows us to explain the decision in terms of the activations of data points. The proposed method outperforms not only the state-of-the-art self-supervised and unsupervised object localization methods by a significant margin but also surpasses recent weakly supervised and few-shot methods.

Acknowledgement. This work was supported by the IITP grants (No.2019-0-01842, No.2021-0-02068, No.2022-0-00926) funded by MSIT, the ISTD program (No.20018334) funded by MOTIE, and the GIST-MIT Research Collaboration grant funded by GIST, Korea.
References


[23] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In ICCV workshops, 2013. 6


[31] Meng Meng, Tianzhu Zhang, Qi Tian, Yongdong Zhang, and Feng Wu. Foreground activation maps for weakly supervised object localization. In ICCV, 2021. 1, 2, 6, 7


[36] Yukun Su, Guosheng Lin, Yun Hao, Yiwen Cao, Wenjun Wang, and Qingyao Wu. Self-supervised object localization with joint graph partition. In *AAAI*, 2022. 1, 2, 4, 6, 7


[44] Pingyu Wu, Wei Zhai, and Yang Cao. Background activation suppression for weakly supervised object localization. In *CVPR*, 2022. 1, 2, 6, 7

[45] Jinheng Xie, Jianfeng Xiang, Junliang Chen, Xianxu Hou, Xiaodong Zhao, and Linlin Shen. C2am: Contrastive learning of class-agnostic activation map for weakly supervised object localization and semantic segmentation. In *CVPR*, 2022. 1, 2, 5, 6, 7, 8


[48] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, 2019. 1


[50] Chen-Lin Zhang, Yun-Hao Cao, and Jianxin Wu. Rethinking the route towards weakly supervised object localization. In *CVPR*, 2020. 2, 5, 6, 7, 8


[54] Xiaolin Zhang, Yunchao Wei, and Yi Yang. Inter-image communication for weakly supervised localization. In *ECCV*, 2020. 2


[58] Lei Zhu, Qi She, Qian Chen, Yunfei You, Boyu Wang, and Yanye Lu. Weakly supervised object localization as domain adaption. In *CVPR*, 2022. 6, 7, 8