Few-Shot Object Detection With Self-Adaptive Global Similarity and Two-Way Foreground Stimulator in Remote Sensing Images

Yuchen Zhang, Bo Zhang, and Bin Wang, Senior Member, IEEE

Abstract—Few-shot object detection (FSOD) aims to localize and recognize potential objects of interest only by using a few annotated data, and it is beneficial for remote sensing images (RSIs) based applications, such as urban monitoring. Previous RSIs-based FSOD works often try to convert the support images from class-agnostic features to class-specific vectors, and then perform feature attention operations on query image features to be tested. However, such methods still face two critical challenges: 1) They ignore the spatial similarity of support-query features, which is indispensable for RSIs detection; 2) They perform the feature attention operation in a unidirectional manner, which means that the learned support-query relations are asymmetric. In this article, to address the challenges above, we design a few-shot object detector, which can quickly and accurately generalize to unseen categories with only a small amount of data. The proposed approach contains two components: 1) the self-adaptive global similarity module that preserves the internal context information to calculate the similarity map between the objects in support and query images, and 2) the two-way foreground stimulator module that can apply the similarity map to the detailed embeddings of support and query images at the same time to make full use of support information, further strengthening the foreground objects and weakening the unconcerned samples. Experiments are conducted on DIOR and NWPU VHR-10 datasets and their results demonstrate the superiority of the proposed method compared with several state-of-the-art methods.

Index Terms—Background suppression (BS), few-shot object detection (FSOD), remote sensing images (RSIs), self-adaptive global similarity (SAGS), two-way foreground stimulator (TFS).

I. INTRODUCTION

With the rapid development of computer vision, more and more methods dealing with object detection in remote sensing images (RSIs) have emerged [1], [2], [3], [4], [5], [6], [7], [8], [9]. The RSIs-based detection methods [1], [2], [3], [4], [5], [6], [7], [8], [9] are to correctly locate the position of interested objects and identify the categories to which the foreground objects belong in the given images captured by satellites or aircrafts. Therefore, it is a rather challenging task combining classification and regression. Thanks to many existing innovative technologies, the number of related applications has increased greatly, such as urban environmental monitoring, land use planning, forest fire alarms, traffic management, and other military uses [10], [11], [12], [13], [14]. Developing object detection in RSIs is bound to have promising prospects and bring great convenience for people managing and monitoring a certain area.

Initially, object detection in RSIs mainly works upon conventional theories [15], like template matching-based methods [16], [17], knowledge-based methods [18], [19], and OBIA-based methods [20], [21]. As deep neural networks (DNNs) [22], [23] advance, deep learning methods [1], [2], [3], [4], [5], [6], [7], [8], [9] has gradually outperformed traditional algorithms and become the mainstream solution for most object detection problems resorting to their strong feature extraction ability and plentiful parameters. However, the DNN-based models still strongly depend on a large amount of annotated training samples. Nevertheless, the acquisition or labeling of RSIs is not as easy as that of natural scene images. Capturing RSIs needs expensive aircrafts or even satellites, and the manual annotation process demands expensive professional knowledge. As a result, the scale of several widely used RSI datasets is far smaller than that of natural image datasets. We are often faced with the data-shortage dilemma when samples of some unseen novel classes suddenly appear. Training ordinary object detection models for RSIs like those mentioned before with insufficient data may cause serious performance degradation due to the well-known overfitting issue, and thus new approaches to ease the dilemma are desiderated.

Thanks to few-shot learning (FSL) inspired by the meta-learning technology, it becomes possible for a data-driven model to learn from a small number of training samples. Recently, FSL has participated in multiple computer vision tasks. In terms of image classification, Vinyals et al. propose a matching network [24], which converts the input images into embeddings and calculates the similarity between each pair of embeddings to match objects to the correct category. Snell et al. design a prototypical network [25] to obtain the prototype center of each class in the semantic space during training and compute the nearest center to the prototype of a test image to predict the category. Sun et al. build a relation network [26] that designs an object relation module to build the relationship between

Manuscript received 9 June 2022; revised 26 July 2022 and 21 August 2022; accepted 28 August 2022. Date of publication 31 August 2022; date of current version 8 September 2022. This work was supported by the National Natural Science Foundation of China under Grant 61731021 and Grant 61971141. (Corresponding author: Bin Wang.)

The authors are with the Key Laboratory for Information Science of Electromagnetic Waves (MoE), Fudan University, Shanghai 200433, China and also with the Image and Intelligence Laboratory, School of Information Science and Technology, Fudan University, Shanghai 200433, China (e-mail: 20210720277@fudan.edu.cn; bo.zhangzx@gmail.com; wangbin@fudan.edu.cn).

Digital Object Identifier 10.1109/JSTARS.2022.3203126

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/
Prototype-CNN (P-CNN) [37] modifies the original RPN of Faster R-CNN to a prototype-guided RPN (P-G RPN) to fuse class prototypes into the region proposal generation process by adding a complementary classifier. Path-aggregation multiscale few-shot detector (PAMS-Det) [38] reconstructs the backbone with an involution operator to enhance the classification precision and extracts multiscale features in a bottom-up flow with the help of semantic information. Oriented feature augmentation (OFA) [39] solves the arbitrary object direction problem with a novel data augmentation module based on dual pipelines. As the latest published work, context-aware aggregation network (CAAN) [40] proposes two components to help the model adapt to images of various scales and learn to strengthen context awareness to make better predictions. These works are indeed innovative and inspiring, but most of them still have the following two problems after an overall analysis.

First, nearly all the methods employ a global average pooling (GAP) to transform support images into vector form to adjust query features. Such operation only preserves numeric information but totally neglects the spatial information within support images, which is a key clue when performing a few-shot detection task. If targeting natural scene images, this might work well because objects from the same class have more negligible diversification in appearance, and the vector value will not change a lot even if some interferences occur. However, according to the patterns of RSIs discussed above, it is important to introduce spatial information for more accurate detection, especially for FSOD, which is vulnerable to bad quality of instances.

Second, the class-specific information extracted from support images is only transmitted forward to query features for channel-wise multiplication or other forms of refinement. Thus, most of the final extracted features come from query images, and the remaining small portion is from support images. However, according to the construction of input data, support images obviously contain more information about each category and should have played an essential role in the final feature. The previous ways discard all already encoded support features, which is a severe waste in a few-shot data situation. Therefore, better utilization of support features is indispensable for RS1-based FSOD. Besides, the support and query features are complementary and symmetrical. To ensure such a support-query feature relation, both support and query features should be learned simultaneously, which requires a two-way attention structure.
In order to solve the problems above, we propose a novel FSOD model in RSIs, which mainly consists of two parts: the self-adaptive global similarity (SAGS) module with background suppression (BS) and the two-way foreground stimulator (TFS) module. Same as many existing works [30], the input to the network is defined as the combination of a query set and a support set. A set of query and support images are fed into the feature extractor at the beginning of the network to generate a basic feature map for each image. Then the BS part subtracts the background noise vector from query features to strengthen the existence of possible objects. In order to keep the context information and augment the representative ability of features, each feature map is then encoded as a spatial relation feature and a detail embedding. Next, SAGS computes the similarity between support and query spatial relation features and outputs a similarity map showing how close the objects are in the support image and the query image globally. Due to the encoding operation, the similarity map learns to express a more precise global relation between query and support instances in a self-adaptive way. Then TFS employs a two-way attention operation by multiplying the similarity map with both support and query detail embeddings to fully use support features, highlight concerned instances, suppress irrelevant samples, preserve the consistency of support-query feature relations, and finally generate the final features for further prediction.

The main contributions of this article are concisely summarized as follows.

1) We propose a SAGS module to additionally introduce the spatial information and compute the global similarity between support and query instances, avoiding the accuracy degeneration brought by great appearance changes of objects from the same class in RSIs. Meanwhile, a simple but effective BS module is implemented to reduce the effect of background noises. This should be a pioneering work of using only a few training samples to maintain the contextual information and local relationships of the remote-sensing support images.

2) We design a TFS module to adopt a two-way attention mechanism and fully exploit the knowledge hidden in support images by feeding the similarity map to the detail embeddings of both query and support features. TFS highlights the interested objects and suppresses the irrelevant samples. With TFS, the helpful information can be preserved for model optimization.

II. RELATED WORKS

A. Object Detection

CNN-based object detection models have advanced a lot in recent years. Generally, these detectors can be divided into two main categories: anchor-based ones and anchor-free ones. Anchor-based detectors generate a group of anchors called region proposals and learn to predict the class and bounding box position of the anchor. According to the way to deal with region proposals, anchor-based models can be further defined as two-stage detectors and one-stage detectors. Typical two-stage detectors first generate proposals for potential objects, and then use a refinement operation to keep the proposals that they actually have objects for learning or inference. RCNN [42] uses a selective search strategy to generate region proposals. Fast R-CNN [43] introduces a region-of-interest (RoI) pooling layer to resize all the RoI features to the same size to benefit the following operations. Faster R-CNN [41] proposes the famous region proposal network (RPN) to learn how to provide better region proposals, making the anchor generation more accurate and efficient. Mask R-CNN [44] additionally appends a mask prediction branch to the original faster R-CNN. On the other hand, one-stage detectors ignore the region proposal refinement process and directly detect objects from the image with a single convolutional network, such as SSD [45] and Yolo v2-v4 [46], [47], [48]. Two-stage detectors usually have better precision than one-stage detectors with the help of their processed region proposals, but this will cost more time and memory to do the calculation. Hence, the usage scenarios should be taken into consideration when choosing detectors.

On the contrary, anchor-free models do not rely on any region proposals, which significantly reduces computational costs and improves speed. CornerNet [49] transform the object position from a bounding box to a pair of keypoints, including the top-left and bottom-right corner. By directly estimating such keypoints, CornerNet achieves instant detection. CenterNet [50] finds another unique way to predict the center of an object according to a generated heatmap, and then regress the bounding box. In general, anchor-free models can efficiently complete the task, but they are prone to facing data imbalance problems due to their way of defining positive and negative instances.

B. Few-Shot Object Detection

FSOD demands the model to detect objects with only a few annotated samples. Meta-learning-based approaches introduce a meta-learner to distill detection knowledge from base class objects and learn to generalize to unseen novel classes. For instance, Meta R-CNN [29] adds a predictor-head remodeling network to acquire class-attentive vectors and remodels the detection head. Feature reweighting [30] compresses each support image to a vector to modulate query features for novel class detection. Fine-tuning-based methods originate from two-stage fine-tuning approach (TFA) [27], which reaches a high precision only by freezing a part of the parameters without devising any new modules. Moreover, FSOD via contrastive proposal encoding (FSCE) [28] is the first to adopt contrastive learning in FSOD by bringing a contrastive head to the vanilla detection head.

C. FSOD in RSIs

Different from the prosperity of FSOD work in natural images, there are very few existing FSOD methods designed for RSIs. The first attempt is FSODM [35] based on Feature reweighting [30], which adds a multiscale mechanism to deal with the scale change of remote sensing objects. SAAN [36] adopts transfer learning instead of meta-learning and considers object-level relations with a relation GRU to detect unseen objects. PAMS-Det [38] replaces the convolutional backbone with
Fig. 2. Overview of the proposed method, including the SAGS module with BS and the TFS module.

an involution-based backbone, and builds a path-aggregation module to establish the feature pyramid to tackle the constant scale change. P-CNN [37] extends Meta R-CNN [29] and uses a P-G RPN instead of vanilla RPN. The P-G RPN takes class-wise prototypes, passes them through several fully connected layers, and uses the output as the weight of a convolution to attach a complementary classifier to the original RPN classification branch, which reaches a good precision. OFA [39] assembles horizontal and vertical flip to increase the amount of training data to ease the arbitrary direction problem of objects in a dual-pipelines structure. CAAN [40] proposes a context-aware pixel aggregation that uses convolution of different sizes to adapt to multiscale objects, and a context-aware feature aggregation to utilize more semantic information with a graph convolution network. However, all these attempts treat the information buried in support images as a vector form and totally give up the context relation, making them unable to adapt to the various appearance changes of objects in RSIs.

III. PROPOSED METHOD

The purpose of our work is to exploit the spatial information hidden in the scarce data and to make better use of the support features. Fig. 2 shows the novel framework with two main innovations: the SAGS module with BS and the TFS module. In this section, we will first state the problem definition of FSOD to clarify our goals. Then the designed components will be explained in detail.

A. Problem Definition

The whole dataset we possess is defined as \( D = D_{\text{base}} \cup D_{\text{novel}} \), where \( D_{\text{base}} = \{(x_i^{\text{base}}, y_i^{\text{base}})\}_{i=1}^{n_1} \) and \( D_{\text{novel}} = \{(x_i^{\text{novel}}, y_i^{\text{novel}})\}_{i=1}^{n_2} \) includes \( n_1 \) pairs of images \( x_i^{\text{base}} \) and annotations \( y_i^{\text{base}} \) from base classes \( C_{\text{base}} \) and \( n_2 \) pairs of images \( x_i^{\text{novel}} \) and annotations \( y_i^{\text{novel}} \) from novel classes \( C_{\text{novel}} \), respectively. Here \( n_1 \gg n_2 \), which means that \( D_{\text{base}} \) have abundant data whereas \( D_{\text{novel}} \) satisfies the few-shot settings with only a few labeled samples, whose purpose is to train a basic detection model out of \( D_{\text{base}} \) so that the model can generalize to recognize instances from novel classes. \( C_{\text{base}} \cap C_{\text{novel}} = \emptyset \) means that novel class examples will remain unseen until the transferring phase.

The construction of the input data is the same as that of [33], which consists of a query set and a support set. The query set only contains a query image that may have more than one object from various classes. The support set has one image per class containing an annotated sample of this category, thus the scale is the number of all classes \( C_{\text{all}} = C_{\text{base}} \cup C_{\text{novel}} \). Usually, the support images provide class-specific information to the
network, and the model learns to exploit such clues and detect the objects in query images.

B. SAGS Module

To tackle the scale diversification of RSI objects, a multiscale detection with features of five different sizes is achieved by taking the output of the last four backbone layers and down-sampling the smallest feature map by two times. The following modules will be explained based on one of the feature maps. After receiving the input query set and support set, the feature extractor outputs \( F = \{ F_{\text{support}} \in \mathbb{R}^{N \times C \times H_s \times W_s}, F_{\text{query}} \in \mathbb{R}^{C \times H_q \times W_q} \} \) as the original feature, where \( N \) denotes the number of categories involved, \( C \) represents the number of channels, \( H_s \) and \( W_s \) are the height and width of support feature maps, respectively, and \( H_q \) and \( W_q \) are the height and width of query feature maps, respectively. \( F_{\text{query}} \) is modified to the same size as \( F_{\text{support}} \) to simplify the following calculation. In order to capture the spatial information in support images, the feature should remain as a map form instead of a vector form, and the designed SAGS is utilized to compare spatial similarity between the query and support features. But this will allow the background noise to participate in the calculation below, thus causing unnecessary interference. To lighten the disturbance of complex backgrounds of RSIs, a simple but telling BS trick is designed before SAGS comes into play. The proportion of object bounding box area to the full picture size is smaller in RSIs than in natural scene images. Thus, the original feature of the query image \( F_{\text{query}} \) goes through a GAP to get a vector \( V_{bg} \) that is more likely to express the background patterns

\[
V_{bg} = \text{GAP}(F_{\text{query}}) = \{ \max(F_{\text{query}}^i) \}_{i=1}^C.
\]

\( F_{\text{query}} \) then subtracts this vector at every pixel and becomes \( F_{\text{query}}^* \) that has slighter background features

\[
F_{\text{query}}^* = F_{\text{query}}(x, y) - V_{bg},
\]

\( x = 1, 2, \ldots, H_q; y = 1, 2, \ldots, W_q. \) (2)

After the BS, the original features \( F = \{ F_{\text{support}}, F_{\text{query}}^* \} \) are encoded as a pair of new features: the spatial relation feature \( F_d \in \mathbb{R}^{N \times C_d \times H_s \times W_s} \) and the detail embedding \( F_d \in \mathbb{R}^{N \times C_d \times H \times W} \), where \( H, W, C_s, \) and \( C_d \) denote the height, width, and channel number of \( F_{\text{support}} \) and \( F_{\text{query}}^* \) respectively. \( C_s \) is one quarter of \( C_d \) to reduce parameters and distill abstract local information. When referring to the query spatial relation feature \( F_{\text{support}} \) and query detail embedding \( F_{d, \text{query}} \), \( N \) is set to be 1. When it comes to the support spatial relation feature \( F_{s, \text{support}} \) and support detail embedding \( F_{d, \text{support}} \), \( N \) is the number of classes participating in the training process. Through learning, \( F_s \) represents the local relation included in the image, and \( F_d \) stands for the refined specific information. The encoder is built with two \( 3 \times 3 \) convolution layers with independent parameters. Such encoding helps by reducing the dimension as well as concentrating related information and could achieve better performance than directly using the original feature as detail embedding, which will be verified by concrete experimental results given in Section IV. After encoding, \( F_s \in \mathbb{R}^{N \times C_s \times H \times W} \) and \( F_d \in \mathbb{R}^{N \times C_d \times (H \times W)} \) are flattened to be \( F_s \in \mathbb{R}^{N \times C_s \times (H \times W)} \) and \( F_d \in \mathbb{R}^{N \times C_d \times (H \times W)} \) for simplicity. Moreover, under few-shot data settings, the images available are so scarce that if they vary in brightness or tone, the network is confused because of their different data distributions, which impacts the stability of training and generalization. Because each row of the flattened \( F_s \) represents one channel of its feature map before flattening, the \( F_{s, \text{query}} \) and \( F_{s, \text{support}} \) are L2-normalized in rows to equalize the data distributions

\[
F_{s, \text{query}}(x, y) = \frac{F_{s, \text{query}}(x, y)}{\sqrt{\sum_{j=1}^{H \times W} (F_{s, \text{query}}(x, j))^2}}, x = 1, 2, \ldots, C_s;
\]

\[
y = 1, 2, \ldots, H \times W;
\]

\[
(3)
\]

\[
F_{s, \text{support}}(x, y) = \frac{F_{s, \text{support}}(x, y)}{\sqrt{\sum_{j=1}^{W_H \times W_c} (F_{s, \text{support}}(x, j))^2}}, x = 1, 2, \ldots, C_s;
\]

\[
y = 1, 2, \ldots, H \times W; i = 1, 2, \ldots, N.
\]

\[
(4)
\]

Then the transpose of \( F_{s, \text{support}} \) multiplies \( F_{s, \text{query}} \) and goes through a softmax normalization to calculate the global similarity map \( M \in \mathbb{R}^{N \times (H \times W) \times (H \times W)} \) for the query image and each support image

\[
M^i(x, y) = \frac{\exp((F_{s, \text{support}}^i)^T \otimes F_{s, \text{query}}(x, y))}{\sum_{j=1}^{H \times W} \exp((F_{s, \text{support}}^i)^T \otimes F_{s, \text{query}}(j, y))},
\]

\( x = 1, 2, \ldots, H \times W; y = 1, 2, \ldots, H \times W; i = 1, 2, \ldots, N, \)

\[
(5)
\]

where \( \otimes \) means matrix multiplication. The softmax normalization also aims to maintain the uniform distributions. No matter whether the appearance changes occur, objects from the same class always have at least a local area alike, which generates similar features through the backbone. When computing \( M \), similar features tend to output larger values, and the values are more likely to stay continuous, but distinct parts will get a sparse and weak similarity score. In this way, \( M \) is able to magnify instances with a local correspondence and suppress objects with different patterns.

Note that the “self-adaptive” in SAGS means that the similarity map can be trained to be more representative because we have one specialized part to hold the context relation and another one to collect details from the original feature. Such separation has a tendency to assign correlative information to the corresponding part, which guarantees the output of SAGS does contain the degree of similarity between different spatial locations.

C. TFS Module

By means of the SAGS, the similarity map is introduced to solve the spatial information neglect problem caused by converting support features into attention-like vectors. However, the attention weights are only applied to query features in a soft-attention or channel-wise modulating manner in previous works. Such processing methods entirely discard existing support features and make the support and query features asymmetrical and inconsistent. Since support images consist of one instance for each class, they naturally carry more specific and abundant information and should be exploited fully before entering the
detection head. Furthermore, the similarity map calculated by SAGS is capable of indicating similar local areas in the detail embeddings because the \(i\)th channel of the similarity map \(M\) represents the local similarity between the query spatial relation feature \(F_{s, \text{query}}\) and the object from the \(i\)th channel. Therefore, the TFS module is created for the challenges above. With the guidance of \(M\), TFS adds two-way attention to the detail embeddings with the following method.

\(M\) is multiplied with query detail embedding \(F_{d, \text{query}}\) and support detail embedding \(F_{d, \text{support}}\) simultaneously, which stimulates the interested objects and suppresses those unconcerned samples or backgrounds. This mechanism outputs two optimized detail embeddings:

\[
F_{d, \text{query}}^* = \{F_{d, \text{query}} \odot M^i\}_{i=1}^N \tag{6}
\]
\[
F_{d, \text{support}}^* = \{F_{d, \text{support}} \odot M^i\}_{i=1}^N \tag{7}
\]

where \(N\) is the number of classes and \(M^i\) is the \(i\)th channel of similarity map \(M\). After the computation, both embeddings are folded to be \(F_{d, \text{query}}^* \in \mathbb{R}^{N \times C_d \times H \times W}\) and \(F_{d, \text{support}}^* \in \mathbb{R}^{N \times C_d \times H \times W}\). Since it is more efficient to have neural networks learn the matching degree of \(F_{d, \text{query}}^*\) and \(F_{d, \text{support}}^*\), the results of the two branches are then concatenated and summed together to produce the final feature \(F_{\text{final}}\):

\[
F_{\text{final}} = \sum_{i=1}^N \text{concat}(F_{d, \text{query}}^i, F_{d, \text{support}}^i). \tag{8}
\]

Fig. 3 demonstrates an example of the calculation process. After computing the optimized detail embeddings of each channel according to (6) and (7), the corresponding results of the channel \(i\), \(F_{d, \text{query}}^i \in \mathbb{R}^{1 \times C_d \times H \times W}\) and \(F_{d, \text{support}}^i \in \mathbb{R}^{1 \times C_d \times H \times W}\) are concatenated at the second dimension to form \(F_{\text{concat}} \in \mathbb{R}^{1 \times (2 \times C_d) \times H \times W}\). In this manner, we get \(N\) concatenated results \(F_{\text{concat}}\) since \(i = 1, 2, \ldots, N\), where \(N\) denotes the number of categories involved. Finally, the values at the same position of all the \(F_{\text{concat}}\) are summed while maintaining the shape of \(F_{\text{final}} \in \mathbb{R}^{1 \times (2 \times C_d) \times H \times W}\) fixed, which is a channel-wise summation. The query detail embedding is reused in the calculation process of every channel. Such two-way strengthening makes foreground objects outstanding and weakens the existence of irrelevant objects in both query and support images, which is substantially a graph-based form of attention implemented with matrix multiplication, while previous works resort to pixel-wise adjustment. Moreover, TFS preserves the coherence between support and query features by treating them equally.

\(F_{\text{final}}\) is then resized to restore its original size for subsequent multiscale detection. The rest parts of the model function as a relation module that decides the matching degree between support and query detail embeddings. RPN takes \(F_{\text{final}}\) as input and preprocesses the anchors to generate region proposals that are more likely to contain foreground objects. Finally, the detection head predicts the class and bounding box for each region proposal and outputs the results.

D. Overall Loss Function and Adaptation Strategy

1) Loss Function: Our loss function consists of a classification loss and a bounding box regression loss for both RPN and the R-CNN detection head. The RPN mainly determines whether a regional proposal is foreground or not, but it does not care about the category. Hence, the RPN classification loss \(L_{\text{rpn}}^{\text{cls}}\) is a binary cross-entropy loss function, and the RPN regression loss \(L_{\text{rpn}}^{\text{reg}}\) adopts smooth-L1 loss to keep the gradients stable during backpropagation. The detection head has to predict the classes of RoIs, therefore the classification loss \(L_{\text{cls}}^{\text{roi}}\) is a multitask cross-entropy loss function, while the regression loss \(L_{\text{reg}}^{\text{roi}}\) is the same as \(L_{\text{rpn}}^{\text{reg}}\):

\[
L_{\text{rpn}}^{\text{cls}}(p_i, y_i) = \frac{1}{N} \sum_i -y_i \log(p_i) - (1 - y_i) \log(1 - p_i) \tag{9}
\]
\[
L_{\text{rpn}}^{\text{reg}}(p_i, y_i) = -\frac{1}{N} \sum_i \sum_{c=1}^C y_{ic} \log(p_{ic}) \tag{10}
\]
\[
L_{\text{reg}}^{\text{roi}}(t_i, t_i^*) = \sum_{i \in \{x,y,w,h\}} \text{smoother}_{\text{L1}}(t_i - t_i^*) \tag{11}
\]
\[
\text{smoother}_{\text{L1}}(x) = \begin{cases} 0.5x^2, & |x| < 1 \\ |x| - 0.5, & \text{otherwise} \end{cases} \tag{12}
\]

The overall optimization objective can be formulated as

\[
\min L = L_{\text{cls}}^{\text{roi}}(p_i, y_i) + L_{\text{cls}}^{\text{roi}}(p_i, y_i) + L_{\text{reg}}^{\text{roi}}(t_i, t_i^*) + L_{\text{reg}}^{\text{roi}}(t_i, t_i^*) \tag{13}
\]
is trained only with sufficient data $D_{base}$, merely from base classes for more iterations to get a basic detection model. After base training, only the last fully connected layer should be replaced by another randomly initialized one whose output matches the number of $C_{all} = C_{base} \cup C_{novel}$. In fine-tuning phase, $K$ instances of each class from $C_{all} = C_{base} \cup C_{novel}$ are randomly extracted to construct a balanced subdataset to generalize the learned knowledge to novel classes swiftly.

### IV. Experimental Results and Analysis

**A. Dataset and Experimental Settings**

1) **Dataset:** In this article, the experiments are conducted on two widely used remote sensing datasets: DIOR [51] and NWPU VHR-10 [15].

DIOR [51] is a large-scale open dataset for RSIs-based object detection. It consists of 23 463 RSIs with annotations and 192 472 samples of 20 categories. The classes are airplane, airport, baseball field, bridge, dam, expressway service area, expressway toll station, harbor, golf course, ground track field, overpass, ship, stadium, storage tank, tennis court, train station, vehicle, and windmill. The whole dataset is divided into a training set, a validation set, and a testing set, containing 5862, 5863, and 11738 images, respectively. Our two-stage training utilizes the combination of the training set and the validation set, and the performance of our method is evaluated on the testing set. All the images are 800 × 800 3-channel RGB pictures.

Split 1 to Split 4 are the same as those of P-CNN [37], and Split 5 is the same as that of FSODM [35], as shown in Table I. Thus, the experiments on the DIOR dataset are separated into two parts. The division of the first four splits is a more reasonable way to construct the data splits since each individual category has one chance to be novel class. The last split is adopted for comparing with more methods. Five classes are assigned as novel classes in every split, and the remaining 15 classes form the adequate base dataset. In the fine-tuning stage, $K = 3, 5, 10, 20, 30$ objects per class are selected to build the balanced subdataset. Since the size of images in NWPU VHR-10 is not the same, the data are resized to 1200 × 1800 uniformly.
2) Experimental Settings: To focus on the foreground objects and facilitate the training of SAGS, the support images only have the object in the center while other positions remain at zero values, as shown in the support set of Fig. 2. Other than resizing the object patches to the same size that changes the original ratio of instances as [37], the objects are cropped according to their bounding box labels from the whole image and the longer edge is resized to 256 while maintaining the ratio of objects. The enlarged patches are placed at the center of a $256 \times 256$ all-zero 3-channel matrix to generate a support image. In order to inform the network where to locate the samples, a mask channel is attached to the image. In summary, the support images are the combination of a cropped and resized object and its corresponding mask channel.

The method is implemented based on Pytorch 1.4.0. ResNet-101 [52] pretrained on ImageNet with an FPN [53] is used as our backbone for feature extraction. For base training, the initial learning rate is 0.0005, decreasing by 0.1 at 6000 and 10 000 iterations. The model is trained for 12 000 iterations. For fine-tuning, the initial learning rate is 0.00025, decreasing by 0.1 at 1150 and 1350 iterations. The model is fine-tuned for 1500 iterations in all. At the end of training, the support original features of each class will be saved and averaged to get the mean class features. In the testing process, the model uses the saved mean class features to generate the support space relation features and the support detail embedding. Our experiments are loaded on 4 NVIDIA 2080ti GPU with 12G memory with a batch size of 4. The optimizer used is stochastic gradient descent with a momentum coefficient of 0.9 and a weight decay of 0.0001. The commonly used criterion mean average precision (mAP) is adopted as our evaluation metric.

B. Comparing Methods

1) DIOR: In Split 1 to Split 4, the proposed method is compared with one object detection approach (Faster R-CNN [41]), four natural-images-based classical FSOD algorithms (RepMet [31],FeatReweighting [30], TFA [27], and Meta R-CNN [29]), and two novel RSIs-based FSOD methods (P-CNN [37] and CAAN [40]). CAAN achieves the best performance in Split 1 and Split 3 at three different shots, while P-CNN shows the state-of-the-art results under other situations. Note that the results of all the comparing methods are cited from P-CNN [37] and CAAN [40]. In Split 5, the proposed method is compared with the two classical FSOD algorithms for natural images, TFA [27] andFeatReweighting [30], and two FSOD approaches in RSIs, FSODM [35] and PAMS-Det [38].

2) NWPU VHR-10: The proposed method is compared with one classical FSOD algorithm for natural images, TFA [27]; and three RSIs-based FSOD methods, FSODM [35], PAMS-Det [38], and OFA [39]. Note that the results of all the methods are quoted from PAMS-Det [38] and OFA [39], which are the state-of-the-art works in NWPU VHR-10 under this data split.

C. Experimental Results

1) DIOR: The FSOD performance on the DIOR dataset is shown in Tables II and III. The experiment runs on five different sets of few-shot data for fine-tuning and the average mAP is taken as the final result. Our method outperforms all the approaches in most situations. For example, at 30-shot in Split 1 and Split 3, the performance is increased by 12.8% and 8.2%, respectively. The different difficulty of data splits is an important reason for the uneven improvement. On the one hand, the instances of novel classes in Split 1 and Split 3 are not as complicated as those in Split 2 and Split 4. For example, the sports ground-related categories that have relatively fewer details and higher detection
TABLE III
DETECTION RESULTS ON THE DIOR DATASET IN THE FIFTH SPLIT

| Method     | Shot | Airplane | Baseball field | Tennis court | Train station | Wind mill | Mean |
|------------|------|----------|----------------|--------------|---------------|-----------|------|
| TFA [27]   | 13   | 51       | 24             | 13           | 25            | 25        |      |
| FeatReweighting [30] | 9    | 33       | 47             | 9            | 13            | 22        |      |
| FSODM [35] | 5    | 9        | 27             | 57           | 11            | 19        | 25   |
| PAMS-Det [38] | 14   | 54       | 24             | 17           | 31            | 28        |      |
| Ours       | 17   | 50       | 62             | 17           | 23            | 34        |      |

TABLE IV
DETECTION RESULTS ON THE NWPU VHR-10 DATASET

| Method     | Shot | Airplane | Baseball diamond | Tennis court | Mean |
|------------|------|----------|------------------|--------------|------|
| TFA [27]   | 12   | 61       | 13               | 13           | 29   |
| FSODM [35] | 15   | 57       | 25               | 25           | 32   |
| OFA [39]   | 3    | 87       | 25               | 25           | 43   |
| PAMS-Det [38] | 21   | 76       | 16               | 16           | 37   |
| Ours       | 35   | 76       | 43               | 43           | 51   |

In addition, the computational cost can be evaluated by the floating point operations (FLOPs) and the amount of parameters. The detailed FLOPs of each module in the training on the two datasets are shown in Table V. For DIOR, the FLOPs in base training, fine-tuning phase and testing are 322.4873, 394.8738, and 173.5893 GFLOPs, respectively, where 1 GFLOPs = 10^9 FLOPs. For NWPU VHR-10, the FLOPs in base training, fine-tuning phase, and testing are 435.7448, 479.4018, and 327.1684 GFLOPs, respectively. The FLOPs reduction in testing is because the feature extractor only extracts features for query images. The number of parameters is at most 49.58 million. The
slight difference is caused by the different number of classes in the base training and fine-tuning phase, just because more classes participating the training process leads to more convolution kernels in the encoder, and thus increases the number of parameters. Moreover, the training time per iteration and the testing time per image are also presented in Table V, where the batch size is 4 during training and the inference time is computed on one GPU. The detection speed is relatively slower than that of FSODM [35] whose speed is 52.91 FPS because our method is based on the two-stage object detection model (Faster R-CNN) while FSODM [35] is based on the one-stage model (Yolov3). The adopted multiscale detection is also a time-consuming strategy. However, although our method has a relatively long detection time as compared to the FSODM, it achieves a significant improvement in the detection accuracy.

### D. Ablation Study

We conduct ablation studies on the DIOR dataset at 30 shots in Split 1 to Split 4 to show the effectiveness of the proposed modules and the encoding strategy. The experiments include two aspects. One is to exclude only a single component at a time, which could verify each part of our network. The other is to delete all the modules one by one, which could express how those modules affect and work with each other. We run the experiments in all data splits under the few-shot settings three times and take the average.

#### 1) Single Module Ablation: 

The ablation results on the DIOR dataset can be observed in Table VI, where “w/ All” means running the intact designed network, and “w/o” means running the model without the following module. Among all the attempts, encoding the original features as two separated ones

| Modules          | DIOR     | NWPU VHR-10 |
|------------------|----------|-------------|
|                  | Base     | Fine-Tuning | Base     | Fine-Tuning |
| Train Time (Per iteration) | 0.661    | 0.693       | 1.090    | 1.206       |
| Test Time (Per image)    | 0.083    | 0.089       | 0.119    | 0.129       |
| Feature Extractor        | 245.8367 | 295.4694    | 396.6536 | 426.4337    |
| Encoder                 | 3.774    | 4.7175      | 1.3209   | 1.887       |
| SAGS                    | 1.0066   | 1.3421      | 0.4698   | 0.6711      |
| TFS                     | 64.4245  | 85.8993     | 29.855   | 42.955      |
| RPN + detection heads   | 7.4455   | 7.4455      | 7.4455   | 7.4455      |

The unit of time measurement is seconds. The unit of FLOPs measurement is GFLOPs, where 1 GFLOPs = 10^9 FLOPs.

### TABLE VI

| Shot | Method | Split1 | Split2 | Split3 | Split4 |
|------|--------|--------|--------|--------|--------|
| 3    | w/ All | 29.30  | 12.61  | 20.89  | 17.46  |
|      | w/o SAGS | 27.88  | 11.40  | 20.30  | 15.56  |
|      | w/o TFS | 28.27  | 10.47  | 19.76  | 12.71  |
|      | w/o BS  | 27.81  | 10.24  | 20.66  | 16.45  |
|      | w/o Enc. | 17.39  | 9.15   | 18.34  | 12.89  |
| 5    | w/ All | 31.63  | 15.53  | 24.82  | 19.70  |
|      | w/o SAGS | 31.10  | 14.32  | 22.93  | 17.44  |
|      | w/o TFS | 30.62  | 13.14  | 23.72  | 15.31  |
|      | w/o BS  | 29.99  | 14.15  | 23.77  | 18.28  |
|      | w/o Enc. | 19.64  | 9.16   | 16.58  | 12.39  |
| 10   | w/ All | 36.39  | 19.70  | 28.62  | 23.68  |
|      | w/o SAGS | 35.49  | 17.97  | 25.68  | 21.66  |
|      | w/o TFS | 35.79  | 16.24  | 26.78  | 18.09  |
|      | w/o BS  | 34.89  | 16.39  | 27.46  | 22.42  |
|      | w/o Enc. | 15.90  | 13.05  | 19.28  | 13.09  |
| 20   | w/ All | 40.15  | 23.78  | 36.06  | 27.65  |
|      | w/o SAGS | 38.95  | 21.29  | 33.15  | 25.90  |
|      | w/o TFS | 38.87  | 20.72  | 33.10  | 19.91  |
|      | w/o BS  | 38.36  | 20.18  | 33.94  | 26.00  |
|      | w/o Enc. | 24.14  | 13.02  | 22.91  | 16.16  |
| 30   | w/ All | 45.37  | 29.59  | 40.30  | 30.23  |
|      | w/o SAGS | 43.87  | 25.49  | 38.54  | 27.16  |
|      | w/o TFS | 43.94  | 25.77  | 36.54  | 25.98  |
|      | w/o BS  | 43.63  | 25.92  | 37.27  | 26.98  |
|      | w/o Enc. | 33.69  | 13.89  | 24.74  | 18.41  |

The evaluation metric is mAP.

### TABLE VII

| Shot | Method | Split1 | Split2 | Split3 | Split4 |
|------|--------|--------|--------|--------|--------|
| 3    | w/ All | 29.30  | 12.61  | 20.89  | 17.46  |
|      | w/o SAGS | 27.88  | 11.40  | 20.30  | 15.56  |
|      | w/o TFS | 28.27  | 10.47  | 19.76  | 12.71  |
|      | w/o BS  | 27.81  | 10.24  | 20.66  | 16.45  |
|      | w/o Enc. | 17.39  | 9.15   | 18.34  | 12.89  |
| 5    | w/ All | 31.63  | 15.53  | 24.82  | 19.70  |
|      | w/o SAGS | 31.10  | 14.32  | 22.93  | 17.44  |
|      | w/o TFS | 30.62  | 13.14  | 23.72  | 15.31  |
|      | w/o BS  | 29.99  | 14.15  | 23.77  | 18.28  |
|      | w/o Enc. | 19.64  | 9.16   | 16.58  | 12.39  |
| 10   | w/ All | 36.39  | 19.70  | 28.62  | 23.68  |
|      | w/o SAGS | 35.49  | 17.97  | 25.68  | 21.66  |
|      | w/o TFS | 35.79  | 16.24  | 26.78  | 18.09  |
|      | w/o BS  | 34.89  | 16.39  | 27.46  | 22.42  |
|      | w/o Enc. | 15.90  | 13.05  | 19.28  | 13.09  |
| 20   | w/ All | 40.15  | 23.78  | 36.06  | 27.65  |
|      | w/o SAGS | 38.95  | 21.29  | 33.15  | 25.90  |
|      | w/o TFS | 38.87  | 20.72  | 33.10  | 19.91  |
|      | w/o BS  | 38.36  | 20.18  | 33.94  | 26.00  |
|      | w/o Enc. | 24.14  | 13.02  | 22.91  | 16.16  |
| 30   | w/ All | 45.37  | 29.59  | 40.30  | 30.23  |
|      | w/o SAGS | 43.87  | 25.49  | 38.54  | 27.16  |
|      | w/o TFS | 43.94  | 25.77  | 36.54  | 25.98  |
|      | w/o BS  | 43.63  | 25.92  | 37.27  | 26.98  |
|      | w/o Enc. | 33.69  | 13.89  | 24.74  | 18.41  |

The evaluation metric is mAP.
Fig. 5. Visualization of detection in Split 1 at 30 shots on the DIOR dataset. (a) Base classes only. (b) Novel classes only. (c) Both base classes and novel classes. (d) Hard cases. (e) Failure cases. Yellow boxes mark the predictions given by the model and red boxes indicate the ground truth of the objects inferred incorrectly.

leads to the most remarkable boost in performance. In our vision, the spatial relation feature $F_s$ and the detail embedding $F_d$ are learned to hold context information and detail information respectively from $F$ at the same time, which demands $F$ to be a more comprehensive representation. But the absence of $F_d$ forces $F$ to keep more specific information, which damages the quality of $F_s$. Conversely, $F_s$ wants more spatial information from $F$, making $F$ lean to the opposite direction. Such an adversarial effect makes $F$ oscillate and impacts the stability of the network. According to our record, results without encoding could fall in an extensive range and fluctuate widely. Therefore, our self-adaptive encoding is more robust and stable.

The GAP is applied to the original feature to get a vector like many existing methods, and the spatial relation feature is replaced with this new vector to remove SAGS, which completely discards the spatial information. Since $M$ is then calculated by multiplying the query vector and spatial vector, the size of $M$ is maintained, controlling other variables unchanged to verify the validity of SAGS. Results show that SAGS is beneficial for all the splits, especially for Split 2. The harbors in Split 2 have considerable variability in appearance because there are always ships staying in them and the shape of the harbors is irregular. SAGS handles such situations well and improves detection precision.

TFS delivers $M$ to detail embedding of both query and support images. To ablate TFS, $M$ is only multiplied with query detail embedding while other operations remain the same. According to Table VI, TFS is effective in most cases. For instance, TFS
raises the precision by 7.74% at 20-shot in Split 4. The train stations and expressway service areas in Split 4 both have buildings and railways, which makes them vulnerable to misclassification, especially when the data is scarce. TFS stimulates the critical characteristics in the objects from the same category, suppresses unconsidered features of other objects or backgrounds, and successfully enhances classification accuracy.

BS is a simple operation to subtract background information from the original query features. The results show that BS is rather effective here since most RSIs have complicated background information that should be preprocessed to ensure that the foreground features stand out.

2) Inter-Module Ablation: Table VII reports the precision change tendency on the DIOR dataset when the parts are removed one by one. Obviously, each module or strategy contributes to the detection. Note that when SAGS and TFS are removed, without encoding will not cause as severe damage as they are present. This confirms our previous suppose that it is the SAGS that makes the original feature oscillate and degenerates the performance.

E. Visualization

The 30-shot results in Split 1 on the DIOR dataset are visualized in Fig. 5 and are divided into five rows. Row (a) shows some examples of base classes alone object detection. The classification and regression accuracy are high since base class data are sufficient. The model can handle some dense and multiscale situations like the storage tank and airplane pictures at the end of (a). Row (b) has one picture for each novel class. Besides, images in Row (c) contain novel and base class objects simultaneously, proving that our method has the ability to detect unseen instances while maintaining the knowledge of previously learned classes. Among all the results, several hard cases are shown in Row (d) where either the appearance changes a lot or misclassification is easy to occur. The overpass in the first image is very similar to a bridge, which may confuse the network. However, our model classifies it correctly. The rest four pictures are here to demonstrate the effect of SAGS and TFS. In the second image, the two basketball courts below are half-court which means that their features may have a significant distinction from the intact court. The trees and their shadow cover the tennis courts in the following picture. In the fourth result, three chimneys have clouds floating above them. In the last image, the harbors look much different from those in the third image of (a) because several ships are parking inside densely. Fig. 5 shows that our method has excellent robustness and can deal with a greater degree of appearance change. SAGS and TFS successfully preserve the spatial information and make objects with similar parts more possible to be matched as the same category.

F. Failure Cases

Some failure examples are presented in Fig. 5(e). The yellow boxes mark the predictions given by the model and the red boxes indicate the ground truth of the objects inferred incorrectly. The cases are mainly divided into two groups: misclassification and small/dense objects miss. Since our method compares the local similarity of query and support features, it is possible to assign a similar but incorrect category to an instance. In the first image, a soccer field at the bottom left corner is considered as a tennis court, and the overpasses in the second picture are predicted to be bridges. These classes indeed have quite similar details that even humans may recognize them wrongly. The detection ability has a lot to improve for extremely small or dense objects, like the vehicles in the third image and the ships in the fourth image. Such samples usually have a size smaller than 20 × 20 and occupy only a tiny portion of the feature map after several convolutional and pooling layers. Besides, objects that are fairly much smaller than their typical size will also suffer from such detection missing problems. To fix the weakness mentioned above, spatial information should be utilized more precisely, and the multiscale situation needs to be taken into account in a more specific way.

V. Conclusion

In this article, we have proposed a novel FSOD model for RSIs with two developed modules: the SAGS module with BS and the TFS module. After the BS preprocessing query features to weaken the background noise, SAGS calculates the global context similarity between the encoded spatial relation feature of query and support images in a self-adaptive way, aiming at preserving the spatial information within support images and resisting the disturbance of various changes in object appearance. To fully exploit support features and keep the feature consistency, TFS applies the similarity map to both support and query detail embeddings simultaneously to highlight the interested objects while restraining the irrelevant features. Overall experiments on DIOR and NWPU VHR-10 datasets show that our method outperforms all the comparing approaches and has good adaptability to the characteristics of RSIs. The ablation studies verify the effectiveness of each module.

However, from the failed cases, there is still much room for improvement in the way we exploit spatial information. The scale change of objects should also be treated from a better perspective to overcome the missing detection. Consequently, an FSOD model with a more sophisticated encoder and a multiscale feature fusion module can promote the transferability of knowledge from base classes to novel classes, which will be a possible topic for our future work.

REFERENCES

[1] G. Cheng, J. Han, P. Zhou, and L. Guo, “Multi-class geospatial object detection and geographic image classification based on collection of part detectors,” ISPRS J. Photogramm. Remote Sens., vol. 98, pp. 119–132, Dec. 2016.
[2] G. Cheng, P. Zhou, and J. Han, “Learning rotation-invariant convolutional neural networks for object detection in VHR optical remote sensing images,” IEEE Trans. Geosci. Remote Sens., vol. 54, no. 12, pp. 7405–7415, Dec. 2016.
[3] G. Zhang, S. Lu, and W. Zhang, “CAD-Net: A context-aware detection network for objects in remote sensing imagery,” IEEE Trans. Geosci. Remote Sens., vol. 57, no. 12, pp. 10015–10024, Dec. 2019.
[4] P. Wang, X. Sun, W. Diao, and K. Fu, “FMSSD: Feature-merged single-shot detection for multiscale objects in large-scale remote sensing imagery,” IEEE Trans. Geosci. Remote Sens., vol. 58, no. 5, pp. 3377–3390, May 2020.

[5] J. Zhang, C. Xie, X. Xu, and Z. Shi, “A contextual bidirectional enhancement method for remote sensing image object detection,” IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens., vol. 13, pp. 4518–4531, Aug. 2020.

[6] W. Cai, B. Zhang, and B. Wang, “Scale-aware anchor-free object detection via curriculum learning for remote sensing images,” IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens., vol. 14, pp. 9946–9958, 2021.

[7] J. Pang, C. Li, J. Shi, Z. Xu, and H. Feng, “R2-CNN: Fast tiny object detection in large-scale remote sensing images,” IEEE Trans. Geosci. Remote Sens., vol. 57, no. 8, pp. 5512–5524, Aug. 2019.

[8] P. Shamsholmoali, M. Zareapoor, J. Chanussot, H. Zhou, and J. Yang, “Rotation equivariant feature image pyramid network for object detection in optical remote sensing imagery,” IEEE Trans. Geosci. Remote Sens., vol. 60, 2021, Art. no. 5608614.

[9] J. Ding, N. Xue, Y. Long, G.-S. Xia, and Q. Lu, “Learning Rol transformer for oriented object detection in aerial images,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 2849–2858.

[10] J. Han et al., “Efficient, simultaneous detection of multi-class geospatial targets based on visual saliency modeling and discriminative learning of sparse coding,” ISPRS J. Photogram. Remote Sens., vol. 89, pp. 57–48, Mar. 2014.

[11] Y. Zheng, J. Kong, X. Jin, X. Wang, T. Su, and M. Zuo, “CropDeep: The crop vision dataset for deep-learning-based classification and detection in precision agriculture,” Sensors, vol. 19, no. 3, Mar. 2019, Art. no. 1058.

[12] F. Zhang, B. Du, L. Zhang, and M. Xu, “Weakly supervised learning based on coupled convolutional neural networks,” IEEE Trans. Geosci. Remote Sens., vol. 54, no. 9, pp. 5553–5563, Sep. 2016.

[13] X. Han, Y. Zhong, and L. Zhang, “An efficient and robust integrated geospatial object detection framework for high spatial resolution remote sensing imagery,” Remote Sens., vol. 9, no. 7, Jun. 2017, Art. no. 666.

[14] X. X. Zhu et al., “Deep learning in remote sensing: A comprehensive review and list of resources,” IEEE Geosci. Remote Sens. Mag., vol. 5, no. 4, pp. 8–36, Dec. 2017.

[15] G. Cheng and J. Han, “A survey on object detection in aerial images,” ISPRS J. Photogramm. Remote Sens., vol. 117, pp. 11–28, Jul. 2016.

[16] K. Stankov and D. He, “Detection of buildings in multispectral very high spatial resolution images using the percentage occupancy hit-or-miss transform,” IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens., vol. 7, no. 10, pp. 4069–4080, Oct. 2014.

[17] S. Leninishia and K. Vani, “Water flow based geometric active deformable model for road network,” ISPRS J. Photogram. Remote Sens., vol. 102, pp. 140–147, Apr. 2015.

[18] A. O. Ok, Automated detection of buildings from single VHR multispectral images using shadow information and graph cuts,” ISPRS J. Photogramm. Remote Sens., vol. 86, pp. 21–40, Dec. 2013.

[19] A. O. Ok, C. Senaras, and B. Yuksel, “Rotation equivariant feature image pyramid network for object detection in optical remote sensing imagery,” IEEE Trans. Geosci. Remote Sens., vol. 51, no. 3, pp. 1701–1717, Mar. 2012.

[20] T. Blaschke et al., “Geographic object-based image analysis—towards a new paradigm,” ISPRS J. Photogramm. Remote Sens., vol. 87, pp. 180–191, Jan. 2014.

[21] D. Contreras, T. Blaschke, D. Tiede, and M. Jilge, “Monitoring recovery after earthquakes through the integration of remote sensing, GIS, and ground observations: The case of L’Aquila (Italy),” Cartogr. Geographic Inf. Sci., vol. 43, no. 2, pp. 115–133, Apr. 2016.

[22] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in Proc. Int. Conf. Learn. Represent., 2015, pp. 1–10.

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in Proc. Int. Conf. Neural Inf. Process. Syst., 2012, pp. 1097–1105.

[24] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, “Matching networks for one shot learning,” in Proc. Int. Conf. Neural Inf. Process. Syst., vol. 29, 2016, pp. 1–9.

[25] J. Snell, K. Swersky, and R. Zemel, “Prototypical networks for few-shot learning,” in Proc. Int. Conf. Neural Inf. Process. System, vol. 30, 2017, pp. 4087–4097.

[26] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. Torr, and T. M. Hospedales, “Learning to compare: Relation network for few-shot learning,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2018, pp. 1199–1208.
Yuchen Zhang received the B.S. degree in electronic information science and technology, in 2020, from Fudan University, Shanghai, China, where he is currently working toward the M.S. degree in circuits and systems with the Department of Electronic Engineering. His main research interests include computer vision and few-shot object detection.

Bo Zhang received the M.S. degree in computer science from School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China, in 2018. He is currently working toward the Ph.D. degree in electronic science and technology with School of Information Science and Technology, Fudan University, Shanghai, China. His research interests include computer vision, domain adaptation, and few-shot learning.

Bin Wang (Senior Member, IEEE) received the B.S. degree in electronic engineering and the M.S. degree in communication and electronic systems from Xidian University, Xi’an, China, in 1985 and 1988, respectively, and the Ph.D. degree in system science from Kobe University, Kobe, Japan, in 1999. After his graduation in 1988, he was with Xidian University as a Teacher. From 1999 to 2000, he was with the Communications Research Laboratory, Ministry of Posts and Telecommunications, Kobe, Japan, as a Research Fellow, working on magnetoencephalography signal processing and its application for brain science. Then, as a Senior Supervisor, he was with the Department of Etching, Tokyo Electron AT Ltd., Tokyo, Japan, from 2000 to 2002, dealing with the development of advanced process control systems for etching semiconductor equipment. Since September 2002, he has been with the Department of Electronic Engineering, Fudan University, Shanghai, China, where he is currently a full Professor and Leader of the Image and Intelligence Laboratory. He has authored or coauthored more than 150 scientific papers in important domestic and international periodicals. He is the holder of several patents. His main research interests include multispectral/hyperspectral image analysis, automatic object/target detection and recognition, pattern recognition, signal detection and estimation, and machine learning.

Dr. Wang is an Associate Editor of the IEEE JOURNAL OF SELECTED TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING (IEEE JSTARS).