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 show 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.

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 re-purpose 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 (Show a Few Images, Locate the Common Object) 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
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 and support 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. Multiple scales are utilized to help the support branch guide the query branch. The final prediction of SILCO Network is as follows:

\[
\phi(q_i, S_i) = q_i + \frac{1}{N} \sum_{j=1}^{N} \text{GAP}(S_i^{(j)})
\]

where \(\text{GAP}\) 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.

#### 3.2.1 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.:

\[
f(q, S) = \text{DET} (\text{CONCAT}_{i \in S}(\tilde{q}_i))
\]

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

There are three choices for function \(\phi\) 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.:
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 $\mathcal{A} = \{A, 1\}$ 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:

$$x^{(k+1)}_l = gcn(\cdot) = \rho(\sum_{F \in \mathcal{A}} F x^{(k)} \theta_F^{(k)}), l = d_1, \ldots, d_k + 1 \tag{8}$$

where $\Theta = \theta_1^{(k)}, \ldots, \theta_{|\mathcal{A}|}^{(k)}, \theta_F^{(k)} \in \mathbb{R}^{d_k \times d_k + 1}$, 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]:

$$\tilde{A}^{(k)}_{i,j} = \varphi_{\theta}(|x_i^{(k)} - x_j^{(k)}|), \tag{9}$$

where $\varphi$ 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:

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

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

### 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, $\phi$ can be further formulated as:

$$\phi(q_i, S_i) = \text{RS}(\text{CONCAT}^N_{j=1}(SSM_j^{(im)}(q_i, S_i)) \odot \text{FRM}(S_i)) \tag{11}$$

where $SSM_j^{(im)}(q_i, S_i)$ is spatial similarity between query image $q_i$ and $j$-th support image $S_i$ at scale $i$, FRM is feature reweighting module, $\odot$ 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 $\mathbb{R}^{B \times C \times W \times H}$.

### 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.

\[
L(x, c, l, g) = \frac{1}{BD} \sum_{i,j=1}^{B,D} \left( bce(c_{ij}, x_{ij}) + \ell_s^a(l_{ij}, g_{ij}) \right),
\]

(12)

where B is the batch size, D is the number of matched default boxes, \( bce \) means binary classification entropy loss function, \( \ell_s^a \) denotes smoothed \( \ell_1 \) 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>
<thead>
<tr>
<th></th>
<th>Group 1</th>
<th>Group 2</th>
<th>mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global</td>
<td>51.71</td>
<td>55.49</td>
<td>53.60</td>
</tr>
<tr>
<td>Image-wise</td>
<td>54.04</td>
<td>57.39</td>
<td>55.71</td>
</tr>
</tbody>
</table>

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.

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 \( T = (S_c, Q_c) \), such that all tuples contain the common classes \( c \in L_{train} \). For evaluation, we randomly sample several tuples \( T = (S_i, Q_j) \), 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 disjoint. The object classes in training are disjoint 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.

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

Table 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
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.
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 $1 \times 1$ convolution. After the ConvLSTM fusion, we scale up the channel number to the original number by another $1 \times 1$ 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.


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


