Towards Few-Shot Open-Set Object Detection

Binyi Su, Hua Zhang, Jingzhi Li, and Zhong Zhou

***Abstract*—Open-set object detection (OSOD) aims to detect the known categories and identify unknown objects in a dynamic world, which has achieved significant attentions. However, previous approaches only consider this problem in data-abundant conditions, while neglecting the few-shot scenes. In this paper, we seek a solution for the few-shot open-set object detection (FSOSOD), which aims to quickly train a detector based on few samples while detecting all known classes and identifying unknown classes. The main challenge for this task is that few training samples induce the model to overfit on the known classes, resulting in a poor open-set performance. We propose a new FSOSOD algorithm to tackle this issue, named Few-shOt Open-set Detector (FOOD), which contains a novel class weight sparsification classifier (CWSC) and a novel unknown decoupling learner (UDL). To prevent over-fitting, CWSC randomly sparses parts of the normalized weights for the logit prediction of all classes, and then decreases the co-adaptability between the class and its neighbors. Alongside, UDL decouples training the unknown class and enables the model to form a compact unknown decision boundary. Thus, the unknown objects can be identified with a confidence probability without any pseudo-unknown samples for training. We compare our method with several state-of-the-art OSOD methods in few-shot scenes and observe that our method improves the recall of unknown classes by 5%-9% across all shots in VOC-COCO dataset setting [1].**

***Index Terms*—Few-shot open-set object detection, over-fitting, class weight sparsification classifier, unknown decoupling learner.**

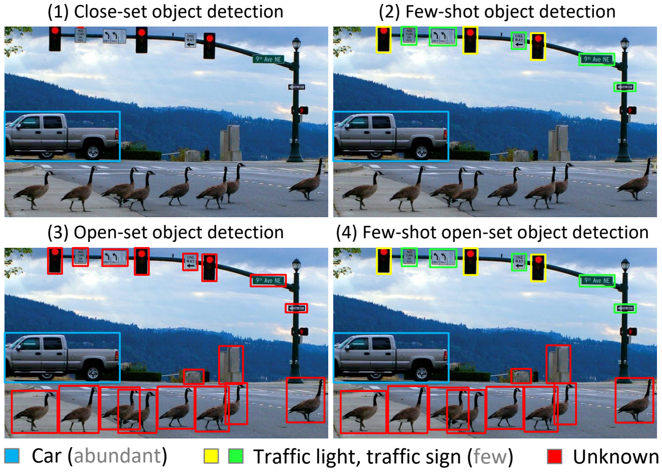
(1) Close-set object detection
(2) Few-shot object detection
(3) Open-set object detection
(4) Few-shot open-set object detection
🟦 Car (abundant)
🟨 🟩 Traffic light, traffic sign (few)
🟥 Unknown

Fig. 1. The visualization of different tasks: closed-set object detection (CSOD), few-shot object detection (FSOD), open-set object detection (OSOD), and few-shot open-set object detection (FSOSOD). The CSOD task can only detect the data-abundant classes in a close-set assumption, where the training and testing sets share the same classes. The FSOD task can detect few-shot classes, while it holds a close-set assumption and cannot detect unknown objects. The OSOD task can detect unknown class, but it requires data-abundant known classes for training. The FSOSOD task can identify the data-abundant known objects, the few-shot known objects, and the unknown objects based on an unbalanced training data.

I. Introduction

Object detection is a fundamental task in the field of computer vision, which aims to localize and recognize objects in an image. With the help of deep learning, object detection has achieved a remarkable progressing. However, existing object detection models [1], [2] are under a strong assumption that there exist enough samples for all the categories, which is time-consuming and expensive to annotate instances for the supervised training.

To alleviate this issue, few-shot object detection (FSOD) methods [3]–[13] are developed to reduce the data dependence of the CNN models. FSOD aims to train a detector based on few samples. Various approaches have presented significant improvements in FSOD problem. However, these methods hold a closed-set assumption, where the training and testing sets share the same classes. In the open-set situations, there are countless unknown classes, not included in the training set. These unknown objects can easily disrupt the rhythm of the close-set models, causing them to identify the unknown classes as known ones with a high confidence score [14].

In order to make the model better handle the open-set scenarios, open-set object detection (OSOD) [14], [15] has been constantly investigated, where the detector trained on the close-set datasets is asked to detect all known classes and identify unknown classes in the open-set conditions. These OSOD methods leverage the good representation of the known classes with sufficient training samples to construct the unknown-class detector. However, open-set detectors suffer from a serious over-fitting problem [16] with few known training samples, which greatly degrades the performance of open-set detection.

In this paper, we seek a solution for the unexplored few-shot open-set object detection (FSOSOD) problem. We commits to training a detector using few samples while detecting all known and unknown objects. FSOSOD has enormous value in safety-critical applications such as autonomous driving and medical analysis. For example, in autonomous driving scenarios, the car needs to detect all known classes (including data-abundant base classes and data-hungry novel classes)

This work was supported by Natural Science Foundation of China under Grant 62272018 and 62072454, Beijing Natural Science Foundation under Grant 4202084, Basal Research Fund of Central Public Research Institute of China under Grant 20212701. (*Corresponding author: Zhong Zhou.*)

B. Su is with the State Key Laboratory of Virtual Reality Technology and Systems, School of Computer Science and Engineering, Beihang University, Beijing 100191, China (e-mail: Subinyi@buaa.edu.cn).

H. Zhang and J. Li are with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China (e-mail: zhanghua@iie.ac.cn, lijingzhi@iie.ac.cn).

Z. Zhou is with the State Key Laboratory of Virtual Reality Technology and Systems, School of Computer Science and Engineering, Beihang University, Beijing 100191, China, and also with Zhongguancun Laboratory, Beijing 100190, China (e-mail: zz@buaa.edu.cn).

[1] Code is available at https://github.com/binyisu/FSOSOD

and identify the unknown objects such as the unexpected obstacles. There are three important reasons to solve the FSOSOD problem. First, a open-set detector that can identify the few-shot classes is more useful than the one that does not. Second, data-abundant open-set detection is a challenge under all settings. However, few-shot open-set detection is harder than the data-abundant open-set detection. Hence, the few-shot setting poses a greater challenge for open-set detection research. Third, like open-set detection, the main challenge of few-shot detection is to make accurate decisions for the unknown data during inference. Once the few-shot detectors have the ability for the unknown detection, they are likely to be more robustness for the real-world applications.

Actually, the data set in the real-world scenes often presents a long-tail distribution [17]. Some categories possess many samples, some categories possess few samples, and the data set is impossible to cover all categories in the world. There are always some unexpected categories that are not included in the data set. FSOSOD aims to use this unbalanced data set to train a detector that can identify all known classes and detect the countless unknown classes. The concept of FSOSOD is expressed in Fig. 1. FSOSOD could be viewed as an extension of the few-shot open-set recognition (FSOSR) [16], [18]–[20]. However, our method is not to migrate the methods from FSOSR to FSOSOD. For example, the previous FSOSR studies adopt pseudo-unknown sample generation methods [18], [19] or prototype-based methods [16], [20] to identify the unknown classes, however, our method is independent of the pseudo-unknown samples or the prototypes that denote the average feature of one class. Furthermore, the previous approaches for OSOD [14], [15], [21]–[26], [32], [33] require abundant known-class samples to train the detector, while this is invalid for the FSOSOD task, which induces the model to overfit on the few-shot known classes, resulting in a poor open-set performance. Therefore, how to solve the over-fitting problem without degrading the performance of the few-shot known classes becomes our main intention.

We know that dropout [27] suppresses over-fitting by reducing the co-adaptation between neurons. Thus, we draw inspiration from the consensus that reduction of the co-adaptability between the class and its neighbors can effectively suppress the over-fitting problem. We propose to identify the unknown classes by decoupling the interactions between the known and unknown classes from two aspects: 1) The optimization process for the unknown class does not consider the interactions with the known classes, decoupling training it; 2) The classifier randomly sparses parts of the normalized weights for the class logit prediction, and then decreases the co-adaptability between the class and its neighbors. Eventually, our method can identify the unknown classes without the performance degradation of the few-shot known classes.

To this end, we propose a novel few-shot open-set object detector, named FOOD, which does not rely on the pseudo-unknown sample generation or prototype to identify the unknown objects. FOOD employs Faster R-CNN [1] as the base detector. We replace the original classifier with a novel class weight sparsification classifier (CWSC) and additionally plug a novel unknown decoupling learner (UDL). These two modules are cooperated with each other to solve the over-fitting problem of FSOSOD. After good optimization, our model can achieve the best few-shot open-set detection performance than other state-of-the-art (SOTA) methods. Our contributions are threefold:

- We propose a new problem, called few-shot open-set object detection (FSOSOD), which aims to quickly train a detector based on few samples while detecting all known and unknown objects.
- We propose a novel FSOSOD algorithm (FOOD) with two well-designed modules: CWSC and UDL, which can improve the model's generalization ability for unknown detection in few-shot scenes.
- We develop the first FSOSOD benchmark. We modify several state-of-the-art OSOD methods into FSOSOD methods. Compared with these methods, our FOOD improves the recall of unknown classes by 5%-9% across all shots in VOC-COCO dataset setting.

This paper is organized as follows: Section II shows an overview of the related works. Section III introduces the proposed method. Section IV presents the extensive experiments. Finally, Section V concludes the paper.

II. Related Work

A. Few-Shot Object Detection

Since the conventional detectors based on the supervised learning require abundant annotated samples for training, few-shot object detection (FSOD) has received significant progresses recently [3]–[7], [10], [11], [28]–[31]. FSOD can be roughly divided into three types. 1) Meta learning-based methods. This type of works aims to learn the task-level knowledge that can be adapted to the new task with few support samples, such as FSRW [28], Meta-RCNN [29], FSOD [30], and Meta-DETR [31]. 2) Transfer-learning based approaches. This line of works adopts a simple two-stage fine-tuning strategy to train the detector, *i.e.*, base-training and few-shot fine-tuning phases, which expects to transfer the general knowledge learned from the base-training phase to the few-shot fine-tuning phase, such as TFA [3], FSCE [4], and DeFRCN [10]. 3) Pseudo-sample generation approaches. This form of works views FSOD as a data unbalanced problem, they employ the data-augmentation technologies to generate the samples of the few-shot classes and train the detector end-to-end, such as [11].

Although these FSOD methods focus on good detection performance under the close-set settings, the unknown detection capability is not guaranteed. We extend a simple FSOD method TFA [3] with various OSOD methods to identity the unknown objects, and find that our method can better detect the unknown classes in few-shot scenes than other state-of-the-art methods.

B. Open-Set Object Detection

Open-set object detection (OSOD) methods intend to detect all known classes and identity the unknown classes, simultaneously. According to the acquisition way of the unknown samples, OSOD can be divided into three categories. 1) Virtual unknown samples synthesis. This type of methods synthesizes the virtual unknown samples to train the unknown branch in the feature space [32] or image space [33]. 2) Select unknown samples from the background. This kind of works selects the background boxes with high uncertainty scores as the unknown class to train the open-set detector, such as ORE [21], UC-OWOD [22], ROWOD [23], and OSODD [24]. 3) Select unknown samples from the known classes. This form of works chooses the known samples with high uncertainty scores as the unknown class to train the open-set detector, such as PROSER [25] and OpenDet [14]. Moreover, there exist several threshold-based methods. This type of works uses the energy or entropy of the predicted box as the uncertainty score, which is compared with a threshold to identify the unknown class, such as DS [15] and MCSSD [26].

The previous methods of OSOD need abundant samples of the known classes to train the model. However, this cannot be satisfied in the few-shot conditions, which causes a serious over-fitting problem of the model to the few-shot known classes, resulting in a poor open-set performance. Inspired by the dropout [27], reduction of the neuron dependence/interaction in optimization can efficiently suppress the over-fitting issue, we propose a class weight sparsification classifier and an unknown decoupling learner to dilute dependencies between all known and unknown classes.

C. Few-Shot Open-Set Recognition

Few-shot open-set recognition (FSOSR) has fascinated scant attentions recently. However, to the best of our knowledge, few-shot open-set object detection is still not exploited. Here, we present several FSOSR works. PEELER [18] utilizes the pseudo-unseen class samples generated from seen classes to train the model. SnaTCHer [16] measures the distance between the query and the transformed prototype, then a distance threshold is set to identify the unseen classes. R3CBAM [19] leverages the outlier calibration network to recognize the objects in FSOSR scenes. SEMAN-G [20] learns an unseen prototype that automatically estimates a task-adaptive threshold for unseen recognition. Different from FSOSR, FSOSOD is a more challenging task, because except for the known classes and the unknown classes, FSOSOD also has a *background* class, which often confuses the detector.

III. Proposed Method

A. Problem Setup

We define the problem setup with reference to TFA [3] and OpenDet [14]. We are given an object detection dataset $D = \{(x, y), x \in \mathbf{X}, y \in \mathbf{Y}\}$, where x denotes an input image and $y = \{(c_i, b_i)\}_{i=1}^{I}$ represents the objects with its class c and its box annotation b. The dataset D is divided into the training set D_{tr} and the testing set D_{te}. $D_{tr} = D_B \cup D_N$ contains K known classes $C_K = C_B \cup C_N = \{1, ..., K = B + N\}$, where $C_B = \{1, ..., B\}$ expresses B data-abundant base classes, and $C_N = \{B + 1, ..., K\}$ denotes N data-hungry novel classes, each with M-shot support samples. D_B and D_N are the training data of the base and novel classes, respectively. We test the detector in D_{te} that includes $C_K = C_B \cup C_N$ known classes and C_U unknown classes. Duo to the countless unknown categories, we merge all of them into one class $C_U = \{K + 1\}$. Our goal is to employ the unbalanced data $D_{tr} = D_B \cup D_N$ to train a detector, which can be used to identify the base classes C_B, the novel classes C_N, the unknown class C_U, and the background class C_{bg}.

B. Baseline Setup

As shown in Fig. 2, Faster R-CNN [1] is adopted as the base detector that is composed of a backbone, a region proposal network (RPN), and an R-CNN. Compared with the standard Faster R-CNN, three tricks are utilized to improve the detector. (1) **Classification and regression decoupling**: The original R-CNN contains two shared fully connected (fc) layer and two separate fc layers for classification and regression. In order to prevent the classification task from disturbing the regression task, the shared fc layers are replaced by two parallel fc layers. Simultaneously, the class-specific box regression is changed to class-agnostic. For example, the standard output of the box predictor is $4 \times (K + 2)$, now we set it as 4, where $K + 2$ denotes K known classes, 1 unknown class, and 1 background class. It means that for each region proposal, we predict one box for all classes, instead of one box per class. The above operations are utilized to decouple the classification and the regression, and then provide convenience to tackle the few-shot open-set object detection task.

(2) **Two-stage fine-tuning strategy**: Following TFA [3], the training process consists of a base training stage and a few-shot fine-tuning stage. In the base training stage, we employ the abundant samples of the base classes C_B to train the entire base detector from scratch, such as Faster R-CNN. Then, in few-shot fine-tuning stage, we first train the last linear layers of the base detector while freezing the other parameters of the model (linear probing [34]) on a small balanced training set consisting of both base and novel classes (C_B+C_N), and then fine-tune all the parameters of the model (fine-tuning) in a soft-freezing way [10], which employs a scaled gradient to slowly update the parameters of the backbone network to get the few-shot open-set object detector.

(3) **Classifier placeholder**: We can view the classifier placeholder [25] as the dummy class. In base training stage, we reserve the dummy classifier placeholders for the novel classes C_N and an unknown class C_U to augment the class number of the open-set classifier. These placeholders reserved for the novel classes will be optimized in the few-shot fine-tuning stage. Overall, there are two advantages for the predefined classifier placeholder: one is that the classifier placeholder omits the additional model surgery step [3], [10], which is used to augment the number of model categories from the base training stage (base classes C_B) to the few-shot fine-tuning stage (base classes C_B + novel classes C_N) for the close-set classifier. This means that our method simplifies the training process of the FSOD methods based on transfer learning. Another is that the dummy sub-classifier for the unknown class is necessary to optimize our proposed unknown decoupling learner (UDL), which can identify the unknown objects without relying on the pseudo-unknown samples for training.

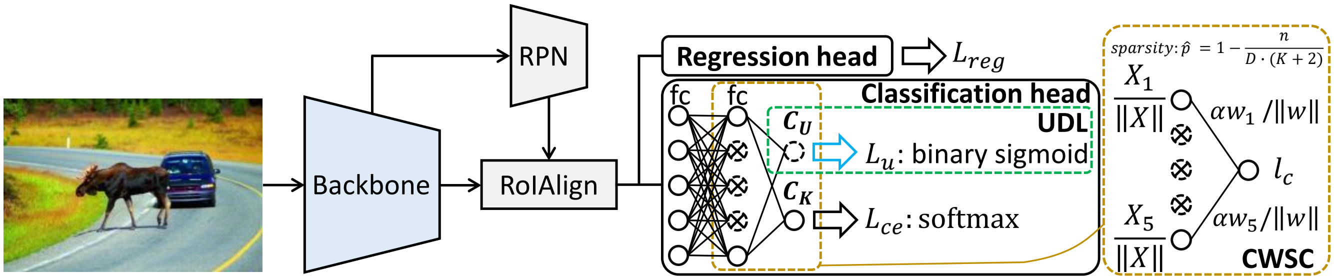

Fig. 2. The framework of our FOOD for few-shot open-set object detection. Compared to the standard Faster R-CNN, FOOD plugs a novel class weight sparsification classifier (CWSC) and a novel unknown decoupling learner (UDL). We sparsity the normalized weights for the class logit prediction and simultaneously optimize a binary sigmoid classifier and a multiply softmax classifier in the classification head. Our method is characterized by no pseudo-unknown sample generation, prototype-free, and threshold-free to detect unknowns in few-shot scenes.

C. Class Weight Sparsification Classifier (CWSC)

Few training samples induce the model to overfit on the known classes, thus the model cannot extract the generalization features that can be used for the unknown object detection. The neuron dropout theory [27] has been employed to suppress the over-fitting issue. The dropout randomly drops some weights during training, but there is no way to randomly drop them during prediction. This means that all weights will be reserved for prediction at test time, which results in a large expected (or mean) difference of the output between training (with dropout) and testing (without dropout), causing unstable results. So we need to make the output expected value of training and testing as consistent as possible. Our CWSC can be understood as class weight sparsification for unknown detection in few-shot scenes. As shown in Fig. 2, our CWSC sparses the normalized weights $w/||w||$ rather than w and measure the cosine similarity with the normalized feature $X/||X||$ for the class logit prediction, where $||\cdot||$ denotes the L2 norm. This makes the expected difference with and without sparsification become bounded, and then the model is more stable for open-set detection in few-shot scenes.

Specifically, $w \in \mathbb{R}^{D\times(K+2)}$ represents the weights of the last linear mapping to $(K+2)$ classes, where D denotes the weight dimension for each class. We use the sparsity probability $\widehat{p} = 1 - \frac{n}{D\cdot(K+2)}$, where n denotes the number of random selected weights for the class logit prediction. Then, the retention probability of the weight can be denoted as $p = 1 - \widehat{p}$. The weights can be denoted as $R * w/||w||$, where $R \in \{0,1\}^{D\times(K+2)}$ is initialized with $R_{i,j} \sim \text{Bernoulli}(p)$ and $*$ represents the element-wise product. The total number of 1 in R is n, where 1 denotes retain the weight and 0 denotes drop the weight. The class logit output could be expressed as:

$$l_c = \alpha \frac{X}{||X||}\left[R * \frac{w}{||w||}\right], \tag{1}$$

where α denotes a positive temperature factor. In our CWSC, if we set the retention probability p, the expected value (or mean) of the logit output l_c with sparsification falls in $[-p\alpha, p\alpha]$, without sparsification, it falls in $[-\alpha, \alpha]$, thus the expected difference with and without sparsification is $\mu^{cos}_{diff} \in [-(1+p)\alpha, (1+p)\alpha]$, which is boundary. For the conventional output $\widehat{l_c} = Xw$, the expected difference of the logit output with and without sparsification is $\mu_{diff} \in (-\infty, +\infty)$. Compared with μ_{diff}, μ^{cos}_{diff} is bounded, that's means the output distribution of our CWSC with and without sparsification is more consistent, producing more stable results for few-shot open-set detection.

Next, we theoretically analyze why the CWSC can suppress the over-fitting issue. Assuming a linear regression task, $y \in \mathbb{R}^N$ is the ground truth label, the model tries to find a $w \in \mathbb{R}^D$ to minimize

$$||y - \alpha \frac{X}{||X||}\frac{w}{||w||}||^2. \tag{2}$$

We set $\bar{x} = X/||X||$ and $\bar{w} = w/||w||$. When the weight sparsification is adopted, the objective function becomes

$$\underset{w}{minimize}\ \mathbb{E}_{R\sim\text{Bernoulli}(p)}[||y - \alpha\bar{x}(R * \bar{w})||^2]. \tag{3}$$

This can reduce to

$$\underset{w}{minimize}||y - \alpha p\bar{x}\bar{w}||^2 + \alpha^2 p(1-p)||\tau\bar{w}||^2, \tag{4}$$

where $\tau = (diag(\bar{x}^{\mathrm{T}}\bar{x}))^{1/2}$. We set $\widetilde{w} = \alpha p\bar{w}$, then

$$\underset{w}{minimize}||y - \bar{x}\widetilde{w}||^2 + \frac{(1-p)}{p}||\tau\widetilde{w}||^2. \tag{5}$$

Eq. 5 can be viewed as a ridge regression with a particular form for τ. If a particular data dimension changes a lot, the regularizer tries to sparse its weight more [27], and then the over-fitting problem is alleviated through regularization. We can control the strength of the regularization by adjusting the sparsity probability $\widehat{p} = 1 - p$. For example, if we set $\widehat{p} = 0$, then $p = 1$, the regularization term is 0, which means that the regularizer does not work. As the sparsity probability $\widehat{p}$ increases, the regularization constant grows larger and the regularization effect becomes more pronounced. Our method benefits from the over-fitting suppression, and we verify that the class weight sparsification greatly improves the model's generalization ability for the open-set detection in few-shot scenes. A derivation of Eq. 2-5 is presented in Appendix.

D. Unknown Decoupling Learner (UDL)

Unknown decoupling learner (UDL) plays a decisive role to detect the unknown class, which provides a dummy unknown class for decoupling optimization, and then boosts the model to form a compact unknown decision boundary. The UDL does not depend on the pseudo-unknown samples generation method [18] to train the dummy unknown class, because the data distribution of the unknown class is more complex and changeable, the generated fake unknown samples often fail to simulate the real distribution of the unknown data. Inspired by the fact that the known class data and the unknown class data are often orthogonal [34], we propose to decouple optimize the unknown class without relying on the predictions of the known classes. Thus, we select a sigmoid function that normalizes the predicted unknown logit to estimate the unknown probability:

$$p_u(l_{C_U}) = \frac{1}{1+e^{-\delta \cdot l_{C_U}}}, \tag{6}$$

where l_{C_U} represents the predicted logit for the dummy unknown class C_U in the classification branch and δ is used to adjust the slope of the sigmoid function. Why do we choose the sigmoid function to compute the unknown probability p_u instead of softmax? If we select the softmax, the objective function for the unknown branch becomes:

$$-\log p_u\uparrow = -\log \frac{e^{l_{C_U}}\uparrow}{\sum\limits_{c_i \in C, c_i \neq c_*} e^{l_{c_i}} + e^{l_{c_*}}\uparrow} \tag{7}$$

where $C = C_B \cup C_N \cup C_U \cup C_{bg}$, and $\uparrow$ denotes that the optimized target should be as large as possible in optimization. It is not hard to see that the optimization direction of the dummy unknown class C_U and the ground truth known class c_* is conflict, which makes it difficult for the unknown class to be converged. While the unknown probability product by the sigmoid function does not care about the logit outputs for other classes, we call it unknown decouple training, which can alleviate the above problem.

Therefore, the loss function of the dummy unknown class (unknown decoupling loss) is defined as follows:

$$\begin{aligned}\mathcal{L}_u = &\frac{1}{N_{pos}} \sum_{i=1}^{N_{pos}} \log(1+e^{-\delta_1 \cdot l_{C_U}(\mathcal{S}^i_{pos})}) \\ &+\frac{1}{N_{neg}} \sum_{j=1}^{N_{neg}} \log(1+e^{-\delta_2 \cdot l_{C_U}(\mathcal{S}^j_{neg})}),\end{aligned} \tag{8}$$

where N_{pos} and N_{neg} represent the number of positive samples (foreground of known classes) $\mathcal{S}_{pos}$ and the number of negative samples (background) $\mathcal{S}_{neg}$ respectively. Noting that the negative samples serve as a very important role in the loss optimization, because the model is also required to identify the unknown class and the background. How do we select the positive and negative samples from all foreground proposals $\mathcal{S}_{fg}$ and background proposals $\mathcal{S}_{bg}$ to train the UDL branch? Energy score [32], [35] has been used to measure the uncertainty of the object. The lager the energy, the higher the uncertainty of the sample [32]. Here, we propose a conditional energy score to select the positive and negative samples with high uncertainty from the proposals of the known classes and the background, respectively, to train the UDL branch. For a region proposal $\mathcal{S}_j \in \mathcal{S}_{fg} \cup \mathcal{S}_{bg}$, the conditional energy score is defined as:

$$E(\mathcal{S}_j)_{c_i \neq C_U} = -\log \sum_{c_i \neq C_U, c_i \in C} e^{l_{c_i}(\mathcal{S}_j)}, \tag{9}$$

where $l_{c_i}(\cdot)$ is the predicted logit for class c_i. Since there are no real unknown class training samples, the term $exp(l_{C_U}(\mathcal{S}_j))$ will become an interference term in the energy-based sampling process. Thus, we discard it and select the top-k samples ranked by the conditional energy score $E(\mathcal{S}_j)_{c_i \neq C_U}$ to optimize the unknown decoupling loss L_u. The positive and negative samples ($\mathcal{S}_{pos}$ and $\mathcal{S}_{neg}$) used to train the UDL branch can be denoted as:

$$\mathcal{S}_{pos} = \underset{\mathcal{S}_{j'} \in \mathcal{S}_{fg}}{topk} \left(-\log \sum_{c_i \neq C_U, c_i \in C} e^{l_{c_i}(\mathcal{S}_{j'})} \right)_{k=N_{pos}}, \tag{10}$$

$$\mathcal{S}_{neg} = \underset{\mathcal{S}_{j''} \in \mathcal{S}_{bg}}{topk} \left(-\log \sum_{c_i \neq C_U, c_i \in C} e^{l_{c_i}(\mathcal{S}_{j''})} \right)_{k=N_{neg}}. \tag{11}$$

The sub-classifier of the dummy unknown class is equivalent to merging the binary sigmoid classifier of the dummy unknown class into the multiply softmax classifier of the real known classes. Then, the model can synchronously identify the specific class of the known and unknown objects in inference. Instead of asynchronously distinguishing the unknown class from the known classes like the threshold-based methods [15], [26], if it is not an unknown class, the model would distinguish its specific known class.

E. Overall Optimization

With the unknown decoupling loss L_u, the final loss function is defined as follows:

$$\mathcal{L} = \mathcal{L}_{RPN} + \mathcal{L}_{reg} + \mathcal{L}_{ce}(D_B \cup D_N) + \lambda \mathcal{L}_u(\mathcal{S}_{pos} \cup \mathcal{S}_{neg}), \tag{12}$$

where $\mathcal{L}_{RPN}$ is the RPN loss that consists of a binary cross entropy loss and a regression loss. $\mathcal{L}_{reg}$ denotes the smooth L1 loss for R-CNN. $\mathcal{L}_{ce}$ expresses the cross-entropy loss for R-CNN. λ is used to balance the proportion of the unknown decoupling loss $\mathcal{L}_u$. The above losses can be used to jointly optimize the FSOSOD model f_θ as:

$$\begin{aligned}\underset{\theta}{argmin}(&\mathbb{E}_{(x,y)\in D_B \cup D_N} \mathcal{L}_{RPN,reg,ce}(x, y; f_\theta) \\ &+\mathbb{E}_{(x,y)\in \mathcal{S}_{pos} \cup \mathcal{S}_{neg}} \mathcal{L}_u(x, y; f_\theta)),\end{aligned} \tag{13}$$

where θ denotes the learned weights for the FSOSOD model.

F. Inference

During inference, we normalize the logits of all classes (base classes, novel classes, the dummy unknown class, and the background class) by the softmax function to get the final classification score:

$$p_{c_m} = \underset{c_j \in C}{max}\left(\frac{e^{l_{c_j}}}{\sum\limits_{c_i \in C = C_B \cup C_N \cup C_U \cup C_{bg}} e^{l_{c_i}}}\right). \tag{14}$$

The specific class of the predicted box is c_m. We not continue to choose sigmoid to compute the probability of the unknown class p_u in inference. The reason is that if we choose sigmoid to compute p_u at test time, our method would become a threshold-based method to divide the unknown class, which means we need set a threshold to identity the unknown class. However, softmax is a threshold-free method during test, the class with the maximum softmax probability (MSP) is the final predicted class of the box. Thus, we select it and our method becomes threshold-free for unknown detection based on few training samples.

IV. Experiment

A. Experimental Setup

1) Datasets: We construct the FSOSOD benchmarks using PASCAL VOC 2007+2012 [36] and MS COCO 2017 [37]. There are two types of benchmark settings. One is the single-dataset benchmark: VOC10-5-5, which means only one dataset (PASCAL VOC) is used to construct the FSOSOD benchmark. Another is the cross-dataset benchmark: VOC-COCO, which means two datasets (PASCAL VOC and MS COCO) are used to construct the FSOSOD benchmark. The VOC-COCO setting can ensure that the model barely sees the unknown classes during training.

VOC10-5-5: The 20 classes of PASCAL VOC are divided into 10 base classes C_B, 5 novel classes C_N, and 5 unknown classes C_U to evaluate the FSOSOD performance of our method. The novel classes C_N have M =1, 2, 3, 5, 10, and 30 objects per class sampled from the training data of PASCAL VOC. Here we select the test set of PASCAL VOC 2007 for the few-shot open-set evaluation. In this paper, we define the above dataset division as the **VOC10-5-5** setting, where C_B={aeroplane, bicycle, bird, boat, bottle, bus, car, cat, chair, cow}, C_N={diningtable, dog, horse, motorbike, person}, C_U={pottedplant, sheep, sofa, train, tvmonitor}={unknown}.

VOC-COCO: The MS COCO dataset contains 80 classes, 20 of which overlap with the PASCAL VOC dataset. We use 20 classes of PASCAL VOC and 20 non-VOC classes of MS COCO as the close-set training data, where PASCAL VOC servers as the base classes C_B and the 20 non-VOC classes of MS COCO are the few-shot splits of novel classes C_N. The novel classes C_N have M =1, 2, 3, 5, 10, and 30 objects per class sampled from the training data of MS COCO. The remaining 40 classes of MS COCO are used as the unknown classes C_U, which are challenging. Meanwhile, we use the validation set of MS COCO for the few-shot open-set evaluation. In this paper, we define the above dataset division as the **VOC-COCO** setting, where C_B={aeroplane, bicycle, bird, boat, bottle, bus, car, cat, chair, cow, diningtable, dog, horse, motorbike, person, pottedplant, sheep, sofa, train, tvmonitor}, C_N={truck, traffic light, fire hydrant, stop sign, parking meter, bench, elephant, bear, zebra, giraffe, backpack, umbrella, handbag, tie, suitcase, microwave, oven, toaster, sink, refrigerator}, C_U={frisbee, skis, snowboard, sports ball, kite, baseball bat, baseball glove, skateboard, surfboard, tennis racket, banana, apple, sandwich, orange, broccoli, carrot, hot dog, pizza, donut, cake, bed, toilet, laptop, mouse, remote, keyboard, cell phone, book, clock, vase, scissors, teddy bear, hair drier, toothbrush, wine glass, cup, fork, knife, spoon, bowl}={unknown}.

2) Evaluation Metrics: The mean average precision (mAP) of known classes (mAP_K) is chosen to evaluate the known object detection performance. mAP_B and mAP_N are used to measure the performance for base and novel classes, respectively. To evaluate the unknown detection performance, the average precision (AP_U), precision (P_U), and recall (R_U) are reported. The unknown recall (R_U) is a popular metric that is currently concerned by the unknown detection [38], which is defined as:

$$R_U = \frac{True\ positives\ of\ unknown}{All\ ground\ truth\ unkonwn\ objects}. \tag{15}$$

3) Implementation Details: Our base detector is Faster R-CNN and ResNet-50 with feature pyramid network (FPN) [39] is selected as the backbone. All models are trained using SGD optimizer with a mini-batch size of 16, a momentum of 0.9, and a weight decay of 1e-4. The learning rate of 0.02 is used in the base training stage and 0.01 in the few-shot fine-tuning stage. For the CWSC, we set the temperature factor $\alpha = 20$, the sparsity probability $\widehat{p} = 0.6$, and the class weight dimension $D = 2048$. For the UDL, we use a slope factor $\delta_1 = \delta_2 = 0.09$ and $N_{pos} : N_{neg} = 3 : 12$. In total loss, we set the trade-off factor $\lambda = 1.0$. Note that both the base training stage and the few-shot fine-tuning stage need to optimize the UDL branch. Then we argue that the generalization knowledge learned by the base training stage between the known and unknown classes can be reserved for the few-shot fine-tuning stage. CWSC is only used in the fine-tuning stage.

4) Baselines: We compare our proposed FOOD with several OSOD methods: OpenDet [14], DS [15], and PROSER [25] combined with TFA [3] for FSOSOD. We also present the FSOD results of TFA as a baseline to determine whether optimizing the dummy unknown class will reduce the performance of the known classes. Moreover, all methods employ the same ResNet-50 with FPN as the backbone for a fair comparison and we report the average results of 10 random runs for all comparison methods.

B. Results

*1) **VOC10-5-5**:* In Table I, we compare our FOOD with several OSOD methods combined with TFA on VOC10-5-5 dataset setting. Our FOOD outperforms other methods across all shots on the unknown metrics AP_U and R_U. We achieve 0.65∼1.35 point improvement in AP_U over the best comparison method and around 5.89∼14.57 point improvement in R_U. Simultaneously, our method outperforms other methods on the mAP_N of the novel classes, which demonstrates its effectiveness for FSOSOD. The unknown recall of our FOOD achieves a significant improvement than the second best, which verifies that our method alleviates the over-fitting issue and evidently improves the model's generalization ability for unknown detection in few-shot scenes. However, in VOC10-5-5 settings, the detector may have seen unknown objects and treated them as background during training, which causes the evaluation bias for FSOSOD. To balance the above bias, we conduct experiments on VOC-COCO dataset setting, where the unknown classes are barely seen in the training data.

TABLE I

The Generalized Few-Shot Open-Set Object Detection Results on **VOC10-5-5** Dataset Setting. ↑ Indicates That the Larger the Evaluation Metrics, the Better the Performance. **Bold** Numbers denote Superior Results, and Blue Background Represents a Significant Improvement Over the Second Best. For a Fair Comparison, We Report the Average Results of 10 Random Runs for All Comparison Methods.

VOC10-5-5	Backbone	1-shot						2-shot					
		mAP_K↑	mAP_B↑	mAP_N↑	AP_U↑	P_U↑	R_U↑	mAP_K	mAP_B	mAP_N	AP_U	P_U	R_U
TFA [3]		**45.31**	**63.71**	8.50	0.00	0.00	0.00	**46.48**	62.55	14.34	0.00	0.00	0.00
DS [15]+TFA		43.82	62.11	7.22	1.90	**7.61**	23.99	46.28	**63.20**	12.44	2.08	**8.20**	24.56
PROSER [25]+TFA	ResNet-50	41.64	58.22	8.49	3.26	5.60	30.95	42.70	57.27	13.56	3.42	5.35	31.53
OpenDet [14]+TFA		43.45	61.04	8.27	3.44	6.62	33.64	45.67	62.74	11.53	3.28	6.85	30.95
Our FOOD		43.97	61.48	**8.95**	**4.26**	3.58	**43.72**	45.85	61.37	**14.80**	**4.61**	3.51	**45.52**
VOC10-5-5		3-shot						5-shot					
		mAP_K	mAP_B	mAP_N	AP_U	P_U	R_U	mAP_K	mAP_B	mAP_N	AP_U	P_U	R_U
TFA [3]		47.55	63.72	15.23	0.00	0.00	0.00	47.88	61.92	19.74	0.00	0.00	0.00
DS [15]+TFA		46.89	63.09	14.48	2.11	**8.32**	23.62	48.01	62.38	19.27	1.91	**8.85**	19.99
PROSER [25]+TFA	ResNet-50	43.30	57.30	15.16	3.23	5.32	32.30	45.12	57.64	20.08	3.56	5.34	32.68
OpenDet [14]+TFA		46.47	62.66	14.09	3.15	6.47	30.62	47.56	62.39	17.90	3.41	6.89	32.13
Our FOOD		**48.48**	**64.30**	**16.83**	**4.50**	3.54	**44.52**	**50.18**	**63.72**	**23.10**	**4.63**	3.51	**45.65**
VOC10-5-5		10-shot						30-shot					
		mAP_K	mAP_B	mAP_N	AP_U	P_U	R_U	mAP_K	mAP_B	mAP_N	AP_U	P_U	R_U
TFA [3]		51.10	63.56	26.19	0.00	0.00	0.00	56.20	65.68	37.23	0.00	0.00	0.00
DS [15]+TFA		48.01	62.38	25.66	1.91	**8.85**	19.99	56.60	66.27	36.96	1.90	**9.95**	18.02
PROSER [25]+TFA	ResNet-50	48.35	59.96	25.13	3.59	5.16	32.61	53.93	62.96	35.86	3.46	4.76	33.93
OpenDet [14]+TFA		50.95	63.85	25.14	4.06	6.61	36.30	56.11	66.05	36.24	4.03	5.62	40.51
Our FOOD		**53.23**	**65.54**	**28.60**	**4.71**	3.59	**45.84**	**58.59**	**67.73**	**40.29**	**4.90**	3.62	**46.40**

*2) **VOC-COCO**:* In Table II, we carry out experiments on VOC-COCO dataset setting. Only look at the detection performance of the unknown class (AP_U, P_U, and R_U), our method presents several significant advantages, especially in unknown recall R_U. For example, the highest recall 23.17% of the unknown class is obtained by our FOOD, which illustrates that our method has the better unknown object detection ability than other methods. Simultaneously, in extremely low shots (1, 2, 3, 5-shot) where the over-fitting problem is easier to occur than high shots (10, 30-shot), our method achieves 8.23%, 9.45%, 9.37%, and 8.53% unknown recall R_U improvements than other best method respectively, which demonstrates the effectiveness of our method in suppressing the over-fitting issue caused by few training samples of known classes.

Compared with the baseline TFA on novel classes, the mAP_N of our method is similar to TFA. Meanwhile, the mAP_K and mAP_B of our method are higher than TFA, which verifies that our method not only improves the open-set detection performance of unknown class, but also slightly improves the performance of known classes $C_K = C_B \cup C_N$ for the close-set few-shot evaluation. Look at other methods, although DS+TFA, PROSER+TFA, and OpenDet+TFA achieve comparable close-set metrics (mAP_K, mAP_B, and mAP_N), the open-set performance is poor. Overall, the proposed FOOD surpasses other methods and thus is of merit. The class weight sparsification and the unknown decoupling training effectively expand model's generalization capability for open-set detection in few-shot conditions, which brings the performance improvements.

C. Ablation Studies

We conduct the comprehensive ablation studies on 10-shot setting of VOC-COCO dataset setting.

1) Effectiveness of different modules: We perform the ablation studies on different modules (CWSC and UDL) in Table III, where the classical FSOD framework TFA [3] is used as the baseline. It can be seen that CWSC boosts the detection performance of the base classes (AP_B) and the novel classes (AP_N) simultaneously, which demonstrates the effectiveness of CWSC to alleviate the over-fitting problem. When exploring the influence of UDL, we find that the branch of UDL improves the detection ability for the close-set known classes in 10-shot setting. Simultaneously, UDL is a necessary condition for the unknown object detection in our FOOD. The best unknown detection result $AP_U = 3.27\%$ is obtained by the cooperation of two modules, although the detection performance of known classes is slightly degraded compared to the UDL alone (↓ 0.10%), this is acceptable in terms of the overall results. Compared with the third line (only UDL) and the fourth line (CWSC+UDL), the unknown recall improves 4.79%, which demonstrates that the unknown detection performance benefits from our CWSC. The class weight sparsification significantly enhances the model's generalization ability for unknown detection in few-shot scenes.

2) Effectiveness of different fine-tuning methods: In Table IV, we carefully evaluate several fine-tuning methods to pick the most appropriate way. The linear-probing (LP) and fine-tuning (FT) [34] have been widely used in transfer learning to alleviate the over-fitting problem. Gradient decoupled layer (GDL) [10] is an auxiliary fine-tuning strategy, which conducts stop-gradient for RPN and scale-gradient (scale=0.001 in this paper) for RCNN. As a **hard-freezing** method, LP can preserve the general knowledge of the base training stage, thus it achieves a competitive detection result in Table IV. However, FT fine-tunes the entire model to fit the few-shot close-set training data. The model gradually forgets the base

TABLE II
THE GENERALIZED FEW-SHOT OPEN-SET OBJECT DETECTION RESULTS ON **VOC-COCO** DATASET SETTING. ↑ INDICATES THAT THE LARGER THE EVALUATION METRICS, THE BETTER THE PERFORMANCE. **BOLD** NUMBERS DENOTE SUPERIOR RESULTS, AND BLUE BACKGROUND REPRESENTS A SIGNIFICANT IMPROVEMENT OVER THE SECOND BEST. FOR A FAIR COMPARISON, WE REPORT THE AVERAGE RESULTS OF 10 RANDOM RUNS FOR ALL COMPARISON METHODS.

VOC-COCO	Backbone	1-shot						2-shot					
		mAP_K↑	mAP_B↑	mAP_N↑	AP_U↑	P_U↑	R_U↑	mAP_K	mAP_B	mAP_N	AP_U	P_U	R_U
TFA [3]	ResNet-50	15.77	29.03	**2.50**	0	0	0	15.82	28.01	**3.63**	0	0	0
DS [15]+TFA		15.47	28.84	2.11	0.48	**11.53**	3.57	16.28	29.36	3.21	0.51	**11.46**	3.77
PROSER [25]+TFA		13.58	24.84	2.32	0.87	7.14	7.53	14.27	24.91	3.62	0.95	6.79	8.66
OpenDet [14]+TFA		**16.01**	**29.72**	2.29	0.86	7.20	7.24	16.39	29.52	3.27	0.96	6.84	8.97
Our FOOD		15.83	29.32	2.35	**2.00**	5.75	**15.76**	**16.56**	**29.55**	3.58	**2.42**	5.68	**18.42**
VOC-COCO		3-shot						5-shot					
		mAP_K	mAP_B	mAP_N	AP_U	P_U	R_U	mAP_K	mAP_B	mAP_N	AP_U	P_U	R_U
TFA [3]	ResNet-50	16.55	28.27	**4.81**	0	0	0	17.13	27.71	6.56	0	0	0
DS [15]+TFA		16.93	29.38	4.49	0.54	**12.47**	3.95	17.10	27.91	6.30	0.57	**14.15**	3.86
PROSER [25]+TFA		15.07	25.53	4.62	1.21	7.27	9.20	15.67	26.95	6.40	1.30	7.65	9.59
OpenDet [14]+TFA		16.82	29.11	4.55	1.16	7.39	9.56	17.16	27.75	6.56	1.48	7.84	11.49
Our FOOD		**17.56**	**30.57**	4.56	**2.72**	6.18	**18.93**	**18.08**	**29.47**	**6.69**	**2.92**	6.06	**20.02**
VOC-COCO		10-shot						30-shot					
		mAP_K	mAP_B	mAP_N	AP_U	P_U	R_U	mAP_K	mAP_B	mAP_N	AP_U	P_U	R_U
TFA [3]	ResNet-50	18.67	28.32	9.02	0	0	0	23.10	31.03	15.16	0	0	0
DS [15]+TFA		19.06	28.66	9.46	0.60	**15.41**	3.75	23.40	31.53	**15.27**	0.63	**15.58**	3.95
PROSER [25]+TFA		17.00	25.24	8.75	1.36	7.87	10.06	21.44	28.58	14.30	1.52	7.31	12.06
OpenDet [14]+TFA		18.53	28.36	8.70	1.82	7.38	13.89	22.93	31.61	14.02	2.64	7.52	18.07
Our FOOD		**20.17**	**30.67**	**9.48**	**3.27**	6.48	**21.48**	**23.90**	**33.32**	14.47	**3.80**	6.68	**23.17**

TABLE III
THE ABLATION STUDY OF DIFFERENT MODULES (AVERAGE OF 10 RANDOM RUNS). **BOLD** NUMBERS DENOTE SUPERIOR RESULTS.

CWSC	UDL	mAP_K	mAP_B	mAP_N	AP_U	P_U	R_U
Baseline (TFA)		18.67	28.32	9.02	0	0	0
✓		19.20	28.97	9.44	0	0	0
	✓	**20.27**	**30.96**	**9.58**	2.35	**6.59**	16.69
✓	✓	20.17	30.67	9.48	**3.27**	6.48	**21.48**

TABLE IV
THE ABLATION STUDY OF DIFFERENT FINE-TUNING METHODS (AVERAGE OF 10 RANDOM RUNS). **BOLD** NUMBERS DENOTE SUPERIOR RESULTS.

Fine-tuning methods	mAP_K	mAP_B	mAP_N	AP_U	P_U	R_U
LP	19.89	30.55	9.22	3.14	6.57	20.91
FT	15.31	20.59	**10.03**	0.78	**9.10**	5.17
FT+GDL	20.10	30.42	9.78	3.20	6.51	21.30
LP-FT	19.87	30.35	9.39	3.22	6.45	21.33
LP-FT+GDL	**20.17**	**30.67**	9.48	**3.27**	6.48	**21.48**

TABLE V
THE ABLATION STUDY ON DIFFERENT SAMPLING METHODS (AVERAGE OF 10 RANDOM RUNS). **BOLD** NUMBERS DENOTE SUPERIOR RESULTS.

Simpling methods	mAP_K	mAP_B	mAP_N	AP_U	P_U	R_U
$Min\ max\text{-}probability$ [14]	19.85	30.30	9.30	3.05	6.38	20.80
$Min(l_{C_U})$	19.53	29.39	**9.66**	2.73	**7.04**	17.55
$Max(entropy)$	19.01	29.17	9.41	2.79	6.29	18.53
$Max(E(x,b))$	19.94	30.57	9.31	3.08	6.42	20.89
$Max(E(x,b)_{c_i \neq C_U})$	**20.17**	**30.67**	9.48	**3.27**	6.48	**21.48**

classes and destroys the general knowledge during few-shot fine-tuning stage. Thus, we can see that compared FT with LP, the results of the base classes degrade 9.96%, and the unknown class descends 2.36%. The advantage of FT is the better performance of the novel classes mAP_N than other approaches. LP-FT exploits the advantages of LP and FT, thus it achieves a compromise FSOSOD result.

Compared with FT, FT+GDL achieves better results in mAP_K and AP_U. This indicates that GDL reserves the general knowledge by the gradient decoupling training, which uses a scale-gradient to slowly update the parameters of the backbone network. We view the GDL as a **soft-freezing** method, which enables the backbone to slowly fit the close-set data while retaining the ability to extract the generalization features. GDL improves the performance of the close-set object detection and achieves a competitive unknown object detection performance. Based on the above analysis, we adopt a **hard-soft** combination approach (LP-FT+GDL), which first trains the last linear layers of the model while freezing other parameters. And then we fine-tunes all the model via a soft-freezing way. As illustrated in Table IV, LP-FT+GDL outperforms other approaches in terms of mAP_K, mAP_B, AP_U, and R_U, which verifies its effectiveness.

3) Effectiveness of different sampling methods: The sampling rules for the positive and negative samples to train the UDL branch are as follows:

- $Min\ max\text{-}probability$ [14]: the samples are sorted in ascending order by the maximum predicted value across all classes, and top-k samples are chosen.
- $Min(l_{C_U})$: the samples are sorted in ascending order by the unknown logit, and top-k samples are chosen.
- $Max(entropy)$: the samples are sorted in descending order by the entropy score, and top-k samples are chosen.
- $Max(E(x,b)_{c_i \neq C_U})$: the samples are sorted in descending order by the conditional energy score. We select top-k samples for optimization.

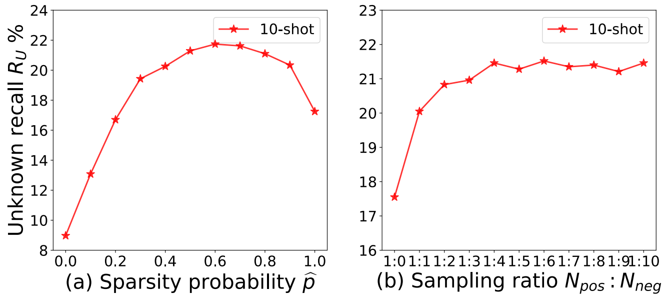

Fig. 3. Effect of different sparsity probability $\widehat{p}$ and sampling ratios N_{pos} : N_{neg} on 10-shot VOC-COCO dataset setting.

TABLE VI
THE ABLATION STUDY OF DIFFERENT BACKBONES (AVERAGE OF 10 RANDOM RUNS). **BOLD** NUMBERS DENOTE SUPERIOR RESULTS.

Backbones	mAP_K	mAP_B	mAP_N	AP_U	P_U	R_U
ResNet-50	20.17	30.67	9.48	3.27	**6.48**	21.48
ResNet-101	20.84	31.48	10.00	3.32	6.02	22.37
Swin-T	**25.64**	**40.49**	**10.80**	**3.41**	6.13	**22.93**

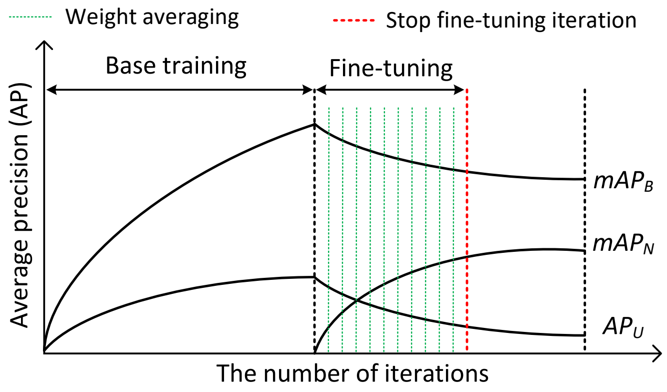

Fig. 4. The relationship between iteration and performance metric (mAP_B, mAP_N, and AP_U) for our FSOSOD method. We are hard to select a proper stop fine-tuning iteration to balance the performance of base classes, novel classes, and the unknown class.

TABLE VII
THE PERFORMANCE OF WEIGHT AVERAGING (WA) (AVERAGE OF 10 RANDOM RUNS). **BOLD** NUMBERS DENOTE SUPERIOR RESULTS, AND BLUE BACKGROUND REPRESENTS A SIGNIFICANT IMPROVEMENT OVER THE SECOND BEST.

VOC-COCO	mAP_K	mAP_B	mAP_N	AP_U	P_U	R_U
Our FOOD	20.17	30.67	**9.48**	3.27	6.48	21.48
Our FOOD+WA [42]	**23.32**	**37.89**	8.76	**4.13**	**6.82**	**24.87**

As presented in Table V, the conditional energy-based sampling method $Max(E(x, b)_{c_i \neq C_U})$ outperforms other methods, which demonstrates its effectiveness for the positive and negative sample selection. Furthermore, the absence of real unknown training samples causes the unknown class C_U to become a distractor, thus the results of our $Max(E(x, b)_{c_i \neq C_U})$ is better than $Max(E(x, b))$. Simultaneously, we can see that these energy-based sampling methods ($Max(E(x, b))$ and $Max(E(x, b)_{c_i \neq C_U})$) perform better than other sampling methods, which proves that the energy score is an excellent uncertainty metric for the positive and negative sample selection in the optimization of our UDL branch.

4) Sparsity probability and sampling ratio: Fig. 3 presents the visualization of different sparsity probabilities and sampling ratios on 10-shot VOC-COCO dataset setting. We fix the ratio of positive and negative samples $1:4$ to explore the performance of different sparsity probabilities. As presented in Fig. 3(a), when the sparsity probability is set to 0.6, the unknown recall R_U arrives at 22.74%, which is higher than other settings. Subsequently, we fix $\widehat{p} = 0.6$ to iterate over different sampling ratios. As we can see from the Fig. 3(b), starting from a sampling ratio of $1:4$, the value of unknown recall begins to be stabilized. Sampling ratio of $1:6$ seems to perform the best but our usual default value of $1:4$ is close to the optimal. Moreover, if the negative samples do not participate in the optimization of the UDL branch (1:0), the unknown detection performance drops significantly.

5) Effectiveness of different backbones: We use ResNet-101 [40] and swin transformer (Swin-T) [41] as the backbones in Table VI, and then compare them with ResNet-50. ResNet-101 tends to perform better than ResNet-50, which presents that our method benefits from the deeper backbone. While employing a more powerful transformer-based backbone (Swin-T), our proposed method achieves additional improvements.

6) Does the unknown-class placeholder hurt the accuracy of the few-shot object detection?: **No**. As illustrated in the first and third lines of Table III, when UDL branch introduces the unknown-class placeholder into the baseline TFA, the performance of the novel classes has achieved 0.56% improvement, which proves that the dummy placeholder of the unknown class does not hurt the detection performance for the few-shot novel classes.

7) Weight averaging: As shown in Fig. 4, the performance trends of the base classes AP_B and the unknown classes AP_U are conflict with the novel classes AP_N in the fine-tuning stage of our FOOD method. Therefore, it is difficult for us to choose a suitable stop fine-tuning iteration that can make the base, novel and unknown classes all perform best. Motivated by weight averaging (WA) [42] that is a simple ensemble method, but it achieves the state-of-the-art performance in domain generalization [43]–[45]. We present WA to approximate the optimal model. The WA is defined as:

$$\theta_{WA}(\mathcal{L}(D_{tr} \cup \mathcal{S}_{pos} \cup \mathcal{S}_{neg})) = \frac{1}{H+1}(\textstyle\sum_{h=1}^{H} \theta_h(\mathcal{L}(D_{tr} \cup \mathcal{S}_{pos} \cup \mathcal{S}_{neg})) + \theta_{final\ model}), \tag{16}$$

where $\{\theta_h\}_{h=1}^{H}$ represents the weights of equidistant dense sampling in a single run, $\theta_{final\ model}$ denotes the final output weights in the above single run. WA uses the idea of ensemble learning to balance the representation bias between the base, novel, and unknown classes. As illustrated in Table VII, when incorporating with WA (the model sampling step is 100 iterations), the evaluation metrics (mAP_K, mAP_B, and AP_U) of our FOOD achieve evidently improvements, which demonstrates its effectiveness. However, the drawback is that WA slightly decreases the performance of the novel

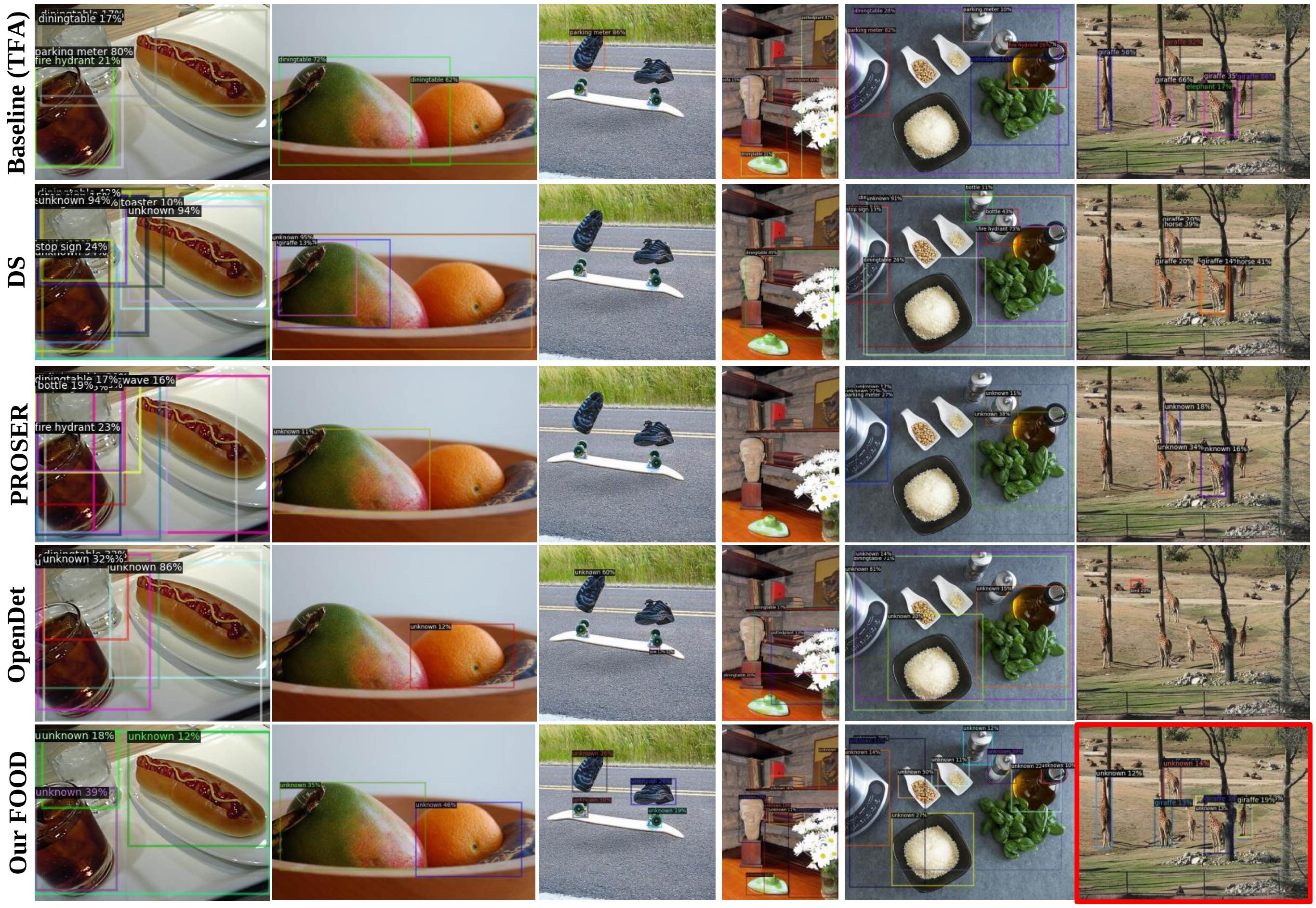

Fig. 5. The visualization results of open-set object detection in few-shot scenes (10-shot VOC-COCO setting). We visualize the bounding boxes with score larger than 0.1. Our FOOD can detect more the unknown objects than other methods. Red box is the failure case, several giraffes (novel class) are misidentified as the unknown class.

classes mAP_N. The main reason is that WA hurts the novel-class performance through the poor weight integration in low fine-tuning iterations. However, if you'd like to significantly improve the detection performance of the base classes and the unknown class at the expense of a little performance for the novel classes, WA is a good choice for FSOSOD.

D. Visualization

We provide qualitative visualizations of the detected unknown objects on 10-shot VOC-COCO dataset setting in Fig. 5. We can observe that other methods easily recognize the unknown objects as known classes and cannot detect the unknown objects frequently. Compared with other methods, our FOOD can detect more unknown objects than them, which demonstrates that our method benefits from the class weight sparsification and the unknown decoupling training. However, our method easily identifies the few-shot known objects as the unknown class. As shown in the red box of Fig. 5, several giraffes (novel class) are misidentified as the unknown class. Simultaneously, this phenomenon can also be seen from the quantitative analysis that our method shows slightly low unknown precision P_U, as illustrated in Table II. We view this situation as a limitation for the dummy class-based method, which needs to be further researched in the future.

V. Conclusion

In this paper, we propose a new task named few-shot open-set object detection (FSOSOD) and build the first benchmark. To tackle the challenging FSOSOD task, we propose a simple method incorporating several tricks in Faster R-CNN with two novel modules: class weight sparsification classifier (CWSC) and unknown decoupling learner (UDL). The CWSC is developed to decrease the co-adaptability between the class and its neighboring classes during training process, thereby improving the model's generalization ability for open-set detection in few-shot scenes. Alongside, the UDL branch is employed to detect the unknown objects in few-shot scenes without depending on the class prototype, threshold, and pseudo-unknown sample generation. Compared with other OSOD methods in few-shot scenes, our method achieves the state-of-the-art results on different shots settings of VOC10-5-5 and VOC-COCO datasets. In the future, we'd like to study the prompt learning [46] to solve the challenging FSOSOD task.

Appendix

We present a derivation of Eq. 2-5. Assuming a linear regression task, $y \in \mathbb{R}^N$ is the ground truth label, the model tries to find a $w \in \mathbb{R}^D$ to minimize

$$||y - \alpha \frac{X}{||X||}\frac{w}{||w||}||^2. \tag{17}$$

We set $\bar{x} = X/||X||$ and $\bar{w} = w/||w||$. When the weight sparsification is adopted, the objective function becomes

$$\begin{aligned} &\underset{w}{minimize}\ \mathbb{E}_{R\sim \mathrm{Bernoulli}(p)}[||y - \alpha \bar{x}(R * \bar{w})||^2] \\ &= \underset{w}{minimize}||y - \alpha p \bar{x}\bar{w}||^2 + \alpha^2(1-p)^2||\tau\bar{w}||^2 \\ &= \underset{w}{minimize}||y - \alpha p \bar{x}\bar{w}||^2 + \alpha^2(p^2-2p)||\tau\bar{w}||^2 + \alpha^2||\tau\bar{w}||^2, \end{aligned} \tag{18}$$

where $\tau = (diag(\bar{x}^{\mathrm{T}}\bar{x}))^{1/2}$. We view the above formula as a function of p. Then, this can reduce to

$$\underset{w}{minimize}||y - \alpha p \bar{x}\bar{w}||^2 + \alpha^2(p^2-2p)||\tau\bar{w}||^2. \tag{19}$$

This can reduce to

$$\underset{w}{minimize}||y - \alpha p \bar{x}\bar{w}||^2 - \alpha^2 p(2-p)||\tau\bar{w}||^2. \tag{20}$$

This can reduce to

$$\underset{w}{minimize}||y - \alpha p \bar{x}\bar{w}||^2 - 4\alpha^2\frac{p}{2}(1-\frac{p}{2})||\tau\bar{w}||^2. \tag{21}$$

This can reduce to

$$\underset{w}{minimize}||y - \alpha p \bar{x}\bar{w}||^2 + \alpha^2 p(1-p)||\tau\bar{w}||^2. \tag{22}$$

We set $\widetilde{w} = \alpha p\bar{w}$, then

$$\underset{w}{minimize}||y - \bar{x}\widetilde{w}||^2 + \underbrace{\frac{(1-p)}{p}||\tau\widetilde{w}||^2}_{Regularization\ term}. \tag{23}$$

References

[1] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2015, p. 91–99.

[2] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2016.

[3] X. Wang, T. E. Huang, T. Darrell, J. E. Gonzalez, and F. Yu, "Frustratingly simple few-shot object detection," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2020, pp. 9861–9870.

[4] B. Sun, B. Li, S. Cai, Y. Yuan, and C. Zhang, "Fsce: Few-shot object detection via contrastive proposal encoding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2021, pp. 7348–7358.

[5] H. Hu, S. Bai, A. Li, J. Cui, and L. Wang, "Dense relation distillation with context-aware aggregation for few-shot object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2021, p. 10185–10194.

[6] B. Li, B. Yang, C. Liu, F. Liu, R. Ji, and Q. Ye, "Beyond max-margin: Class margin equilibrium for few-shot object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2021, pp. 7359–7368.

[7] A. Wu, Y. Han, L. Zhu, and Y. Yang, "Universal-prototype enhancing for few-shot object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2021, p. 9567–9576.

[8] C. Zhu, F. Chen, U. Ahmed, Z. Shen, and M. Savvides, "Semantic relation reasoning for shot-stable few-shot object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2021, p. 8782–8791.

[9] Z. Fan, Y. Ma, Z. Li, and J. Sun, "Generalized few-shot object detection without forgetting," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2021, p. 4527–4536.

[10] L. Qiao, Y. Zhao, Z. Li, X. Qiu, J. Wu, and C. Zhang, "Defrcn: Decoupled faster r-cnn for few-shot object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2021, pp. 8661–8670.

[11] P. Kaul, W. Xie, and A. Zisserman, "Label, verify, correct: A simple few shot object detection method," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2022.

[12] L. Zhu, H. Fan, Y. Luo, M. Xu, and Y. Yang, "Few-shot common-object reasoning using common-centric localization network," *IEEE Trans. Image Process.*, vol. 30, pp. 4253–4262, 2021.

[13] B. Su, H. Zhang, Z. Wu, and Z. Zhou, "Fsrdd: An efficient few-shot detector for rare city road damage detection," *IEEE T. Intell. Transp.*, 2022.

[14] J. Han, Y. Ren, J. Ding, X. Pan, K. Yan, and G. Xia, "Expanding low-density latent regions for open-set object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2022.

[15] D. Miller, L. Nicholson, F. Dayoub, and N. Sunderhauf, "Dropout sampling for robust object detection in open-set conditions," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2018, pp. 3243–3249.

[16] M. Jeong, S. Choi, and C. Kim, "Few-shot open-set recognition by transformation consistency," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2021, pp. 12 561–12 570.

[17] C. Feng, Y. Zhong, and W. Huang, "Exploring classification equilibrium in long-tailed object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2021, pp. 3397–3406.

[18] B. Liu, H. Kang, H. Li, G. Hua, and N. Vasconcelos, "Few-shot open-set recognition using meta-learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2020, pp. 8795–8804.

[19] D. Pal, V. Bundele, R. Sharma, B. Banerjee, and Y. Jeppu, "Few-shot open-set recognition of hyperspectral images with outlier calibration network," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, 2022, pp. 3801–3810.

[20] S. Huang, J. Ma, G. Han, and S.-F. Chang, "Task-adaptive negative envision for few-shot open-set recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2022.

[21] K. Joseph, S. Khan, F. S. Khan, and V. N. Balasubramanian, "Towards open world object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2021, p. 5830–5840.

[22] Z. Wu, Y. Lu, X. Chen, Z. Wu, L. Kang, and J. Yu, "Uc-owod: Unknown-classified open world object detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2022.

[23] X. Zhao, X. Liu, Y. Shen, Y. Qiao, Y. Ma, and D. Wang, "Revisiting open world object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2022.

[24] J. H. L. P. N. B. Jiyang Zheng, Weihao Li, "Towards open-set object detection and discovery," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2022.

[25] D. Zhou, H. Ye, and D. Zhan, "Learning placeholders for open-set recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2021, p. 4401–4410.

[26] D. Miller, F. Dayoub, M. Milford, and N. Sunderhauf, "Evaluating merging strategies for sampling-based uncertainty techniques in object detection," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2019.

[27] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, pp. 1929–1958, 2014.

[28] B. Kang, Z. Liu, X. Wang, F. Yu, J. Feng, and T. Darrell, "Few-shot object detection via feature reweighting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, p. 8420–8429.

[29] X. Yan, Z. Chen, A. Xu, X. Wang, X. Liang, and L. Lin, "Meta r-cnn: Towards general solver for instance-level low- shot learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, p. 9577–9586.

[30] Q. Fan, W. Zhuo, C.-K. Tang, and Y.-W. Tai, "Few-shot object detection with attention-rpn and multi-relation detector," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2020, pp. 4013–4022.

[31] G. Zhang, Z. Luo, K. Cui, and S. Lu, "Meta-detr: Few-shot object detection via unified image-level meta-learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2022.

[32] X. Du, Z. Wang, M. Cai, and Y. Li, "Vos: Learning what you don't know by virtual outlier synthesis," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2022.

[33] M. Cai and Y. Li, "Out-of-distribution detection via frequency-regularized generative models," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, 2023.

[34] A. Kumar, A. Raghunathan, R. Jones, T. Ma, and P. Liang, "Fine-tuning can distort pretrained features and underperform out-of-distribution," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2022.

[35] W. Liu, X. Wang, J. Owens, and Y. Li, "Energy-based out-of-distribution detection," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2020.

[36] E. Mark, V. Luc, and W. Christopher, "The pascal visual object classes (voc) challenge," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2010.

[37] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft coco: Common objects in context," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2015.

[38] A. Gupta, S. Narayan, K. J. Joseph, S. Khan, F. S. Khan, and M. Shah, "Ow-detr: Open-world detection transformer," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2022.

[39] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, pp. 936–944.

[40] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 770–778.

[41] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2021, pp. 9992–10 002.

[42] I. Pavel, P. Dmitrii, G. Timur, V. Dmitry, , and W. Andrew, "Averaging weights leads to wider optima and better generalization," in *Proc. Conf. Uncertainty Artif. Intell. (UAI)*, 2018.

[43] G. Ishaan and L.-P. David, "In search of lost domain generalization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2021.

[44] C. Junbum, C. Sanghyuk, L. Kyungjae, C. Han-Cheol, P. Seunghyun, L. Yunsung, and P. Sungrae, "Swad: Domain generalization by seeking flat minima," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2021.

[45] A. Rame, M. Kirchmeyer, T. Rahier, A. Rakotomamonjy, P. Gallinari, and M. Cord, "Diverse weight averaging for out-of-distribution generalization," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2022.

[46] Y. Gao, X. Shi, Y. Zhu, H. Wang, Z. Tang, X. Zhou, M. Li, and D. N. Metaxas, "Visual prompt tunting for test-time domain adaptation," in *arXiv:2210.04831*, 2022.

Binyi Su received the B.S. degree in intelligent science and technology from the Hebei University of Technology, Tianjin, China, in 2017, and the M.S degree in control engineering from the Hebei University of Technology, Tianjin, China, in 2020.

He is currently pursuing the Ph.D. degree in computer science and technology from Beihang University, Beijing, China. His current research interests include computer vision and pattern recognition, intelligent transportation system, and smart city.

Hua Zhang received the Ph.D. degrees in computer science from the School of Computer Science and Technology, Tianjin University, Tianjin, China in 2015.

He is currently an associate professor with the Institute of Information Engineering, Chinese Academy of Sciences. His research interests include computer vision, multimedia, and machine learning.

Jingzhi Li is currently an assistant professor with the Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China.

She received the Ph.D. degree in cyberspace security from the University of Chinese Academy of Sciences, Beijing, China. Her current research interests include face privacy, unbiased facial recognition, and machine learning.

Zhong Zhou received the B.S. degree in material physics from Nanjing University in 1999 and the Ph.D. degree in computer science and technology from Beihang University, Beijing, China, in 2005.

He is currently a Professor and Ph.D. Adviser at the State Key Laboratory of Virtual Reality Technology and Systems, Beihang University. His main research interests include virtual reality, augmented reality, computer vision, and artificial intelligence.