# Accurate Few-shot Object Detection with Support-Query Mutual Guidance and Hybrid Loss

Lu Zhang[1] Shuigeng Zhou[1*] Jihong Guan[2] Ji Zhang[3]
[1]Shanghai Key Lab of Intelligent Information Processing, and School of Computer Science, Fudan University, China
[2]Department of Computer Science & Technology, Tongji University, China
[3]Zhejiang Laboratory, China
{l_zhang19,sgzhou}@fudan.edu.cn, jhguan@tongji.edu.cn, Ji.Zhang@zhejianglab.com

## Abstract

*Most object detection methods require huge amounts of annotated data and can detect only the categories that appear in the training set. However, in reality acquiring massive annotated training data is both expensive and time-consuming. In this paper, we propose a novel two-stage detector for accurate few-shot object detection. In the first stage, we employ a support-query mutual guidance mechanism to generate more support-relevant proposals. Concretely, on the one hand, a query-guided support weighting module is developed for aggregating different supports to generate the support feature. On the other hand, a support-guided query enhancement module is designed by dynamic kernels. In the second stage, we score and filter proposals via multi-level feature comparison between each proposal and the aggregated support feature based on a distance metric learnt by an effective hybrid loss, which makes the embedding space of distance metric more discriminative. Extensive experiments on benchmark datasets show that our method substantially outperforms the existing methods and lifts the SOTA of FSOD task to a higher level.*

## 1. Introduction

Object detection is one of the most fundamental problems in computer vision. Most existing object detection methods [23] require huge amounts of annotated training data, and it is hard to generalize a well-trained model to detect unseen classes. On the contrary, human beings have strong capability to recognize and locate an object that they have never seen before, but only if they have a few examples of such an object. In other words, the visual system of humans can learn novel concepts and features from a few examples quickly and then recognizes objects of the same kind in other images, even when the objects are under large variance of view, illumination, and shape, etc.

In real-world scenarios, the number of training samples could be small, and the cost to label high-quality samples can be very high. Thus, it is important to study the problem of *few-shot object detection* (FSOD) [2, 10]. Specifically, given a few *support images* with annotations of some target classes, FSOD aims to recognize and localize objects of these classes in some *query images*. Similar to few-shot learning, the class space is divided into base (seen) classes and novel (unseen) classes. During training, only base classes are used. The inference is conducted on novel classes, which are different from the base classes.

FSOD is a challenging problem that has not yet been extensively studied. Among the existing works, some are based on one-stage detectors, which usually have relatively low accuracy [10, 28]. In contrast, two-stage detector can achieve relatively high accuracy [30, 11, 5, 27]. However, existing two-stage methods still have some obvious drawbacks: 1) Support information is not fully exploited to guide bounding boxes generation from the query. 2) In $k$-shot scenario, they treat different supports equally and aggregate their features by simple averaging, which will also seriously impact detection accuracy. 3) Generally, their detection accuracy is still too low to be applicable in real applications.

To address the problems above, in this paper, we propose a novel method for accurate few-shot object detection. Our method is a two-stage detector with multi-level architecture. In the first stage, we employ a *support-query mutual guidance* mechanism to generate more support-relevant proposals. To this end, on the one hand, considering that the similarities between different supports and the query image may be quite different, it is natural to think that the supports contribute differently to detect objects from the query image. Thus, we design a *query-guided support weighting*

[*]correspondence author

module to aggregate different supports. On the other hand, we develop a *support-guided query enhancement* module. In particular, we design a kernel generator to dynamically generate *support-specific kernels*, which are used to convolve the query feature, making the query feature more attentive to the support class.

In the second stage, we filter the proposals with a *multi-level proposal scoring* module. Specifically, we perform dense comparison level-by-level between the feature of each proposal candidate and the support feature, and aggregate level-wise similarities to get proposal scores. The feature comparison is done with a learnt *distance metric*. To learn a better embedding space for the distance metric, we design a *hybrid loss* that skillfully combines the merits of contrastive loss, adaptive margin loss and focal loss to effectively discriminate different novel classes and the background. Our extensive experiments show that the proposed method outperforms the existing methods.

In summary, our contributions are as follows:

1) We propose a new few-shot object detection method whose novelty is twofold: a) A *support-query mutual guidance mechanism* for generating more support-relevant proposals, which is implemented by a *support-guided query enhancement* module and a *query-guided support weighting* module. b) A *hybrid loss* that combines the merits of contrastive loss, adaptive margin loss and focal loss to learn the distance metric for accurately discriminating proposals of different unseen classes and the background.

2) We conduct extensive experiments on benchmark datasets. Results show that our method substantially outperforms the existing methods, and advances the SOTA of FSOD to a higher level.

For better understanding the difference between our method and the latest existing FSOD models, in Table 1 we present a qualitative comparison from three perspectives: *support fusion method*, *support-query mutual guidance mechanism* and *loss function*. We can see that 1) only our method adopts weighted averaging over different supports with their similarities to the query, while existing methods all use simple averaging, i.e., they treat all supports equally. 2) Our method is the only one that uses support-query mutual guidance, i.e., using query to weight supports ($Q \rightarrow S$) meanwhile using support feature to enhance query ($S \rightarrow Q$). 3) Our method employs a powerful hybrid loss that is different from that of the existing models.

## 2. Related Work

**Few-shot Learning**. Few-shot classification can be roughly divided into two main streams: metric-based methods and optimization-based methods. Optimization-based methods [20, 1, 7, 25, 32] aim to train models that can generalize well to the new tasks with model parameters finetuning. Our method is more related to metric-based methods [13, 26, 12, 14, 9], which aim to represent samples in a feature space where data from different categories can be distinguished by distance metrics.

| Method | Supports fusion | S-Q mutual guidance | Loss |
|---|---|---|---|
| LSTD*[2] | Simple avg. | None | CE |
| RepMat[11] | Simple avg. | None | CE+Emb |
| MetaYOLO[10] | Simple avg. | $S \rightarrow Q$ | CE |
| MetaRCNN[30] | Simple avg. | $S \rightarrow Q$ | Meta loss |
| TFA*[27] | Simple avg. | None | CE |
| A-RPN[5] | Simple avg. | $S \rightarrow Q$ | CE |
| Ours | Weighted avg. | $S \leftrightarrow Q$ | Hybrid loss |

Table 1. A qualitative comparison of our method with existing ones. Method with '*' means the method is based on finetuning. 'S': Support. 'Q': Query. RPN and regression loss are omitted.

**Few-shot Object Detection (FSOD)**. Different from few-shot classification, FSOD is more challenging. Up to now, there are only several works in this area, which can be divided into finetuning-based methods and finetuning-free methods. Finetuning-based methods [2, 31, 27] formulate this problem in a transfer learning setting. They finetune the pre-trained model from source domain to target domain. Most recent works do not require finetuning. Among them, methods based on one-stage detectors [10, 22, 28] use support information to perform feature fusion with query features, in order to locate the support category. These methods usually have low accuracy due to the lack of full utilization of support information. Methods based on two-stage detectors [11, 30, 6, 5, 29] usually use metric learning to address the FSOD task. Proposals are filtered according to the distance between support and proposal features. Nevertheless, existing two-stage methods have various drawbacks, which limits their performance.

Different from the methods above, our method uses a support-query mutual guidance mechanism to obtain more support-relevant proposals. Furthermore, we propose a hybrid loss function to learn distance metric for better few-shot object detection.

## 3. Method

### 3.1. Problem Definition

Similar to few-shot learning, we align training and testing with the episodic paradigm. In each episode, we first randomly select a class $c$ and $k$ supports of class $c$. Here, the $k$ supports are $k$ objects possibly from multiple support images. Then, we train a detector by episodic meta-training with the input $k$ supports to detect all objects of class $c$ in a query image $Q$. The difference between meta-testing and meta-training lies in that the ground-truth bounding boxes of class $c$ in $Q$ are available only during meta-training.

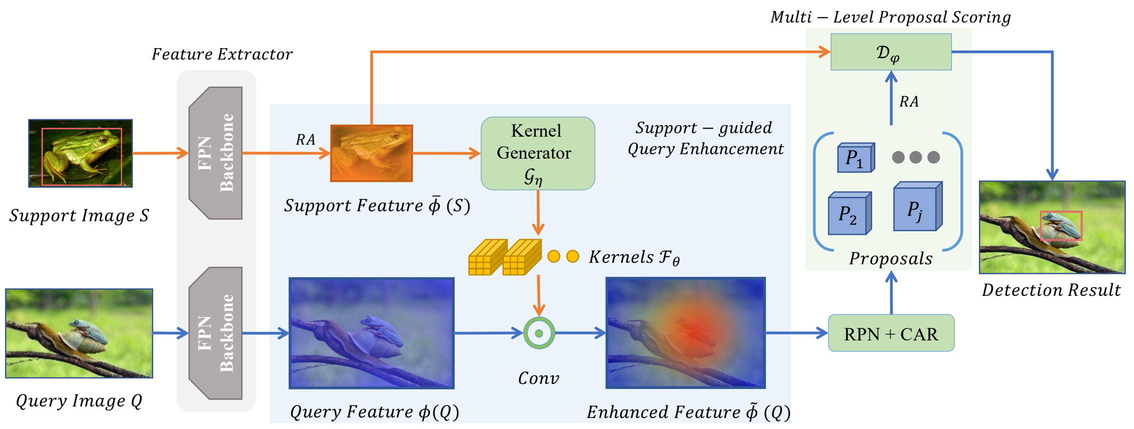
Feature Extractor
Multi − Level Proposal Scoring
$\mathcal{D}_\varphi$
RA
FPN Backbone
RA
Kernel Generator $\mathcal{G}_\eta$
Support − guided Query Enhancement
Support Image S
Support Feature $\bar{\phi}$ (S)
$P_1$
$P_2$
$P_j$
Proposals
Kernels $\mathcal{F}_\theta$
Detection Result
FPN Backbone
RPN + CAR
Conv
Query Image Q
Query Feature $\phi(Q)$
Enhanced Feature $\tilde{\phi}$ (Q)

Figure 1. The framework of our method in 1-shot setting.

Classes used in training are disjoint with those in testing.

## 3.2. Framework

We present the framework of our method in 1-shot setting without loss of generality in Fig. 1. Here, there are two major modules: *support-guided proposal generation* (SPG) and *multi-level proposal scoring* (MPS). In SPG, the *feature extractor* first extracts the features of support and query, then the *support-guided query enhancement* module enhances the query feature by dynamic kernel convolution, and finally more support-relevant proposals are generated. After that, the MPS module filters proposals by dense feature comparison using a *distance metric*, which is learnt using the proposed *hybrid loss*. For $k$-shot setting, a *query-guided support weighting* (QSW) module is used to automatically aggregate the $k$ supports.

## 3.3. Support-guided Proposal Generation (SPG)

### 3.3.1 Feature Extractor

As we begin with 1-shot setting, only one support image $S$ with one annotated bounding box $y^s(c)$ of class $c$ is available during an episode. We use siamese backbone with feature pyramid network (FPN) $\phi(\cdot)$ to extract multi-level features from support image $S$ and query image $Q$. We denote image feature of support and query at the $i^{th}$ level as $\phi^i(S) \in \mathbb{R}^{H_S^i \times W_S^i \times C_1}$ and $\phi^i(Q) \in \mathbb{R}^{H_Q^i \times W_Q^i \times C_1}$ respectively. Here, $i \in [1, L]$, $L$ is the number of FPN levels, $C_1$ is the number of channels. Considering that the support object is only a small patch of the image $S$, we get a size-fixed feature $\overline{\phi^i}(S) \in \mathbb{R}^{a \times a \times C1}$ of the support object with ROI Align ($RA$) operation, i.e., $\overline{\phi^i}(S) = RA(\phi^i(S), y^s(c))$, where $a$ is the output size of $RA$.

### 3.3.2 Support-guided Query Enhancement by Dynamic Kernel Convolution

Some existing methods [11, 2, 27] do not use support to enhance the query, thus generate a large number of support-irrelevant proposals, and have to filter them in the subsequent steps. SiamRPN++ [15], MetaYOLO [10], Attention-RPN [5] and MetaRCNN [30] perform channel-wise multiplication to reweight the feature by depth-wise cross correlation, while [6] uses pixel-wise weighting to calculate pixel-wise weights for feature map with non-local operation. Here, we choose to enhance query by support feature with dynamic convolution [3, 16, 8]. Convolution operation performs feature fusion between convolution kernels and features. However, traditional convolution kernels stay fixed after training. Inspired by dynamic network [3, 16, 8], we use a *kernel generator* $\mathcal{G}_\eta$ to dynamically generate support-specific kernels ${\mathcal{F}_\theta}^i$ conditioned on the support feature $\overline{\phi^i}(S)$,

$${\mathcal{F}_\theta}^i = \mathcal{G}_\eta(\overline{\phi^i}(S)) \tag{1}$$

where $\theta \in \mathbb{R}^{b \times b \times C_1 \times C_2}$ indicates parameters, $b$ is the kernel size, $C_1$ is the number of channels in $\overline{\phi^i}(S)$, $C_2$ is the number of kernels.

Then, to highlight the regions containing the target objects, we use the generated kernels to enhance the query feature as follows:

$$\widetilde{\phi^i}(Q) = {\mathcal{F}_\theta}^i \odot \phi^i(Q) \ \in \mathbb{R}^{(\frac{H_Q^i - b}{r}+1) \times (\frac{W_Q^i - b}{r}+1) \times C_2} \tag{2}$$

where $\odot$ is convolutional operation, $\widetilde{\phi^i}(Q)$ is the enhanced query feature, and $r$ is the stride of convolution. With the guidance of support information, objects of support class in the generated query feature are enhanced. The operations of Eq. (1) and Eq. (2) are performed at each level $i$.

In our experiments, we implement $\mathcal{G}_\eta$ with $C_2$ convolution subnetworks, whose kernel size are $a - b + 1$, input

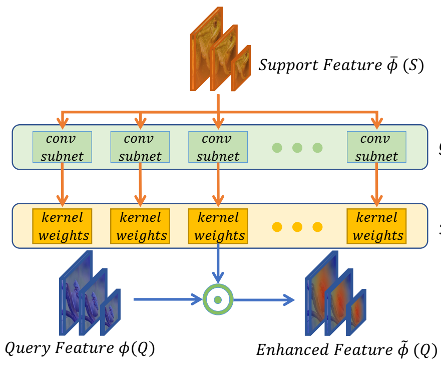

Figure 2. The structure of kernel generator $\mathcal{G}_\eta$.

channel $C_1$, output channel $C1$, indicating generating $C_2$ kernels and each with $C_1$ channels. The parameters $\eta$ of $\mathcal{G}_\eta$ are shared across all levels. The structure of kernel generator $\mathcal{G}_\eta$ is shown in Fig. 2.

### 3.3.3 Proposal Generation and Refinement

$\widetilde{\phi^i}(Q)$ is fed into RPN head to propose potential boxes that contain objects of class $c$ with high probability. To further boost the accuracy of generated proposals, we add a *class-agnostic regressor (CAR)* module after RPN to refine the positions and sizes of proposals, which is implemented in a fully connected network. CAR takes the features of proposals as input and outputs the coordinate adjustment values $(dx, dy, dh, dw)$ where $dx$ and $dy$ are used to adjust the top-left coordinates of the proposals, $dh$ and $dw$ are used to adjust the height and width. The refined proposals are used in the following steps.

## 3.4. Multi-Level Proposal Scoring (MPS)

To filter out the low-quality proposals, we develop the MPS module to rank the generated proposals and output the top ranked proposals as the final detection result.

To rank the $j^{th}$ proposal $P_j$, we first project $P_j$ into all FPN levels and use ROI Align to get its features at all $L$ levels, which are denoted as $\{\overline{P_j^i}\}_{i=1}^L$. Then, we perform dense feature comparison between support and proposal $P_j$ level-by-level using a learnt distance metric $\mathcal{D}_\varphi(\cdot,\cdot)$ and get the similarity score $sim(S, P_j)$ as follows:

$$sim(S, P_j) = \frac{1}{L}\sum_{i=1}^{L}\mathcal{D}_\varphi(\overline{\phi^i}(S), \overline{P_j^i}) \tag{3}$$

In our experiments, we implement $\mathcal{D}_\varphi(\cdot,\cdot)$ with relation network [26] without the last *sigmoid* activation and its parameters $\varphi$ is learnable during the training phase.

By ranking the $M$ candidate proposals $\{P_j\}_{j=1}^M$, we output the final proposals according to a prespecified score threshold and non maximum suppression.

## 3.5. Query-guided Support Weighting (QSW)

For $k$-shot setting, we propose the QSW module to weight different support objects. Fig. 3 illustrates the QSW architecture of 3-shot setting.

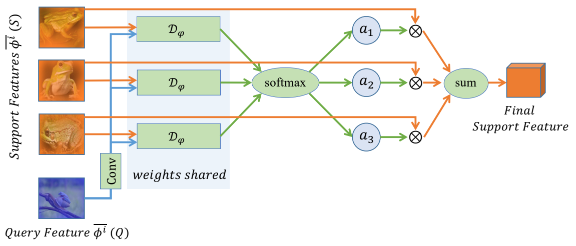

Figure 3. The QSW architecture of 3-shot setting.

As different support objects may differ in view, shape and illumination etc., their relevance to the query image may be quite different. Different from previous weighting scheme [33], we weight each support object according to its similarity with the query image using an attention mechanism. Taking the $i^{th}$ level for example, given the $k$ support features $\{\overline{\phi_j^i}(S)\}_{j=1}^k$ and the original size query feature $\phi^i(Q)$ at level $i$, we get the size-fixed query feature $\overline{\phi^i}(Q)$ with $RA$ operation. Then, the attention scores $\{a_j^i\}_{j=1}^k$ to weight different supports are as follows:

$$c_j^i = \mathcal{D}_\varphi(\overline{\phi_j^i}(S), Conv(\overline{\phi^i}(Q))), \tag{4}$$

$$a_j^i = \frac{e^{c_j^i}}{\sum\limits_{j=1}^{k} e^{c_j^i}} \tag{5}$$

where $\mathcal{D}_\varphi(\cdot,\cdot)$ is the distance metric in MPS, $Conv(\cdot)$ is a convolution operation to extract semantic feature and compress the feature into a uniform space with the proposals. The parameters $\varphi$ for $\mathcal{D}_\varphi(\cdot,\cdot)$ are shared between MPS and QSW. The final support feature at level $i$ is the weighted sum of $\{\overline{\phi_j^i}(S)\}_{j=1}^k$.

## 3.6. Hybrid Loss

In FSOD, we often face two problems: 1) *False positives*. For example, given a support object of class $c$, and a query image $Q$, even though $Q$ does not contain any object of class $c$, the detector may still output some bounding boxes, which are false positives. This is because the learnt metric space tends to focus on separating foreground and background, ignoring different classes of foreground objects. 2) *Foreground-background imbalance*. Most proposals $P_j$ are from background, leading to an extreme imbalance between foreground and background classes during training the distance metric $\mathcal{D}_\varphi(\cdot,\cdot)$. Here, we address these two problems by introducing a hybrid loss that combines the merits of contrastive loss, adaptive margin loss and focal loss.

### 3.6.1 Basic Contrastive Loss

We first solve the false positive problem mentioned above. A basic idea is to treat proposals of other foreground classes as negative samples, i.e. comparison targets. For joint training, similar to 2-way training strategy in [5], we try to make the model learn to compare foreground and background proposals, and different foreground classes. Let $\{P_c\}$ and $\{P_{nc}\}$ be the sets of foreground proposals of class $c$ and the other classes respectively, and $\{P_{back}\}$ be the set of proposals from the background, we formulate our basic contrastive loss as follows:

$$\mathcal{L}_{contra} = -\frac{1}{|\{P_c\}|} \log \frac{\mathcal{N}(C)}{\mathcal{N}(C) + \mathcal{N}(NC, Back)^{contra}} \tag{6}$$

where

$$\begin{aligned} \mathcal{N}(C) &= \sum_{P \in \{P_c\}} e^{sim(S,P)} \\ \mathcal{N}(NC, Back)^{contra} &= \sum_{P \in \{P_{nc}, P_{back}\}} e^{sim(S,P)} \end{aligned} \tag{7}$$

With $\mathcal{L}_{contra}$, we can separate different foreground classes.

### 3.6.2 Adaptive Margin

An shortcoming of $\mathcal{L}_{contra}$ is that different classes in the embedding space are not separated by a proper distance. An intuitive idea is to add margin between different classes. Inspired by [14], we introduce the adaptive margin mechanism to $\mathcal{L}_{contra}$ by rewriting $\mathcal{N}(NC, Back)^{contra}$ as follows:

$$\begin{aligned} \mathcal{N}(NC, Back)^{margin} = &\sum_{P \in \{P_{nc}\}} e^{sim(S,P)+M_{S,P}} \\ &+ \sum_{P \in \{P_{back}\}} e^{sim(S,P)+M_{S,P}} \end{aligned} \tag{8}$$

where $M_{S,P}$ is the adaptive margin generated according to the semantic similarity between classes as follows:

$$M_{S,P} = \mu \cdot \mathcal{D}_\tau(v_S, v_P) + \nu \tag{9}$$

where $v_S$ and $v_P$ are semantic vectors of support and proposals classes extracted by a word embedding model. $\mathcal{D}_\tau$ is a metric to measure the semantic similarity between classes. In our experiments, we get semantic vectors from Glove [21] and implement $\mathcal{D}_\tau$ by cosine similarity. Parameters $\mu, \nu$ are learnable during the training. Similarity between support and background proposals $\mathcal{D}_\tau(v_S, v_{back})$ is a hyperparameter.

### 3.6.3 Imposing Focal Loss to Background

Here we address the foreground-background imbalance problem. A simple solution is to downsample $\{P_{back}\}$ randomly or using OHEM [24], which can be considered as hard-weighting to $\{P_{back}\}$, i.e., assigning weights of 1 (chosen) or 0 (ignored) to the proposals. Inspired by focal loss [18], we adaptively down-weight the easy-separating $P_{back}$ and focus on the hard negative samples by reshaping $\mathcal{N}(NC, Back)^{margin}$ to

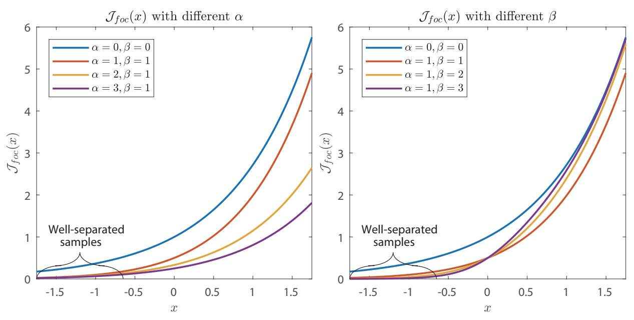

Figure 4. Plots of $\mathcal{J}_{foc}(x)$ with different $\alpha$ and $\beta$ values.

$$\begin{aligned} \mathcal{N}(NC, Back)^{margin+focal} = &\sum_{P \in \{P_{nc}\}} e^{sim(S,P)+M_{S,P}} \\ &+\gamma \sum_{P \in \{P_{back}\}} \mathcal{J}_{foc}(sim(S,P) + M_{S,P}) \end{aligned} \tag{10}$$

where $\gamma$ is a hyperparameter that balances the sum of negative sample terms. $\mathcal{J}_{foc}(x)$ is a function defined as follows:

$$\mathcal{J}_{foc}(x) = f_m(x)e^x. \tag{11}$$

Above, $f_m(x)$ is a modulating factor as follows:

$$f_m(x) = \frac{1}{\alpha + e^{-\beta x}}. \tag{12}$$

We implement $f_m(x)$ with a parameterized sigmoid function. Fig. 4 illustrates $\mathcal{J}_{foc}(x)$ with different $\alpha$ and $\beta$ values. When $\alpha = 0, \beta = 0$, $\mathcal{J}_{foc}(x)$ degenerates to $e^x$. In this case, even well-separable proposals contribute much to the loss, and the contribution from $\{P_{back}\}$ can overwhelm that from $\{P_c\}$ and $\{P_{nc}\}$. We can see that a large $\alpha$ slows down the growth of $\mathcal{J}_{foc}(x)$, so as to prevent the negative proposals from dominating the gradient. And a large $\beta$ substantially restrains the values for well-separable proposals, allowing the model to pay more attention to proposals that are difficult to distinguish.

By replacing $\mathcal{N}(NC, Back)^{contra}$ in Eq. (6) with $\mathcal{N}(NC, Back)^{margin+focal}$ in Eq. (10), we get a *hybrid loss* $\mathcal{L}_{HY}$ that combines the merits of contrastive loss, adaptive margin loss and focal loss to effectively learn the metric distance of FSOD.

### 3.6.4 Total Loss Function

Finally, the total loss of our model is as follows:

$$\mathcal{L} = \mathcal{L}_{\text{RPN}_{\text{Cls}}} + \mathcal{L}_{\text{RPN}_{\text{Reg}}} + \lambda_1 \mathcal{L}_{\text{CAR}} + \lambda_2 \mathcal{L}_{\text{HY}} \tag{13}$$

where the first two terms are the cross entropy and regression loss of RPN respectively. $\mathcal{L}_{\text{CAR}}$ is the loss of the class-agnostic regressor, which is in the form of smooth $L_1$ loss. $\mathcal{L}_{\text{HY}}$ is the hybrid loss.

## 4. Experiments

### 4.1. Datasets

Following the settings of existing works, the datasets and splits are as follows:

**MS COCO dataset** [19]. It is a benchmark dataset for object detection. There are 80 categories in COCO, 20 of which also appear in PASCAL VOC. We select the 20 common categories with VOC as novel classes for evaluation, and the remaining 60 classes as base classes for training. Training is performed on the base classes with 80k training images and a 35k subset of val images, and evaluation is on the remaining 5k val images.

**PASCAL VOC dataset** [4]. VOC 2007 and 2012 are divided into training set, validation set and test set. We utilize 2007 and 2012 train/val images for training, and 2007 test set for evaluation. There are totally 20 categories of annotated objects and we follow previous works to take three different base/novel classes splits. In each split, there are 15 base classes for training and 5 novel classes for evaluation.

### 4.2. Implementation Details

We report results with ResNet-50/101 as backbone and the number of FPN levels is set to 5. We train our model on 4 NVIDIA TITAN RTX in parallel. The model is trained in an end-to-end manner for totally 60000 episodes. We employ SGD with a weight decay of 0.0001 and a momentum of 0.9 as optimizer. For the first 40000 episodes, the learning rate is 0.002, and we multiply it by 0.1 for the next 20000 episodes. The short side of each image is resized to 800 pixels, and the longer side is capped at 1333. $C_1$ and $C_2$ are set to 256. The output size $a$ of ROI Align is set to 7. In SPG, the kernel size $b$ of $\mathcal{F}_\theta$ and the stride $r$ in Eq. (2) are set to 1. For the hybrid loss, when $P$ is background, $\mathcal{D}_\tau(v_S, v_{back})$ in Eq. (9) is set to 0.3, $\gamma$ in Eq. (10) is set to 0.25, $\alpha$ and $\beta$ in Eq. (12) are set to 1.5, $\lambda_1$ and $\lambda_2$ in Eq. (13) are set to 0.01 and 1 respectively.

### 4.3. Comparison with Existing Methods

**COCO dataset**. The results on COCO are given in Table 2. Since previous works used different backbones, for fair comparison we report our results of two backbones. We can see that even with ResNet-50 as backbone, our method exceeds all the existing SOTAs by a large margin. With ResNet-101 as backbone, our method further brings a 1.3%AP improvement over ResNet-50 and outperforms SOTAs by 2.8%, 9.1% and 1.1% in terms of $AP$, $AP_{50}$, $AP_{75}$ respectively. This shows the generalization power of our approach to novel classes. Furthermore, we can see that our method performs more excellently in terms of $AP_S$ than the existing methods, almost doubles the $AP_S$ of MPSR [29], which demonstrates the superiority of our method in detecting small-size objects. Results under 30-shot are in the supplementary file.

| Method | Bb | $AP$ | $AP_{50}$ | $AP_{75}$ | $AP_S$ | $AP_M$ | $AP_L$ |
|---|---|---|---|---|---|---|---|
| LSTD[2] | SSD | 3.2 | 8.1 | 2.1 | 0.9 | 2.0 | 6.5 |
| Incremental[22] | H104 | 5.1 | – | – | – | – | – |
| MetaYOLO[10] | D19 | 5.6 | 12.3 | 4.6 | 0.9 | 3.5 | 10.5 |
| MetaDet[28] | V16 | 7.1 | 14.6 | 6.1 | 1.0 | 4.1 | 12.2 |
| MetaRCNN[30] | R101 | 8.7 | 19.1 | 6.6 | 2.3 | 7.7 | 14.0 |
| TFA w/cos[27] | R101 | 10.0 | – | 9.3 | – | – | – |
| MPSR[29] | R101 | 9.8 | 17.9 | 9.7 | 3.3 | 9.2 | 16.1 |
| A-RPN[5] | R50 | 11.1 | 20.4 | 10.6 | – | – | – |
| Ours | R50 | 12.6 | 27.0 | 10.9 | 7.3 | 13.4 | 17.8 |
| Ours | R101 | **13.9** | **29.5** | **11.7** | **7.6** | **15.2** | **19.0** |

Table 2. Results on the COCO minival set for 20 novel classes under 10-shot. '–': No reported results. 'Bb': Backbone; 'SSD': the models use backbone of SSD; 'H104': Hourglass-104; 'D19': DarkNet-19; 'V16': VGG16; 'R101/50': ResNet101/50.

**VOC dataset**. Table 3 presents the results on VOC dataset, we can see that our method performs best in most cases. Especially, when there are fewer supports, our method performs much better than the existing methods. In 1-shot case, our method with ResNet-101 as backbone exceeds SOTA (MPSR [29]) by 11.5% mAP on average over 3 splits. Furthermore, to measure the overall performance of different methods, we take the average of 5 shots over 3 splits and present the results in the last column. For fair comparison, methods that do not disclose all shot results are ignored and are replaced with '-'. As we can see, the improvement on average score is up to 10 points, much larger than the gap among previous approaches, demonstrating the effectiveness of our approach.

### 4.4. Ablation Study

We analyze the effects of various components in our system. Unless otherwise specified, the experiments are carried out on COCO under 10-shot with ResNet-50 as backbone.

**Effects of Different Modules**. Here we investigate the effects of different modules and summarize the results in Table 4. The implementation details are: 1) For models without kernel generator $\mathcal{G}_\eta$, we directly feed the origin query feature $\phi(Q)$ into RPN. 2) For models without CAR, we use the proposals from RPN without location refinement. 3) For models without MPS, we perform feature comparison only at the top level of FPN. 4) For models without the hybrid loss (*HY loss* in short), we use binary cross entropy loss instead. 5) For models without QSW, all supports have the same weights.

| Method / Shot | Back bone | Novel Set 1 | | | | | Novel Set 2 | | | | | Novel Set 3 | | | | | Avg |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 5 | 10 | 1 | 2 | 3 | 5 | 10 | 1 | 2 | 3 | 5 | 10 | |
| LSTD[2] | SSD | 8.2 | 1.0 | 12.4 | 29.1 | 38.5 | 11.4 | 3.8 | 5.0 | 15.7 | 31.0 | 12.6 | 8.5 | 15.0 | 27.3 | 36.3 | 17.1 |
| MetaYOLO[10] | D19 | 14.8 | 15.5 | 26.7 | 33.9 | 47.2 | 15.7 | 15.3 | 22.7 | 30.1 | 39.2 | 19.2 | 21.7 | 25.7 | 40.6 | 41.3 | 27.3 |
| MetaDet[28] | V16 | 18.9 | 20.6 | 30.2 | 36.8 | 49.6 | 21.8 | 23.1 | 27.8 | 31.7 | 43.0 | 20.6 | 23.9 | 29.4 | 43.9 | 44.1 | 31.0 |
| MetaRCNN[30] | R101 | 19.9 | 25.5 | 35.0 | 45.7 | 51.5 | 10.4 | 19.4 | 29.6 | 34.8 | 45.4 | 14.3 | 18.2 | 27.5 | 41.2 | 48.1 | 31.1 |
| Context-Trans[31] | SSD | 34.2 | – | – | 44.2 | – | 26.0 | – | – | 36.3 | – | 29.3 | – | – | 40.8 | – | – |
| TFA w/cos[27] | R101 | 39.8 | 36.1 | 44.7 | **55.7** | 56.0 | 23.5 | 26.9 | 34.1 | 35.1 | 39.1 | 30.8 | 34.8 | 42.8 | 49.5 | 49.8 | 39.9 |
| MPSR[29] | R101 | 41.7 | – | 51.4 | 55.2 | **61.8** | 24.4 | – | 39.2 | 39.9 | 47.8 | 35.6 | – | 42.3 | 48.0 | 49.7 | – |
| Ours | R50 | 46.8 | 49.2 | 50.2 | 52.0 | 52.4 | 39.4 | 43.1 | 43.6 | 44.1 | 45.7 | 44.1 | 49.8 | 50.5 | 52.3 | 52.8 | 47.7 |
| Ours | R101 | **48.6** | **51.1** | **52.0** | 53.7 | 54.3 | **41.6** | **45.4** | **45.8** | **46.3** | **48.0** | **46.1** | **51.7** | **52.6** | **54.1** | **55.0** | **49.8** |

Table 3. mAP on the PASCAL VOC dataset. The evaluation is performed over 3 different splits. '–': No reported results or missing data.

| $\mathcal{G}_\eta$ | CAR | MPS | HY loss | QSW | $AP_{50}$ | $AP_{75}$ |
|---|---|---|---|---|---|---|
| | | | | | 15.6 | 3.2 |
| ✓ | | | | | 18.8 | 4.0 |
| ✓ | ✓ | | | | 19.3 | 7.4 |
| ✓ | ✓ | ✓ | | | 23.1 | 9.1 |
| ✓ | ✓ | ✓ | ✓ | | 26.5 | 10.7 |
| ✓ | ✓ | ✓ | ✓ | ✓ | **27.0** | **10.9** |

Table 4. Effects of different modules.

| Loss | $AP_{50}$ | $AP_{75}$ |
|---|---|---|
| Basic Contrastive Loss | 18.0 | 8.5 |
| Basic Loss + AM | 18.7 | 8.7 |
| Basic Loss + AM + Downsampling | 24.4 | 9.5 |
| Basic Loss + AM + OHEM | 25.1 | 9.8 |
| Hybrid Loss (Ours) | **27.0** | **10.9** |

Table 5. Effect of hybrid loss. 'AM': Adaptive Margin.

We can see that models removing any module we design get worse performance. With the kernel generator $\mathcal{G}_\eta$, performance gets 3.2%/0.8% improvement in terms of $AP_{50}$/$AP_{75}$, which shows that it is helpful to introduce support information to guide the generation of proposals. Applying CAR can increase $AP_{75}$ up to 3.4%, due to more accurate positions of predicted boxes. The MPS module can increase $AP_{50}$ and $AP_{75}$ by 3.8% and 1.7% respectively, which means the feature comparison at all levels is necessary. As for the hybrid loss, it boosts the performance by 3.4%/1.6% in terms of $AP_{50}$/$AP_{75}$, which means our hybrid loss is helpful for learning desirable distance metric. The QSW module brings a slight performance lift, but when the number of supports is small, the effect is obvious, which will be analyzed later.

**Effect of the Hybrid Loss**. We compare our hybrid loss with some streamlined versions to evaluate the effectiveness of the key components of our loss. Results are presented in Table 5. We can see that: 1) Basic contrastive loss has poor performance since the negative proposals dominate the gradient. 2) Adaptive margin can improve basic contrastive loss moderately due to a better embedding space with adaptive inter-class distance. Moreover, we observe that the learnt coefficient $\mu$ in Eq. (9) is positive, showing that the margin between similar classes should be larger than that between dissimilar classes. 3) Hard-weighting methods such as downsampling and OHEM [24] can solve the imbalance problem to some extent, there is still room for performance improvement. 5) By considering the inter-class distance of different foreground classes and the easy-hard negative proposals, our hybrid loss outperforms all the other loss functions.

Moreover, we compare visualization of embedding space between BCE loss and hybrid loss (HY loss). Fig. 5 shows the visualization results of the first 5 novel class features obtained by tSNE. Our hybrid loss has better separation among novel classes. In BCE, intra-class distance is minimized but inter-class distance is not considered. It works well for separating base classes, but is not satisfactory for novel classes, because inter-class distance must be increased to ensure unseen class separability.

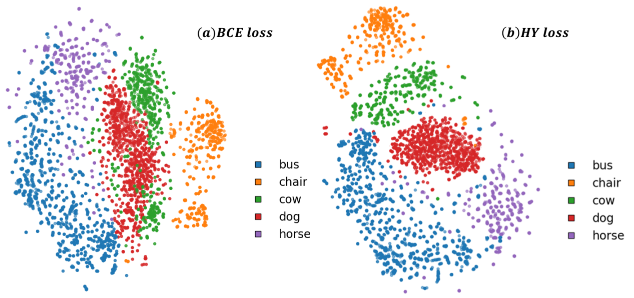

Figure 5. tSNE plots of visual features for the first 5 novel classes in COCO. (a) BCE loss pushes the visual features to their groundtruth, which leads to the minimizing of intra-class distance, but ignoring of inter-classes distance. (b) Our HY loss considers both to ensure novel classes separability.

**Hyperparameter Analysis**. We perform analysis on 4 important hyperparameters: $\alpha$, $\beta$, $\gamma$ and $b$. We vary $\alpha$ with a fixed $\beta = 0$. Then, with the optimal $\alpha$, we vary $\beta$ from 0 to 3. Similarly, we study the influence of $\gamma$ and $b$. The results are shown in Fig. 6, with which we have the settings in implementation details.

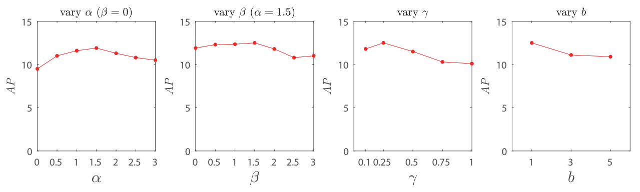

Figure 6. Hyperparameter analysis.

**Will $\mathcal{G}_\eta$ and CAR Improve Recall of Proposals**? To verify that RPN can generate more support-relevant proposals when the query feature is convolved by the support-specific kernel $\mathcal{F}_\theta$, we check the recall with/without kernel generator $\mathcal{G}_\eta$. Meanwhile, to verify that CAR can further improve the accuracy of the bounding box location, we also check the recall with/without CAR. The results are shown in the left part of Fig. 7. We can see that the recall when $\mathcal{G}_\eta$ and CAR are used is much better than the other settings. Without $\mathcal{G}_\eta$, the recall curve starts from a relatively small value when IoU = 0.5 (up to 10%). For models without CAR, the recall curve drops sharply as IoU increases, indicating that the positions of the proposals are not accurate enough. In addition, we compare our kernel generator with other query feature enhancement methods, depth-wise cross correlation (DCC) [15, 10, 5, 30] and pixel-wise weighting [6]. The results are in the right part of Fig. 7. For fair comparison, CAR is used in all methods. As we can see, our query feature enhancement scheme outperforms the others and can effectively improve recall, and better recall of proposals subsequently makes higher $AP$ possible.

**Does Multi-Level Scoring with FPN Help**? In addition to our method, we consider two additional schemes for proposals scoring conducted on only one feature level, they are: a) Top-Level: Feature comparison is done only at the top level of FPN. In this case, FPN is removed and feature aggregation is only performed at the top level. b) Optimal-Level: It also compare feature in one level, but the level is selected according to the size of the proposals. The selection method is the same as that in [17]. The results are summarized in Table 6. We can see that our method outperforms the other two, which indicates the feature comparison level-by-level is useful and necessary, especially for small-size objects.

**Effect of QSW**. To verify the effectiveness of the QSW module, we compare QSW with feature averaging, which is used in some existing methods [30, 11, 10, 5]. Results are shown in Table 7. As we can see, QSW can boost the performance, especially when the number of supports is small. Since the smaller the support number, the easier it is for some randomly selected low-quality supports or query-irrelevant supports to impact on the results. QSW can measure the contribution from each support to get a better support feature for the current query.

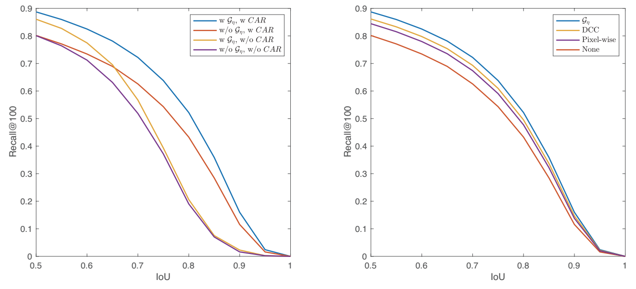

Figure 7. Left: Recall of proposals with/without kernel generator $\mathcal{G}_\eta$ and CAR. Right: Comparison of different query feature enhancement schemes. We keep top-100 proposals per image.

| Method | $AP$ | $AP_S$ | $AP_L$ |
|---|---|---|---|
| Top-Level | 10.3 | 4.0 | 16.7 |
| Optimal-Level | 11.5 | 5.9 | 17.0 |
| MPS (Ours) | **12.6** | **7.3** | **17.8** |

Table 6. Ablation study for MPS with FPN. MPS has a better improvement for $AP_S$ ($AP$ of small size objects) than $AP_L$(large).

| Method/Shot($k$) | 2 | 3 | 4 | 5 | 10 |
|---|---|---|---|---|---|
| Feature Averaging | 22.7 | 23.4 | 24.8 | 25.6 | 26.5 |
| QSW(Ours) | **23.6** | **25.2** | **25.9** | **26.2** | **27.0** |

Table 7. $AP_{50}$ comparison between feature averaging and QSW. QSW can significantly improve performance when $k$ is small.

## 5. Conclusion

In this work, we propose a novel method for few-shot object detection based on support-query mutual guidance ana hybrid loss. On the one hand, support is used to guide the enhancement of query. On the other hand, query is used to guide the weighting of multiple supports in few-shot setting. Furthermore, we adopt multi-level proposal scoring based on a skillfully designed hybrid loss. Our method can effectively overcome the drawbacks of existing works, and achieves the state of the art performance. As for future work, we will focus on improving the efficiency of our model and further boosting its performance by new mechanisms and network structures.

**Acknowledgement**. This work was supported by Zhejiang Lab (No. 2019KB0AB05), partially by Science and Technology Commission of Shanghai Municipality Project (No. 19511120700), and Shanghai Artificial Intelligence Innovation and Development Projects funded by Shanghai Municipal Commission of Economy and Informatization. Jihong Guan was supported by NSFC (No. U1936205).

## References

[1] Antreas Antoniou, Harrison Edwards, and Amos Storkey. How to train your MAML. In *ICLR*, 2019. 2

[2] Hao Chen, Yali Wang, Guoyou Wang, and Yu Qiao. Lstd: A low-shot transfer detector for object detection. In *AAAI*, 2018. 1, 2, 3, 6, 7

[3] Yinpeng Chen, Xiyang Dai, Mengchen Liu, Dongdong Chen, Lu Yuan, and Zicheng Liu. Dynamic convolution: Attention over convolution kernels. In *CVPR*, 2020. 3

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

[5] Qi Fan, Wei Zhuo, Chi-Keung Tang, and Yu-Wing Tai. Few-shot object detection with attention-rpn and multi-relation detector. In *CVPR*, 2020. 1, 2, 3, 5, 6, 8

[6] Ting-I Hsieh, Yi-Chen Lo, Hwann-Tzong Chen, and Tyng-Luh Liu. One-shot object detection with co-attention and co-excitation. In *NIPS*, 2019. 2, 3, 8

[7] Muhammad Abdullah Jamal and Guo-Jun Qi. Task agnostic meta-learning for few-shot learning. In *CVPR*, 2019. 2

[8] Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc V Gool. Dynamic filter networks. In *NIPS*. 2016. 3

[9] Liang Song Jinlu Liu and Yongqiang Qin. Prototype rectification for few-shot learning. In *ECCV*, 2020. 2

[10] Bingyi Kang, Zhuang Liu, Xin Wang, Fisher Yu, Jiashi Feng, and Trevor Darrell. Few-shot object detection via feature reweighting. In *ICCV*, 2019. 1, 2, 3, 6, 7, 8

[11] 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. 1, 2, 3, 8

[12] Junsik Kim, Tae-Hyun Oh, Seokju Lee, Fei Pan, and In So Kweon. Variational prototyping-encoder: One-shot learning with prototypical images. In *CVPR*, 2019. 2

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

[14] Aoxue Li, Weiran Huang, Xu Lan, Jiashi Feng, Zhenguo Li, and Liwei Wang. Boosting few-shot learning with adaptive margin loss. In *CVPR*, 2020. 2, 5

[15] Bo Li, Wei Wu, Qiang Wang, Fangyi Zhang, Junliang Xing, and Junjie Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In *CVPR*, 2019. 3, 8

[16] Ji Lin, Yongming Rao, Jiwen Lu, and Jie Zhou. Runtime neural pruning. In *NIPS*. 2017. 3

[17] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017. 8

[18] Tsung-Yi Lin, Priyal Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39:2999–3007, 2018. 5

[19] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollar, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 6

[20] Jelena Luketina, Mathias Berglund, Klaus Greff, and Tapani Raiko. Scalable gradient-based tuning of continuous regularization hyperparameters. In *ICML*, 2016. 2

[21] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014. 5

[22] Juan-Manuel Perez-Rua, Xiatian Zhu, Timothy M. Hospedales, and Tao Xiang. Incremental few-shot object detection. In *CVPR*, 2020. 2, 6

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

[24] Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *CVPR*, 2016. 5, 7

[25] Xingyou Song, Wenbo Gao, Yuxiang Yang, Krzysztof Choromanski, Aldo Pacchiano, and Yunhao Tang. Es-maml: Simple hessian-free meta learning. In *ICLR*, 2020. 2

[26] 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, 4

[27] Xin Wang, Thomas Huang, Joseph Gonzalez, Trevor Darrell, and Fisher Yu. Frustratingly simple few-shot object detection. In *ICML*, 2020. 1, 2, 3, 6, 7

[28] Yu-Xiong Wang, Deva Ramanan, and Martial Hebert. Meta-learning to detect rare objects. In *ICCV*, 2019. 1, 2, 6, 7

[29] Jiaxi Wu, Songtao Liu, Di Huang, and Yunhong Wang. Multi-scale positive sample refinement for few-shot object detection. In *ECCV*, 2020. 2, 6, 7

[30] Xiaopeng Yan, Ziliang Chen, Anni Xu, Xiaoxi Wang, Xiaodan Liang, and Liang Lin. Meta r-cnn : Towards general solver for instance-level low-shot learning. In *ICCV*, 2019. 1, 2, 3, 6, 7, 8

[31] Ze Yang, Yali Wang, Xianyu Chen, Jianzhuang Liu, and Yu Qiao. Context-transformer: Tackling object confusion for few-shot detection. In *AAAI*, 2020. 2, 7

[32] Huaxiu Yao, Xian Wu, Zhiqiang Tao, Yaliang Li, Bolin Ding, Ruirui Li, and Zhenhui Li. Automated relational meta-learning. In *ICLR*, 2020. 2

[33] Chi Zhang, Guosheng Lin, Fayao Liu, Rui Yao, and Chunhua Shen. Canet: Class-agnostic segmentation networks with iterative refinement and attentive few-shot learning. In *CVPR*, 2019. 4