SILCO: Show a Few Images, Localize the Common Object

Tao HU, Pascal Mettes, Jia-Hong Huang and Cees G. M. Snoek
University of Amsterdam

Abstract

Few-shot learning is a nascent research topic, motivated by the fact that traditional deep learning requires tremendous amounts of data. In this work, we propose a new task along this research direction, we call few-shot common-localization. Given a few weakly-supervised support images, we aim to localize the common object in the query image without any box annotation. This task differs from standard few-shot settings, since we aim to address the localization problem, rather than the global classification problem. To tackle this new problem, we propose a network that aims to get the most out of the support and query images. To that end, we introduce a spatial similarity module that searches the spatial commonality among the given images. We furthermore introduce a feature reweighting module to balance the influence of different support images through graph convolutional networks. To evaluate few-shot common-localization, we repurpose and reorganize the well-known Pascal VOC and MS-COCO datasets, as well as a video dataset from ImageNet VID. Experiments on the new settings for few-shot common-localization shows the importance of searching for spatial similarity and feature reweighting, outperforming baselines from related tasks.

1. Introduction

Convolutional networks exhibit superior accuracy in a wide variety of computer vision challenges, but a key limitation remains their hunger for labeled data [8, 33, 45]. Typically, large amounts of annotated examples are required to achieve a high accuracy. This issue becomes even more severe for localization tasks, which typically require additional localized annotations [31, 38, 39]. In recent years, a new research line has emerged that strives to learn new concepts from limited amounts of data, known as few-shot learning [43, 48]. Though widely explored in tasks like image classification, few-shot learning is rarely considered for object localization problems.

In this paper, we propose the new task of few-shot common-localization, which takes N support images (without box annotations) and one query image as input and tries to localize the common object in the query image guided by the support images. Our task is demonstrated in Figure 1 for four support images and one query image. Unique to our task is we only know there is a common class among the support and query images, but the class itself and its spatial extent is unknown. In practice, we can obtain query and support images with a common class by leveraging social tags, hash tags, or an off-the-shelf image classification network. Our method adds the bounding box for free.

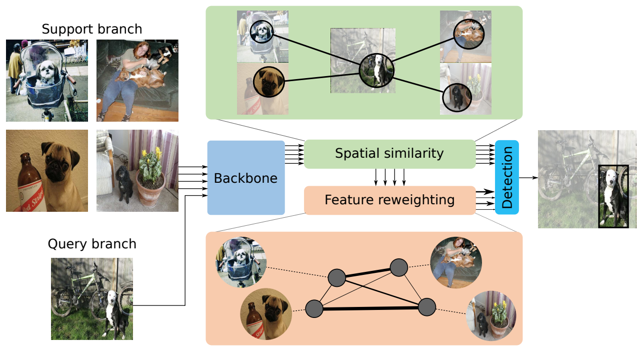

Figure 1. **Few-shot common-localization.** Starting from a few weakly-supervised support images and a query image, we are able to localize the common object in the query image without the need for a single box annotation. We only know that there is a common object, we do not know *where* and *what* the common object is.

We investigate this task in an attempt to alleviate the double burden of annotation in visual localization tasks, namely regarding the number of examples and regarding the box annotations for each example. This task is therefore on the intersection between few-shot learning [43, 48] and weakly-supervised detection [4, 46]. We also envision a number of applications that arise from this task. First, few-shot common-localization enables us to search spatially for specific instances in large and complex scenes. Second, we can use this task to enhance other learning tasks. Few-shot common localization can for example be used as a quick annotation tool or as a form of prior annotation for tasks such as active learning [14]. Third, this approach enables easier search in tasks such as remote sensing [32, 23].

At the core of few-shot common-localization is getting the most out of the limited information. To that end, we propose a new deep network made for this task, shown in Figure 1. First, the support and query images are fed into a backbone network to obtain spatial features. Second, we discover what is common among the support and query images. We propose a spatial similarity module that learns the spatial regions of commonality through non-local operations. Third, we hypothesize that support images are not equally important and propose a feature reweighting module. This module employs graph convolutional networks to balance the support images. Fourth, we use the spatial and weight information with a class-agnostic localization network to localize the object in the query image.

To experiment on few-shot common-localization, we repurpose and reorganize the well-known Pascal VOC 2007, Pascal VOC 2012, MS-COCO, and ImageNet VID datasets. Experimental results show the importance of spatial similarity and feature reweighting for few-shot common-localization. This results in a system that outperforms baselines from related tasks such as object detection and few-shot learning. The setup and method serve as a catalyst for future work in this task and are all publicly available along with the code of our networks and modules at http://taohu.me/SILCO/.

2. Related work

Object detection. Modern object detectors can be categorized into two categories: one-stage and two-stage detectors. One-stage detectors such as YOLO [38] and SSD [31] directly use the backbone architecture for object instance detection. Two-stage detectors such as Faster R-CNN [39] and FPN [29] first propose many possible object locations and use a sub-network for determining and regressing the best proposals. In this work, we rely on basic components such as SSD [31]. Where standard object detection requires many examples and dense annotations, we utilize such networks to deal with few examples and no box annotations. Weakly-supervised object detection [4, 7, 9, 41, 46] has recently been investigated for the scenario where many examples are given, but these examples are not annotated with boxes. Compared with our method, both approaches do not require bounding box annotations. Our method localizes arbitrary objects from a few support images only, while weakly-supervised localization requires many examples per class from a pre-defined vocabulary [17].

Object co-detection. More closely connected to the task of few-shot common-localization is object co-detection [3, 18, 20]. Given two images with the same object, the goal of co-detection is to localize the common instance in both two images. This task differs from few-shot common-localization in two aspects. First, co-detection can only handle the scenario with two input images, while we can handle more inputs. Second, our task evaluates few-shots from previously unseen classes, while co-detection uses the same classes for training and evaluation.

Few shot learning. A central task in few-shot learning is global classification [12, 28, 43, 48, 53, 51]. Approaches such as deep siamese networks [28], matching networks [48], and prototypical networks [43] aim to solve this task by learning embedding spaces. The work of Garcia *et al.* [12] leverages graph convolutional networks [27] for few-shot, semi-supervised, and active learning. We are inspired by the success of graph convolutional networks in few-shot settings and incorporate them in the context of common-localization from few examples.

A number of works have investigated few-shot learning beyond classification [6, 10, 40, 24, 34, 35, 37, 25, 26]. HU *et al.* [24] propose a model for image segmentation from few examples. While effective, this work requires dense pixel-wise annotations for the support images, same as [40]. In this work, we relax this constraint by localization without any spatial annotations. Dong *et al.* [10] study object detection using a large pool of unlabeled images and only a few labeled images per category. Pseudo-labels for the unlabeled images are utilized to iteratively refine the detection result. Akin to HU *et al.* [24], Dong *et al.* rely on spatial annotations for the support examples, while we do not utilize any box annotations for our few examples. Chen *et al.* [6] construct a target-domain detector from few target training annotations by leveraging rich source-domain knowledge. Different from their work, our method tries to solve this problem by utilizing weak prior information of common object existence.

Recently, object detection and segmentation have been investigated from a zero-shot perspective [2, 13, 50]. While promising, the results are not yet at the level of supervised tasks, hence we do not compare to zero-shot approaches.

3. Method

3.1. Problem formulation

For our task of few-shot common-localization, the goal is to learn a model $f(S_c^N, Q_c)$ that, when given a support image set S_c^N of N images and query image Q_c, predicts bounding boxes for class c. The function $f(\cdot)$ is parameterized by a deep network containing a support branch and a query branch. During training, the algorithm has access to a set of image tuples $T = (S_c^N, Q_c)$, where $c \in L_{train}$. At testing, we focus on new (unseen) semantic classes, i.e. $c \in L_{test}$ and $L_{train} \cap L_{test} = \emptyset$.

3.2. SILCO network

For the problem of few-shot common-localization, we propose the SILCO (**S**how a Few **I**mages, **L**ocalize the **C**ommon **O**bject) network. An overview of our approach is shown in Figure 1. Our framework starts from the Single Shot Detector (SSD) architecture [31], using VGG [42] as

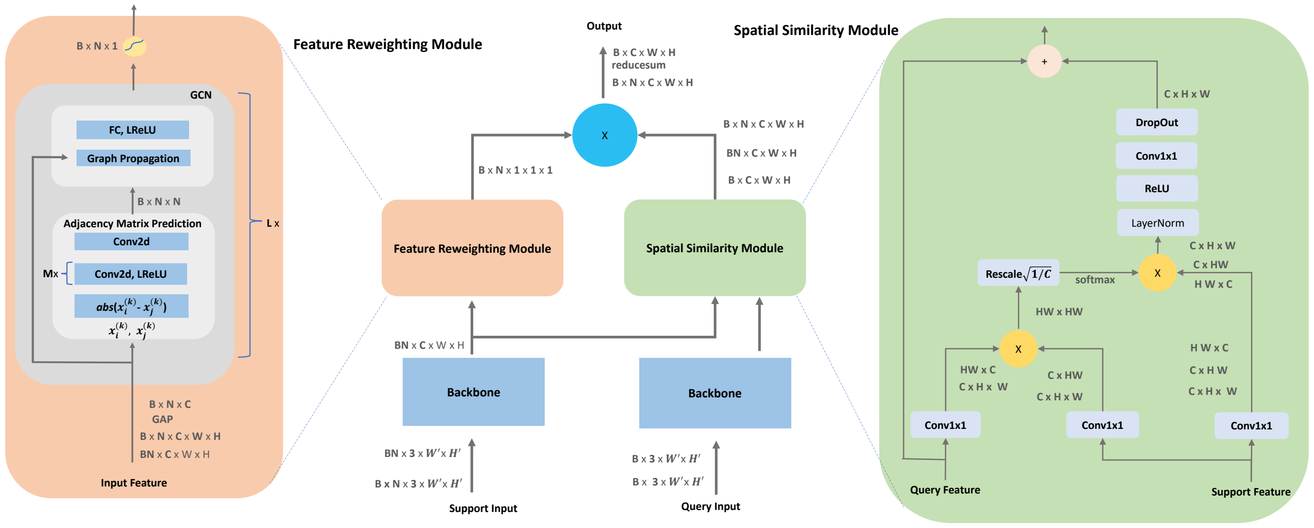

Figure 2. Overview of the spatial similarity module, feature reweighting module, and the aggregation. The feature maps are shown as the shape of their tensors . B, N, C denotes batch size, number of support images, and number of channels respectively. The image size is $W^{'} \times H^{'}$, the size after the backbone is $W \times H$. GAP denotes Global Average Pooling. L denotes the number of GCN blocks. M denotes the number of Conv2d-LReLU combinations. Graph Propagation means multiplication between vertex feature and graph adjacency matrix.

our backbone. Motivated by multi-scale fusion [29], different scales of features are used to deal with different scales of bounding boxes. The SILCO network facilitates the query image to help the few shot co-localization based for the common class given the weakly-supervised support images at different scales, the key function is to try to integrate support feature and query feature, *i.e.*:

$$\tilde{q}_i = \phi(q_i, S_i), \tag{1}$$

for query feature q_i and support feature S_i at scale i. In total, akin to SSD [31], our network contains five scales. Output $\tilde{q}_i \in \mathbb{R}^{B \times C \times W \times H}$ denotes the result of combining the query and support branches. Multiple scales are utilized to help the support branch guide the query branch. The final prediction of SILCO Network is as follows:

$$f(q, S) = \text{DET}(\text{CONCAT}_{i \in \mathcal{S}}(\tilde{q}_i)) \tag{2}$$

where CONCAT means concatenation along channel axis, DET is the final detection module used for classification and localization, and $\mathcal{S}$ denotes the set of scales.

There are three choices for function ϕ in our network. We first present a basic way to perform few-shot common-localization with this network. Then we introduce two modules to best leverage the few weakly-supervised support images for common-localization.

3.2.1 A basic version: Global Average Pooling

The common object may exist in different zones in every support image. Therefore, a starting point in the SILCO network is to only consider the channel support and remove spatial information, *i.e.*:

$$\phi(q_i, S_i) = q_i + \frac{1}{N}\sum_{j=1}^{N} \text{GAP}(S_i^{(j)}) \tag{3}$$

where GAP denotes global average pooling to remove spatial information, enabling us to directly obtain a representation for localization. Auto Broadcasting is conducted when shape is different. However, this setup does not fully leverage the few support examples we have been given. Therefore, we introduce two new modules.

3.2.2 Spatial Similarity Module

Building upon our starting network and inspired by recent success of the Transformer structure in language processing [47] and non-local blocks [49], we have designed a spatial similarity module, depicted on the right of Figure 2. The main goal of this module is to search for spatial support between the support images and the query image.

The inputs of the spatial similarity module are the query and support features. The outputs are spatially enhanced query features. We investigate two ways to perform spatial similarity. For the first one, the spatial similarity calculates the inner product of the features from support and query branch first, after which softmax is applied to formulate a pixel-wise attention matrix. This matrix then is multiplied with the support features in order to enforce a spatial similarity search between support image and query image. The overall process of this spatial similarity is formulated as:

$$\begin{aligned}\mathrm{SSM}^{j}_{im}(q_i, S_i) =& c_1(soft(c_2(q_i)^T \times c_3(S_i^{(j)})) \\ & \times c_4(S_i^{(j)})) + q_i,\end{aligned} \tag{4}$$

where c_1, c_2, c_3, c_4 are convolutional layers, $soft$ means softmax activation, and $\times$ denotes matrix multiplication, $\mathrm{SSM}^{j}_{im}(q_i, S_i)$ denotes the spatial similarity between query image q_i and j-th support image $S_i^{(j)}$ at scale i. For simplicity, some normalization operators such as Dropout [44], Rescaling, Layer Normalization [1] are ignored here. More details of the spatial similarity can be observed in Figure 2. For the final spatial similarity, we can simply take the average over the enhanced features from all support images:

$$\mathrm{SSM}_{im}(q_i, S_i) = \frac{1}{N}\sum_{j=1}^{N} \mathrm{SSM}^{j}_{im}(q_i, S_i), \tag{5}$$

where N is the number of support images.

The first image-wise spatial similarity is performed for each support image separately. Another choice of spatial similarity calculation is to consider **all** support images at once (dubbed "global spatial similarity" throughout this work). The global spatial similarity is given as:

$$\begin{aligned}\mathrm{SSM}_{g}(q_i, S_i) =& c_1(soft(c_2(q_i)^T \times c_3(S_c)_i) \\ & \times c_4(S_c)_i) + q_i,\end{aligned} \tag{6}$$

where S_c is the concatenation of all support features. Global spatial similarity considers all support features at once and tries to search for the spatial similarity accordingly. The final formulation of ϕ is given as:

$$\phi(q_i, S_i) = \mathrm{SSM}_{im/g}(q_i, S_i), \tag{7}$$

where $\mathrm{SSM}_{im/g}(q_i, S_i)$ denotes the choice for image-wise or global spatial similarity.

3.2.3 Feature Reweighting Module

The spatial similarity module incorporates spatial commonality between support and query images. It assumes that each support image is equally informative for common-localization. Here, we propose a feature reweighting module that reweights the influence of examples in the support branch by interpreting the few-shot images as a connected graph. The weights of this graph are learned through graph convolutional networks (GCNs). The overall structure of the feature reweighting module is demonstrated in Figure 2.

The input of the module are the features of the support images, the output is the weight of each support image. The structure of the module is formulated by a GCN. First we detail how to calculate the weight per support example, then we detail how to conduct the feature reweighting.

Support weights. A GCN is typically fed with an input signal $x \in \mathbb{R}^{N\times d}$ on the vertices of a weighted graph G. We consider an operator family $\mathcal{A}$ of graph intrinsic linear operators that act locally on this signal. The simplest is the *adjacency operator* A. Motivated by ResNet [21], the identity operator is also applied as a form of skip connection in long-ranges. Therefore, we opt for the the operator family $\mathcal{A} = \{A, \mathbf{1}\}$ in our work. A GCN receives a feature input $x^{(k)} \in \mathbb{R}^{N\times d_k}$ and produces $x^{(k+1)} \in \mathbb{R}^{N\times d_{k+1}}$, which can be formulated as:

$$x_l^{(k+1)} = gcn(\cdot) = \rho(\sum_{F\in\mathcal{A}} Fx^{(k)}\theta_{F,l}^{(k)}), l = d_1, ..., d_{k+1} \tag{8}$$

where $\Theta = \theta_1^{(k)}, ..., \theta_{|\mathcal{A}|_k}^{(k)}, \theta_{F,l}^{(k)} \in \mathbb{R}^{d_k\times d_{k+1}}$, are trainable parameters and $\rho(\cdot)$ is a point-wise non-linearity, LReLU [52] in our work. Furthermore, the graph adjacency matrix in *adjacency operator* can also be learned from the current node hidden representation [15]:

$$\tilde{A}_{i,j}^{(k)} = \varphi_{\tilde{\theta}}(|\boldsymbol{x}_i^{(k)} - \boldsymbol{x}_j^{(k)}|), \tag{9}$$

where φ is a symmetric function that can be parameterized by a neural network, the neural network is stacked after the absolute difference between two vector nodes. To obtain the feature weight, Eq. 8 will be cascaded for L times to capture the long-range connection in the graph. In the end, inspired by SENet [22], a sigmoid layer is appended to generate final weight. The detail can be formulated as:

$$\mathrm{FRM}(S) = \sigma(gcn(\cdots gcn(S))), \tag{10}$$

where σ is a sigmoid layer, and the output represents feature weight for every support image $\mathrm{FRM}(S) \in \mathbb{R}^{B\times N}$.

Feature Reweighting. To combine the features from the spatial similarity module and the weights from the feature reweighting module, we multiply them both the feature image-wise. In the end, by utilizing the Equation 4, ϕ can be further formulated as:

$$\phi(q_i, S_i) = \mathrm{RS}(\mathrm{CONCAT}_{j=1}^{N}(\mathrm{SSM}^{j}_{im}(q_i, S_i))\otimes \mathrm{FRM}(S_i)), \tag{11}$$

where $\mathrm{SSM}^{j}_{im}(q_i, S_i))$ is spatial similarity between query image q_i and j-th support image S_i at scale i, FRM is feature reweighting module, $\otimes$ is hadamard product(broadcasting is ignored if shape mismatches), CONCAT is the concatenation operation, which is a mapping from $\mathbb{R}^{B\times C\times W\times H}$ to $\mathbb{R}^{B\times N\times C\times W\times H}$. The final RS denotes the reduce_sum operation that eliminates the second dimension and leads to $\mathbb{R}^{B\times C\times W\times H}$.

3.2.4 Optimization

Similar to the framework of SSD, our loss function is also composed of a bounding box regression loss and a cross entropy classification loss. The difference is that our classification is class-agnostic, it depends on the common class of the support images and query image.

$$L(x,c,l,g) = \frac{1}{BD}\sum_{i,j=1}^{B,D}(\boldsymbol{bce}(c_{ij},x_{ij}) + \ell_1^s(l_{ij},g_{ij})), \tag{12}$$

where B is the batch size, D is the number of matched default boxes, ***bce*** means binary classification entropy loss function, ℓ_1^s denotes smoothed ℓ_1 norm loss function [16]. c_{ij},x_{ij}, l_{ij} , g_{ij} are the class probability, class ground truth, predicted coordinate, ground truth coordinate of i-th image, j-th bounding box proposal, respectively.

4. Experimental setup

4.1. Common-localization datasets

To accompany the new task of few-shot common-localization, we have prepared a revised setup for three well-known datasets intended for object detection, namely Pascal VOC [11], MS-COCO [30], and ImageNet VID [8].

CL-VOC. We divide the 20 classes of PASCAL VOC into two disjoint groups, one group is used for training, the other for validation/testing. We use both groups for both tasks and report the mean performance of the two runs. We perform experiments both on Pascal VOC 2007 and 2012, dubbed CL-VOC-07 and CL-VOC-12 respectively. The training set D_{train} is composed of all image pairs from the PASCAL VOC training set that include one common class from the label-set L_{train}. The validation set D_{val} and test set D_{test} are both from the PASCAL VOC validation set. For a detailed explanation of our dataset organization procedure, please refer to the supplementary materials.

CL-COCO. We furthermore recompile a common-localization dataset based on the MS-COCO 2014 dataset [30]. The 80 classes in MS-COCO are divided into two disjoint groups. The classes in each group are provided in the supplementary materials.

CL-VID. To evaluate a generalization to videos, we employ the ImageNet VID dataset [8], a benchmark for video object detection. We use the 3,862 video snippets from the training set for evaluation, which includes 30 objects. We employ this dataset to evaluate our approach on open-set (*i.e.*, unseen) classes. We train our model on CL-VOC-12. There are some overlapping classes between Pascal VOC and ImageNet VID. We keep videos which have one target class and no overlap with any Pascal VOC class. For details on the retained classes, please refer to the supplementary materials. The support images are selected from ImageNet DET [8] for evaluation. Each frame of a test video acts as query image.

Table 1. **Spatial similarity module.** Mean average precision (%) for image-wise versus global spatial similarity on CL-VOC-12. For both groups, image-wise similarity works better and we will use this form of spatial similarity for further experiments.

	Group 1	Group 2	mean
Global	51.71	55.49	53.60
Image-wise	**54.04**	**57.39**	**55.71**

4.2. Implementation details

We use PyTorch [36] for implementation. The network is trained with SGD [5] with a learning rate of 1e-4 and momentum of 0.99 on one Nvidia GTX 1080TI. The weights of the support and query branch are pre-trained on ImageNet [8]. All the images in the support and query branch are resized to 300 × 300 and the batch size is set to 6. For the query branch we choose photo-metric distortion, random mirror, random sample crop, akin to SSD [31].

4.3. Evaluation

For the training tuples, we randomly sample tuples $T = (S_c^N, Q_c)$, such that all tuples contain the common classes $c \in L_{train}$. For evaluation, we randomly sample several tuples $T = (S_c^N, Q_c)$, which contain the common class $c \in L_{test}$. We evaluate on 5000 tuples in CL-VOC and 10000 tuples in CL-COCO. Our training, validation, and test images are always disjunct. The object classes in training are disjunct from those in validation/test. The hyperparameter search is done once on Group 1 of CL-VOC-12. We use the same hyperparameters for all experiments on CL-VOC-07, CL-VOC-12, CL-COCO, and CL-VID. On the respective validation sets we choose the best model.

We employ the (mean) Average Precision as evaluation measure throughout our experiments. The overall mAP is averaged on the mAPs of the two groups and computed using the setup of [11]. For evaluation we only consider the top 200 detected bounding boxes, and rank these boxes according to their objectness score. Each prediction that overlaps with the closest ground truth with a value of at least 0.5 will be regarded as a positive detection. After that, a non-maximum suppression with a threshold of 0.45 is applied.

5. Experimental results

5.1. Ablation study

Spatial similarity module. In the spatial similarity module, there are two ways to relate features from the support and query branches. The first, image-wise spatial similarity, computes a matrix of size $HW \times HW$ for each support image. The second, global spatial similarity, computes a single matrix of size $HW \times NHW$, which regards all N support images as a whole to the spatial similarity. We compare the two different forms of similarities in Ta-

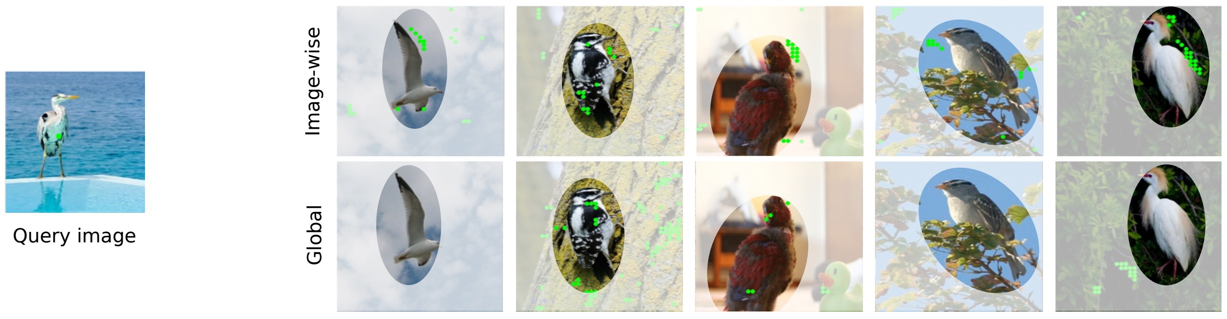

Figure 3. **Spatial similarity module** visualization. Two examples are demonstrated, the left is the query image, the top, bottom images are image-wise similarity visualization and global similarity visualization respectively. For image-wise similarity, the top 20 activations are visualized per image. For global similarity, the top 100 activations are visualized in all 5 images. The green dot in the query image is the reference point. The green dots in the support images are calculated based on the reference point in the query image. Best viewed in color.

Table 2. **Ablation of spatial similarity and feature reweighting.** The metric is mean average precision (%). As we adopt spatial similarity (SSM) and feature reweighting (FRM) the accuracy gradually increases over a simple global average pooling (GAP), indicating the effectiveness of our proposed modules.

dataset	GAP	SSM	FRM	Group 1	Group 2	mean
CL-VOC-07	✓			55.17	52.18	53.67
		✓		56.12	55.52	55.82
		✓	✓	**57.17**	**56.45**	**56.82**
CL-VOC-12	✓			53.55	54.61	54.08
		✓		54.04	57.39	55.71
		✓	✓	**55.11**	**58.62**	**56.86**
CL-COCO	✓			13.37	6.50	9.94
		✓		18.50	7.70	13.10
		✓	✓	**18.62**	**8.20**	**13.40**

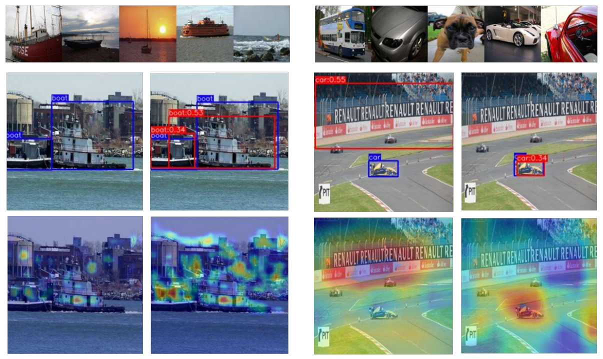

Figure 4. **Feature reweighting** heatmap visualization (blue means low, red means high). The first row shows the support images, the second row shows the query image with ground truth (blue box) and prediction (red box), the bottom row shows the heatmap visualization. The heatmaps are the normalized feature map selected from the output of our network. For both examples, the left and right columns show the results with and without feature reweighting. The heatmap results show that feature reweighting can better highlight the areas containing the common object. Label information is only used for illustration here, we don't utilize them in experiment.

ble 1. We observe that image-wise spatial similarity outperforms global spatial similarity. Our hypothesis is that image-wise spatial similarity more explicitly exploits the prior knowledge of the common-localization task that all support images are of the same class. To highlight this ability of image-wise spatial similarity, we visualize the top activation pixels in Figure 3. We find that image-wise spatial similarity balances the attention of every support image, while global spatial similarity exhibits a less uniform attention distribution. For the bird example, the global similarity misses the common object in the first and fourth support image, while many irrelevant areas in the second support images are targeted. Based on this study, image-wise spatial similarity will be adopted for the rest of the experiments.

Feature reweighting module. We also explore the effect of feature reweighting based on the previous results. The ablation result is indicated in Table 2. We first observe that spatial similarity outperforms the global average pooling baseline, further validating its effectiveness. Across all three datasets, adding feature reweighting on top of the spatial similarity benefits the common-localization accuracy. To better understand the inner mechanism of feature reweighting, we visualize the feature heatmaps before and after feature reweighting in Figure 4. The figure shows that the reweighted features better focus on the common class to further enhance the common-localization.

Effect of support images. Our common-localization is optimized to work with few examples as support. To show this capability, we have explored the effect of gradually increasing the number of support images in Figure 5. We have evaluated with 3, 5, 7, and 9 support images. The results show that our approach obtains high accuracy with only a few support images. As the number of support images increases, the gap of our approach with and without spatial similarity and feature reweighting gradually becomes larger, which indicates that our modules can capture the

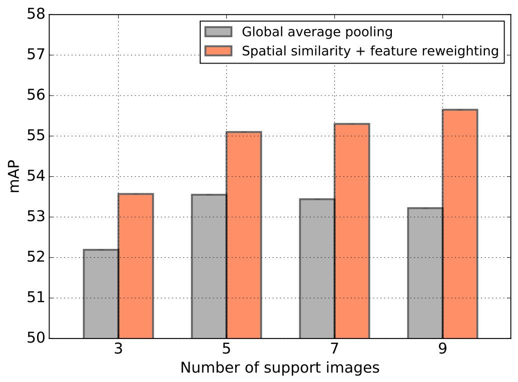

Figure 5. **Effect of support images** on CL-VOC-12. For better visualization results, our y-axis starts from 50% mAP. Compared to a global average pooling (GAP) baseline, our spatial similarity (SSM) and feature reweighting modules (FRM) already works well with little support, and our modules capture the common object even better when support increases.

Table 3. **Effect of object size** on CL-VOC-12 in mAP(%). Compared to a global average pooling (GAP) baseline, our spatial similarity (SSM) and feature reweighting modules (FRM) especially improves for medium-sized objects.

method	small	medium	large
GAP	9.60	10.87	28.64
SSM, FRM	**10.99**	**13.85**	**29.24**

common object even better when the support set grows. We have also investigated the ability to localize more than one common object and we show qualitative examples in the supplementary materials.

Support image corruption. Our approach even works when some of the support images do not contain the common object. We did an experiment for the 5-shot setting, where we insert a corrupted support image. On CL-VOC-12 the mAP drops from 56.86 to 56.53 and on CL-COCO from 13.40 to 13.03.

Effect of object size. We explore the effect of different object sizes on CL-VOC-12 in Table 3. The small, medium, large object are defined as area ratio per image ranging from [0, 0.15],[0.15, 0.3],[0.3, 1]. We observe most gain for medium-sized objects, while we observe a gain for all settings. Localizing large objects may be easier, so the gain is modest, explaining their relatively modest improvement.

Success and failure case analysis. Figure 6 shows that our method can perform common-localization in complex query images, which contain multiple objects. The right example of row two shows that our method even works well when multiple instances exist in a single query image. We also observe several failure cases: 1). Saliency. The most salient object is often mistaken as the true positive. 2). Object size. Our method fails to localize the object that is extremely small. 3). Context information. Our method doesn't consider the context information, for example in this case, that a chair is unlikely to be on a table. 4). Instance concept. Our method may fail if there exists no clear boundary between instances. These all provide interesting avenues for future enhancement of common-localization.

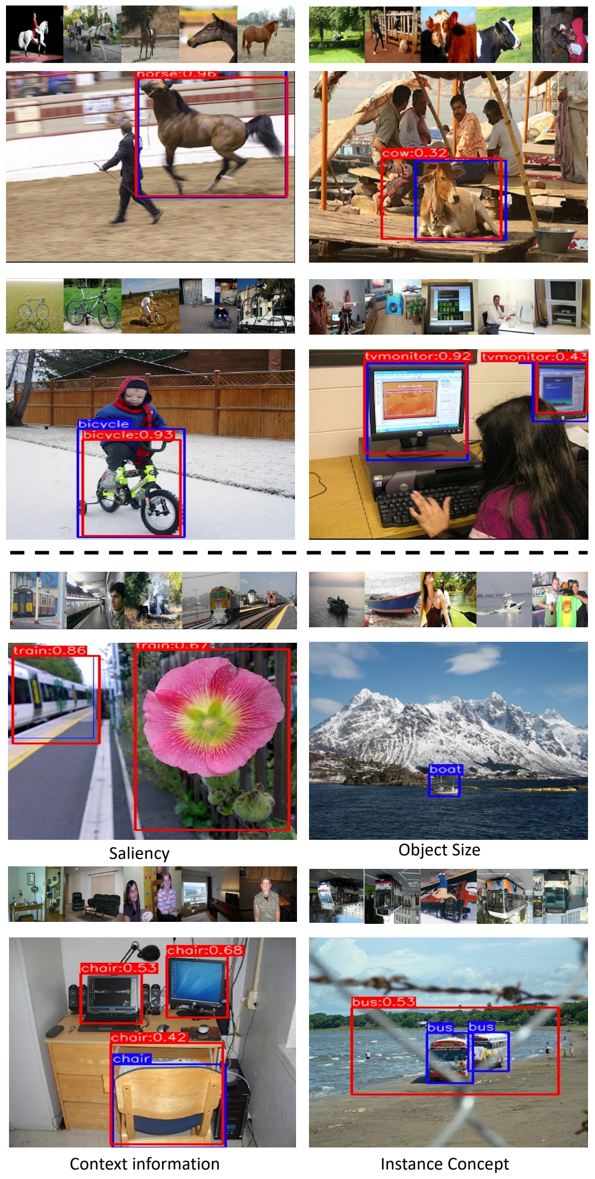

Figure 6. **Success and failure case analysis** on CL-VOC-12. Top two rows: four success cases. Bottom two rows: four common failure cases, where confusion is caused by saliency, object size, lack of context, or instance confusion. For each case, we present 5 support images in the top and the query image in the bottom. Blue indicates ground truth, red means prediction. Label information is only used for illustration here, we don't utilize them in experiment. More cases can be seen in `http://taohu.me/SILCO/`.

Table 4. **Comparative evaluation** on CL-VOC-07, CL-VOC-12, and CL-COCO. Across all datasets, our approach outperforms the center box and Region Proposal Network (RPN) [39] baselines by a large margin. Furthermore, our Spatial Similarity (SSM) and Feature Reweighting (FRM) modules are preferred over the ConvLSTM of HU *et al*. [24] for few-shot common-localization.

	CL-VOC-07			**CL-VOC-12**			**CL-COCO**			**CL-VID**
	group 1	group 2	mean	group 1	group 2	mean	group 1	group 2	mean	mean
Center box	5.28	4.05	4.67	6.56	6.90	6.73	0.54	0.51	0.53	-
RPN [39]	18.72	15.13	16.93	20.23	18.54	19.39	3.20	1.93	2.57	-
Contrastive RPN [19]	18.89	15.33	17.21	19.21	18.53	18.87	3.54	2.01	2.77	-
This paper										
w/ ConvLSTM [24]	53.46	54.90	54.18	51.34	57.42	54.38	16.37	6.72	11.54	53.02
w/ SSM and FRM	**57.17**	**56.10**	**56.64**	**55.12**	**58.62**	**56.87**	**18.62**	**7.47**	**13.04**	**54.40**

5.2. Comparative evaluation

Baselines. We first compare to baselines using a fixed center box or a Region Proposal Network(RPN) [39]. The center box baseline serves as a sanity check to understand the complexity of the few-shot common-localization task. The RPN serves as a state-of-the-art comparison to standard object detection. For the center box baseline, we simply select the center box of the query image as the final object proposal. The optimal size of the center box is determined through grid search per dataset. For the RPN, we first train a class-agnostic RPN, after which we extract both the ROI scores and features from the query image and support images. Second, we generate candidate support features by choosing the ROI with the highest score per support image. Third, we match candidate support features and the query ROI features according to L2 distance. The query ROI with the lowest feature distance is used as the final proposal.

We also extend the RPN baseline by adding a Siamese Network constrained by a contrastive loss [19] to learn a discriminative distance metric. To obtain pairs during training, we sample ground truth boxes with a 1:1 ratio. A distance margin of 0.5 is chosen by cross-validation. The remaining process is the same as the RPN baseline.

To evaluate the spatial similarity and feature reweighting modules, we extend our base approach with the ConvLSTM of HU *et al*. [24], previously used for few-shot image segmentation. ConvLSTM is adopted between support and query features. Because this baseline is GPU-inefficient, we scale down the number of channels to half the original size through a 1×1 convolution. After the ConvLSTM fusion, we scale up the channel number to the original number by another 1×1 convolution.

Results. The results are shown in Table 4. The center box baseline scores lowest across all datasets, indicating that this task is not easy to solve. For the RPN baselines, we obtain a stable gain compared to the center box baseline, illustrating that deep features and their similarity have an important role in our task. Our method, either with ConvLSTM [24] or the proposed spatial similarity and feature reweighting modules outperforms the center box and RPN baseline, which shows that our common-localization structure is more suitable for the few-shot common-localization task. Furthermore, our method with the combination of spatial similarity and feature reweighting works best compared to all other methods in three datasets, which indicates the spatial similarity and feature reweighting modules can function well in a mutually beneficial way.

Time and complexity analysis. SILCO has 37.1M parameters and an inference time of 0.12 seconds with 5 shots. For comparison, ConvLSTM [24] has 56.3M parameters, while the inference time is a bit faster with 0.09 seconds.

5.3. Video common-localization

To validate generalization abilities, we also evaluate on video detection dataset ImageNet VID [8]. We use randomly selected images from ImageNet DET as support images, and every frame of a video from VID as query image. The results in Table 4 again confirm our method obtains consistent gains by incorporating spatial similarity and feature reweighting.

6. Conclusion

This paper introduces the task of few-shot common-localization. From a few support images without box annotations, we aim to localize the object in the query image that is common among all images. To that end, we introduce a network specific to this problem and propose two modules to improve the common-localization. The first module enhances the spatial similarity among the support and query images. The second module balances the influence of each support image and reweights the features from the spatial similarity accordingly. Experiments show that our approach can robustly localize the common objects from few examples, outperforming baselines from related fields. We see this work as a first step into localized learning from double-weak supervision, where examples are both scarce and without box annotations.

Acknowledgments. The authors thank Zeynep Akata for useful discussions and the anonymous reviewers for their insightful suggestions.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv*, 2016. 4

[2] Ankan Bansal, Karan Sikka, Gaurav Sharma, Rama Chellappa, and Ajay Divakaran. Zero-shot object detection. In *ECCV*, 2018. 2

[3] Sid Yingze Bao, Yu Xiang, and Silvio Savarese. Object co-detection. In *ECCV*, 2012. 2

[4] Hakan Bilen and Andrea Vedaldi. Weakly supervised deep detection networks. In *CVPR*, 2016. 1, 2

[5] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*. 2010. 5

[6] Hao Chen, Yali Wang, Guoyou Wang, and Yu Qiao. LSTD: A low-shot transfer detector for object detection. *AAAI*, 2018. 2

[7] Ramazan Gokberk Cinbis, Jakob Verbeek, and Cordelia Schmid. Weakly supervised object localization with multi-fold multiple instance learning. *PAMI*, 2017. 2

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009. 1, 5, 8

[9] Ali Diba, Vivek Sharma, Ali Mohammad Pazandeh, Hamed Pirsiavash, and Luc Van Gool. Weakly supervised cascaded convolutional networks. In *CVPR*, 2017. 2

[10] Xuanyi Dong, Liang Zheng, Fan Ma, Yi Yang, and Deyu Meng. Few-example object detection with model communication. *PAMI*, 2018. 2

[11] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (VOC) challenge. *IJCV*, 2010. 5

[12] Victor Garcia and Joan Bruna. Few-shot learning with graph neural networks. *ICLR*, 2018. 2

[13] Kirill Gavrilyuk, Amir Ghodrati, Zhenyang Li, and Cees G. M. Snoek. Actor and action video segmentation from a sentence. In *CVPR*, 2018. 2

[14] Efstratios Gavves, Thomas Mensink, Tatiana Tommasi, Cees G. M. Snoek, and Tinne Tuytelaars. Active transfer learning with zero-shot priors: Reusing past datasets for future tasks. In *ICCV*, 2015. 1

[15] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. *arXiv*, 2017. 4

[16] Ross Girshick. Fast R-CNN. In *ICCV*, 2015. 5

[17] Ramazan Gokberk Cinbis, Jakob Verbeek, and Cordelia Schmid. Multi-fold mil training for weakly supervised object localization. In *CVPR*, 2014. 2

[18] Xin Guo, Dong Liu, Brendan Jou, Mojun Zhu, Anni Cai, and Shih-Fu Chang. Robust object co-detection. In *CVPR*, 2013. 2

[19] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *CVPR*, 2006. 8

[20] Zeeshan Hayder, Mathieu Salzmann, and Xuming He. Object co-detection via efficient inference in a fully-connected crf. In *ECCV*, 2014. 2

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 4

[22] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *CVPR*, 2018. 4, 11

[23] Tao Hu, Yao Wang, Peng Lu, and Heng Wang. Sobel heuristic kernel for aerial semantic segmentation. 2018. 1

[24] Tao Hu, Pengwan Yang, Chiliang Zhang, Gang Yu, Yadong Mu, and Cees G. M. Snoek. Attention-based multi-context guiding for few-shot semantic segmentation. In *AAAI*, 2019. 2, 8

[25] Bingyi Kang, Zhuang Liu, Xin Wang, Fisher Yu, Jiashi Feng, and Trevor Darrell. Few-shot object detection via feature reweighting. *arXiv*, 2018. 2

[26] Leonid Karlinsky, Joseph Shtok, Sivan Harary, Eli Schwartz, Amit Aides, Rogerio Feris, Raja Giryes, and Alex M Bronstein. RepMet: Representative-based metric learning for classification and few-shot object detection. In *CVPR*, 2019. 2

[27] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv*, 2016. 2

[28] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML Workshop*, 2015. 2

[29] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 2, 3

[30] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 5

[31] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: Single shot multibox detector. In *ECCV*, 2016. 1, 2, 3, 5

[32] Emmanuel Maggiori, Yuliya Tarabalka, Guillaume Charpiat, and Pierre Alliez. Can semantic labeling methods generalize to any city? the inria aerial image labeling benchmark. In *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, 2017. 1

[33] Gary Marcus. Deep learning: A critical appraisal. *arXiv*, 2018. 1

[34] Claudio Michaelis, Matthias Bethge, and Alexander S. Ecker. One-shot segmentation in clutter. In *ICML*, 2018. 2

[35] Claudio Michaelis, Ivan Ustyuzhaninov, Matthias Bethge, and Alexander S. Ecker. One-shot instance segmentation. *arXiv*, 2018. 2

[36] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 5

[37] Kate Rakelly, Evan Shelhamer, Trevor Darrell, Alexei A. Efros, and Sergey Levine. Few-shot segmentation propagation with guided networks. *arXiv*, 2018. 2

[38] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016. 1, 2

[39] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 1, 2, 8

[40] Amirreza Shaban, Shray Bansal, Zhen Liu, Irfan Essa, and Byron Boots. One-shot learning for semantic segmentation. In *BMVC*, 2017. 2

[41] Miaojing Shi, Holger Caesar, and Vittorio Ferrari. Weakly supervised object localization using things and stuff transfer. In *ICCV*, 2017. 2

[42] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *NeurIPS*, 2014. 2

[43] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *NeurIPS*, 2017. 1, 2

[44] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*, 2014. 4

[45] Meng Tang, Federico Perazzi, Abdelaziz Djelouah, Ismail Ben Ayed, Christopher Schroers, and Yuri Boykov. On regularized losses for weakly-supervised cnn segmentation. In *ECCV*, 2018. 1

[46] Peng Tang, Xinggang Wang, Angtian Wang, Yongluan Yan, Wenyu Liu, Junzhou Huang, and Alan Yuille. Weakly supervised region proposal network and object detection. In *ECCV*, 2018. 1, 2

[47] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 3

[48] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *NeurIPS*, 2016. 1, 2

[49] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018. 3

[50] Yongqin Xian, Subhabrata Choudhury, Yang He, Bernt Schiele, and Zeynep Akata. Semantic projection network for zero-and few-label semantic segmentation. In *CVPR*, 2019. 2

[51] Yongqin Xian, Saurabh Sharma, Bernt Schiele, and Zeynep Akata. f-VAEGAN-D2: A feature generating framework for any-shot learning. In *CVPR*, 2019. 2

[52] Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li. Empirical evaluation of rectified activations in convolutional network. *arXiv*, 2015. 4

[53] Flood Sung Yongxin Yang, Li Zhang, Tao Xiang, Philip H. S. Torr, and Timothy M. Hospedales. Learning to compare: Relation network for few-shot learning. In *CVPR*, 2018. 2

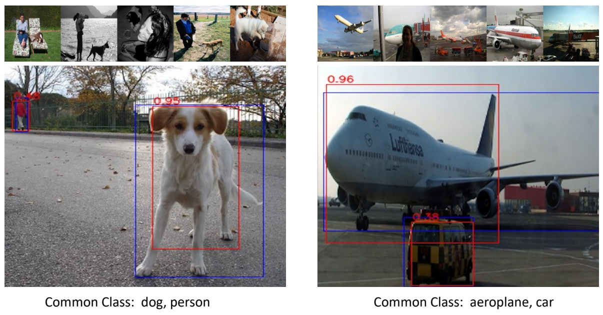

Figure 7. Localization from support images with multiple objects . Blue denotes ground truth, red denotes prediction. The two objects common in all support images are localized correctly in the query images.

Table 5. The #image of CL-VOC 07, CL-VOC 12, CL-COCO Few-Shot Common Localization Dataset.

dataset	train	val	test
CL-VOC07	2501	1010	1500
CL-VOC12	5717	2623	3200
CL-COCO	62783	20000	40504

Table 6. The comparison between "before channel attention" and "after channel attention". A typical channel attention method SENet[22] is deployed in our experiment.

dataset	method	Group 1	Group 2	mAP(%)
CL-VOC07	before	57.17	56.45	56.82
	after	**57.50**	**56.96**	**57.23**
CL-VOC12	before	55.11	58.62	56.86
	after	**56.55**	**59.22**	**57.89**

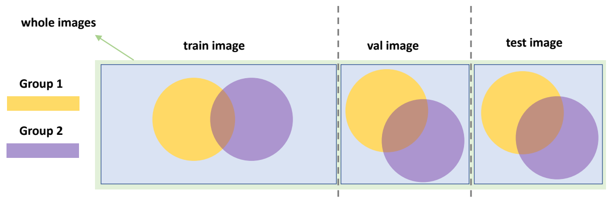

Figure 8. The dataset splitting demonstration. The total data are composed of train image, validation image, test image. Our labels are separate between different images.

Table 7. The class grouping for CL-VOC. One group is used to initialize the model, the other group is used to perform the few-shot common-localization.

Group	Semantic Classes
1	aeroplane, bicycle, bird, boat bottle, bus, car, cat, chair, cow
2	dining table, dog, horse, motorbike person, potted plant, sheep, sofa train, tv/monitor

SILCO Supplimentary File

Multiple common object. We also assess the ability to localize more than one common object. We have re-trained SILCO on images that also include multiple objects. Figure 7 shows two examples. In both cases, we detect the two objects that appear in all five support images. This further indicates the practical use of SILCO.

Dataset Details. we mainly elaborate the image numbers of CL-VOC07, CL-VOC12, CL-COCO in Table 5, the detail classes of all datasets is shown in Table 7, Table 8.

For experiment in VID, we only keeps the classes: bear, elephant, fox, giant panda, hamster, lion, lizard, monkey, rabbit, red panda, snake, squirrel, tiger, turtle, whale.

The data splitting principle of different subset is shown in Figure 8. For train,validation and test, the images are isolated with each other. When training, we random sample tuples(1+N for N-shot common-localization) from the pool.The detail algorithm is shown in Algorithm 1.

Another perspective to understand our method. From the aspect of space, our spatial similarity can be viewed as a kind of ***spatial*** attention mechanism between query feature and support feature. Furthermore, from the perspective of image, feature reweighting tries to rescale and balance the influence of different support ***images***. A natural question is whether we can introduce ***channel***-level attention into our framework to reach a ***spatial, channel, image***-level attention. Therefore, we try to adopt SENet[22] upon our framework and obtain the result in Table 6. It indicates that the performance can be further improved by channel attention. The result demonstrates that ***spatial, channel, image***-level attention mechanism can be combined in a mutually beneficial way.

Table 8. The class grouping for CL-COCO. One group is used to initialize the model, the other group is used to perform the few-shot common-localization.

Group	Semantic Classes
1	person, bicycle, car, motorcycle, airplane,bus, train, truck, boat, traffic light, fire hydrant, stop sign , parking meter, bench, bird, cat, dog, horse, sheep, cow , elephant, bear, zebra, giraffe, backpack, umbrella, handbag, tie , suitcase, frisbee, skis, snowboard, sports ball, kite, baseballbat, baseball glove, skateboard, surfboard, tennis racket bottle
2	wine glass, cup, fork, knife, spoon, bowl, banana, apple, sandwich, orange, broccoli, carrot, hot dog, pizza, donut, cake, chair, couch, potted plant, bed, dining table, toilet, tv, laptop, mouse, remote, keyboard, cell phone, microwave, oven, toaster, sink, refrigerator, book, clock, vase, scissors, teddy bear, hair drier, toothbrush

Algorithm 1 Dataset Preparation Algorithm

```
    Notations:
    $\mathcal{C}$: target class set
    $\mathcal{D}_i$ : the i-th image in the dataset $\mathcal{D}$
    $d$: dictionary for (label, image_list) pairs

for $\mathcal{D}_i$ in $\mathcal{D}$ do
    for $\mathcal{C}_j$ in $\mathcal{C}$ do
        if $\mathcal{D}_i$ has the class of $\mathcal{C}_j$ and
                    $d$ has key $\mathcal{C}_j$ then
            d [$\mathcal{C}_j$ ] .append( $\mathcal{D}_j$ )
        else
            d [$\mathcal{C}_j$ ] = [$\mathcal{D}_j$]
        end if
    end for
end for
while Not Stop do
    random choose 1 item $\tilde{C}$ from $\mathcal{C}$
    random choose k+1 items $\mathcal{R}$ from $d[\tilde{C}]$
    yield $\mathcal{R}$
end while
```