Incremental Detection of Remote Sensing Objects With Feature Pyramid and Knowledge Distillation

Jingzhou Chen, Shihao Wang, Ling Chen, Haibin Cai, and Yuntao Qian, *Member, IEEE*

***Abstract*— When a detection model that has been well-trained on a set of classes faces new classes, incremental learning is always necessary to adapt the model to detect the new classes. In most scenarios, it is required to preserve the learned knowledge of the old classes during incremental learning rather than reusing the training data from the old classes. Since the objects in remote sensing images often appear in various sizes, arbitrary directions, and dense distribution, it further makes incremental learning-based object detection more difficult. In this article, a new architecture for incremental object detection is proposed based on feature pyramid and knowledge distillation. Especially, by means of a feature pyramid network (FPN), the objects with various scales are detected in the different layers of the feature pyramid. Motivated by Learning without Forgetting (LwF), a new branch is expended in the last layer of FPN, and knowledge distillation is applied to the outputs of the old branch to maintain the old learning capability for the old classes. Multitask learning is adopted to jointly optimize the losses from two branches. Experiments on two widely used remote sensing data sets show our promising performance compared with state-of-the-art incremental object detection methods.**

***Index Terms*— Deep learning, incremental learning, object detection, remote sensing.**

I. Introduction

THE rapid development of remote sensing technology has significantly increased the quantity and quality of remote sensing images available to characterize various objects on the earth's surface. Object detection in remote sensing plays a crucial role in a wide scope of applications, such as intelligent monitoring, urban planning, and precision agriculture [1]. Benefiting from powerful feature representation capabilities and some publicly available natural image data sets, such as Microsoft common objects in context (MSCOCO) [2] and PASCAL visual object classes (VOC) [3], many deep learning-based object detection approaches have achieved great success in natural scene images, e.g., fast RCNN [4], faster RCNN [5], feature pyramid network (FPN) [6], YOLOv2 [7], SSD [8], and RetinaNet [9]. Somewhat different to natural images, objects in remote sensing images often appear in various scales, arbitrary orientations, and cluttered arrangements. Therefore, several data sets have been developed in recent years, such as NWPU VHR-10 [10], DOTA [11], and DIOR [12]. Deep learning-based object detection methods also achieve good performance on these remote sensing data sets [13], [14].

One of the challenging tasks encountered in object detection is how to adapt the detector trained on a set of old target classes to unseen new classes. Each time the model meets a new class, conventional approaches have to be completely retrained on all images of the new class along with the previous classes. This requires the storage of all the available training sets and high computational time. For these reasons, conventional methods have critical limitations for deploying the well-trained models in some platforms where new target classes are often encountered, but the training sample set of old classes is no longer available. Therefore, it is necessary that, without utilizing images from the old classes, the updated detector could provide satisfactory results on unseen new classes and preserve the knowledge learned from old classes at the same time. Such a problem belongs to incremental learning, which refers to the situation of continuous model adaptation based on a constantly arriving data stream [15]. Many incremental learning approaches have been proposed [16]–[24].

Fine-tuning [25] is a simple technique to adapt a network to new classes. This type of method adjusts the output layer of the original network in two ways: one replaces it with new classes to construct a new network just for new classes, and the other adds new nodes corresponding to new classes to the existing nodes. The whole parameters of the network are tuned with the training data from new classes. However, it suffers from catastrophic forgetting—an abrupt degradation of performance on the old classes when the training objective function is adapted to new classes [26], [27]. This phenomenon of forgetting is caused by two main factors [28], [29]. First, the internal nodes are largely interconnected with each other, and a small change in one single neuron could affect many neurons [28]. Second, all parameters in the feedforward network participate in computations for every data sample, and the backpropagation updates every parameter in each training step [29].

Manuscript received June 7, 2020; revised August 20, 2020, November 3, 2020, and November 26, 2020; accepted November 28, 2020. Date of publication December 21, 2020; date of current version December 2, 2021. This work was supported in part by the National Major Program for Technological Innovation 2030—New Generation Artificial Intelligence of China under Grant 2018AAA0100500, in part by the National Key Research and Development Program of China under Grant 2018YFB0505000, and in part by the National Natural Science Foundation of China under Grant 62071421. *(Corresponding author: Yuntao Qian.)*

Jingzhou Chen, Shihao Wang, Ling Chen, and Yuntao Qian are with the College of Computer Science, Zhejiang University, Hangzhou 310027, China (e-mail: ytqian@zju.edu.cn).

Haibin Cai is with the School of Software Engineering, East China Normal University, Shanghai 200062, China.

Digital Object Identifier 10.1109/TGRS.2020.3042554

Several works have been proposed to avoid catastrophic forgetting. One important approach is adding regularization [16] to the weights of the model according to their importance to the old classes. Another research direction is knowledge distillation [30], which uses the new data to distill the knowledge from the old model to mimic its behavior when training the new model [17], [18]. Notably, Learning without Forgetting (LwF) [17] expands a new branch in the last layer of the network for classifying the new classes, and the old branch outputs the results of the old classes. The new branch could be learned with the training samples of new classes while sharing parameters of previous layers with the old branch. It can well copy with the problem of catastrophic forgetting. LwF first trains a classification model by using the images of old classes and stores it for the subsequent training process. We denote the stored model as the frozen model. Then, LwF encourages the outputs of the old branch trained with the new classes to approximate the outputs of the frozen model by applying knowledge distillation.

In this article, inspired by LwF, we propose a new architecture for class-incremental object detection based on feature pyramid and knowledge distillation. In the proposed network, FPN is used to detect objects in different layers of the feature pyramid, thus obtaining better detection results for remote sensing images teemed with objects of various scales. In addition to horizontal box regression, a rotated bounding box is also introduced to tackle the arbitrary orientation and dense arrangement of objects in remote sensing images [14], [31]. There exist other tailored methods [10], [32]–[34] for localization of objects, which can be borrowed instead of the horizontal and rotated bounding box methods in our proposed model. Incremental learning with knowledge distillation is embodied in the architecture from two aspects. First, the branch expansion is used to build a new branch in the last layer of FPN for detecting the new classes. Second, knowledge distillation is applied to the outputs of the old branch to preserve the knowledge learned in the old classes. The multitask learning is adopted for jointly optimizing the distillation loss from the old branch and the detection loss from the new branch. Experimental results demonstrate the effectiveness of our method on two popular data sets: DOTA and DIOR. To the best of our knowledge, this is the first study to address the incremental target detection problem in the remote sensing field, in which the old model is incrementally updated only with a training set of new object classes, while the training samples of old classes are not required.

The rest of this article is organized as follows. Section II briefly reviews the related works. Section III clarifies the proposed method, followed by the details of the implementation. Section IV presents the experiments conducted on two commonly used remote sensing data sets to validate the effectiveness of our approach. Section V further discusses the results of the proposed method. Finally, we conclude this article in section VI.

II. Related Work

In general, deep learning-based object detection approaches can be divided into two- and one-stage methods. Two-stage methods first produce region proposals for subsequent fine-grained classification and regression. One-stage methods directly output final results without producing intermediate region proposals. For two-stage methods, inspired by LwF, [19] is the first work of introducing incremental learning into object detection based on Fast R-CNN. References [20] and [21] follow similar ideas and further extend to Faster R-CNN. Faster R-CNN introduces the region proposal network (RPN) to generate region proposals and achieves better performance. The key problem of such extension is how to adapt previous knowledge for generating the class-agnostic region proposals of the current new classes. Reference [19] uses EdgeBoxes [35] and multiscale combinatorial grouping (MCG) [36] to generate such proposals by grouping extracted edges and performing hierarchical image segmentation, respectively. Reference [20] measures the mean squared error of the feature map before the classifier in RPN between the frozen model and the current training model trained on the new classes. In addition to measuring the discrepancy of feature maps between two models, [21] computes the difference of classification and regression outputs in RPN by using knowledge distillation. However, it is difficult to choose the appropriate intermediate feature map to measure the discrepancy, as feature maps in different layers of the network correspond to different receptive fields. Moreover, all these works have proved that the branch expansion could better model distinct distributions between the old and the new classes compared with the single branch. Two-stage methods usually obtain higher accuracy, while one-stage methods have faster inference speed. Based on one-stage methods, [22] distills three types of knowledge from the frozen model to mimic its behavior on classification, regression, and feature extraction. Unlike knowledge distillation, [23] presents an alternative way by mining memory neurons in the old model, and then, the rest neurons are updated and employed to detect new classes. Our method adopts a two-stage network FPN as the backbone structure, expands a new branch in the output layer for detecting new classes, and utilizes knowledge distillation to transfer measure the knowledge of the frozen model into the old branch.

In the field of remote sensing, several attempts have been made to develop incremental learning methods for classifying optical, SAR, and hyperspectral images [37]–[42]. Compared with classification task, only a few works focus on incremental object detection problem. Most of them are proposed to improve the detector with arriving new samples from the old classes [24], [43], rather than adapting to unseen new object classes. Furthermore, several specific characteristics pronounced for images in remote sensing should be given more attention: 1) the images often contain objects of various scales overwhelmed by complex surrounding scenes; 2) the targets are often densely arranged, such as vehicles and ships; and 3) the objects could appear in arbitrary orientations. Therefore, class-incremental object detection in remote sensing images is still a challenging problem. With consideration of these pronounced characteristics, our proposed incremental object detection method uses some specific strategies, such as multiscale feature map and rotated bounding box.

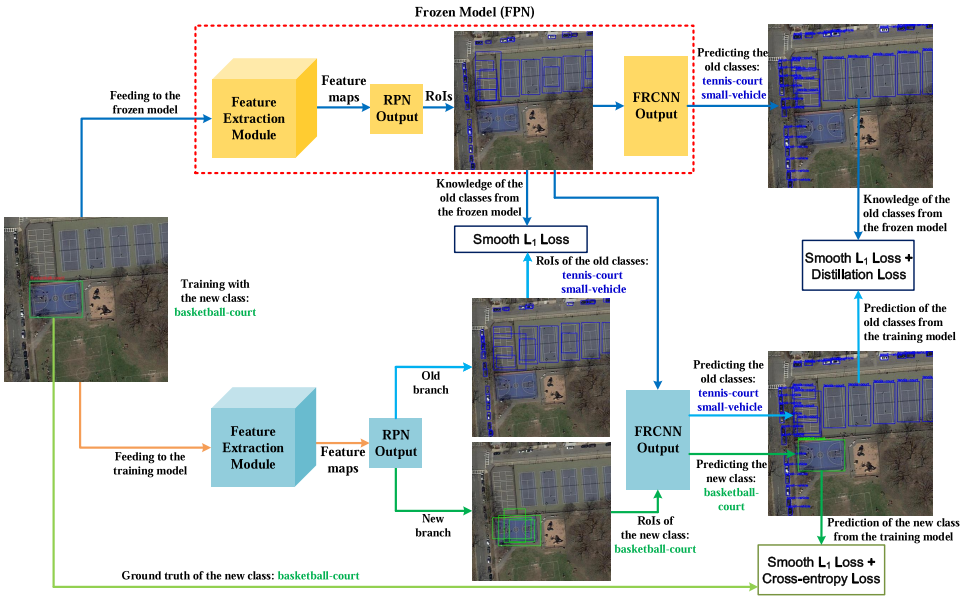


Fig. 1. Framework for learning object detectors incrementally. It is composed of a frozen copy of the old detector (frozen model) for the old classes and the new detector (training model) adapted for the new classes.

III. Proposed Method

Given an object detection model that is well-trained with the old set of classes C_o, our goal is to update the model with only images of the new set of classes C_n, and $C_n \cap C_o = \phi$. The model of our incremental detection method is illustrated in Fig. 1. We train an FPN to detect the old classes C_o and store it as the frozen model. Derived from the frozen model, the model is trained to detect the new classes C_n and maintain the original detection capability with the frozen model. The updated model should work well on all the observed classes $C = C_o \cup C_n$. For simplicity, we denote the tasks of detecting the old and new classes as the old and new tasks, respectively.

The proposed network includes three main components: the feature extraction module, the RPN output, and the FRCNN output. The feature extraction module is composed of consecutive convolution layers for transferring an input image into feature maps. The RPN output represents the output layer in RPN, and the FRCNN output refers to the last output layer in the network. For detecting new classes, we expand a new branch in the RPN output and the FRCNN output. With the aid of the frozen model, the old branch tries to mimic the outputs from the frozen model to preserve the knowledge learned from the old classes. Especially, we utilize a smooth L_1 loss to measure the discrepancy of outputs between the frozen model and the old branch in the RPN output. With regions of interest (RoIs) extracted from the frozen model as input, the outputs of the FRCNN output approximate the corresponding outputs in the frozen model by applying knowledge distillation. The expanded new branch is trained in the same way as FPN, i.e., smooth L_1 loss for regression and cross-entropy loss for classification.

In the following sections, we explain the details of the network and its implementation. First, we clarify the specific structure of FPN, which is used as the backbone structure of our network. Second, we introduce the regression of the horizontal and rotated bounding boxes, respectively. Third, the branch expansion for new target classes is discussed. Then, we demonstrate how to perform incremental training with knowledge distillation. Finally, the multitask learning strategy is adopted to maintain the knowledge of the old task and commit detection on the new task simultaneously.

A. FPN

Since objects in remote sensing images often appear across a large range of scales, we adopt FPN as the backbone structure, which detects objects of different scales in the different layers of the feature pyramid. Its architecture is shown in Fig. 2. The RPN output and the FRCNN output refer to the last fully-connected layers used for regressing and classifying anchors and RoIs, respectively. In the feature extraction module, the input image is processed by consecutive convolution layers corresponding to different receptive fields. The shallow convolution layers merge the abstract semantic information from deeper layers by upsampling. Such top–down architecture with lateral connections could help build high-level semantic feature maps at all scales. The RPN output produces class-agnostic RoIs containing the objects with various scales based on the feature maps extracted from these convolution layers in different depths. The RoI pooling used in the conventional FPN conducts coarse spatial quantization for feature extraction, which hinders the model from locating small-size objects. We replace RoI pooling with RoI align that faithfully preserves exact spatial locations. After RoIs have been collected, distributed, and fed to the RoI align layer, the FRCNN output receives these RoIs and performs class-specific regression and classification.

B. Horizontal and Rotated Object Detection

The RPN output provides coarse proposals to the second stage for subsequent processing. In the second stage,

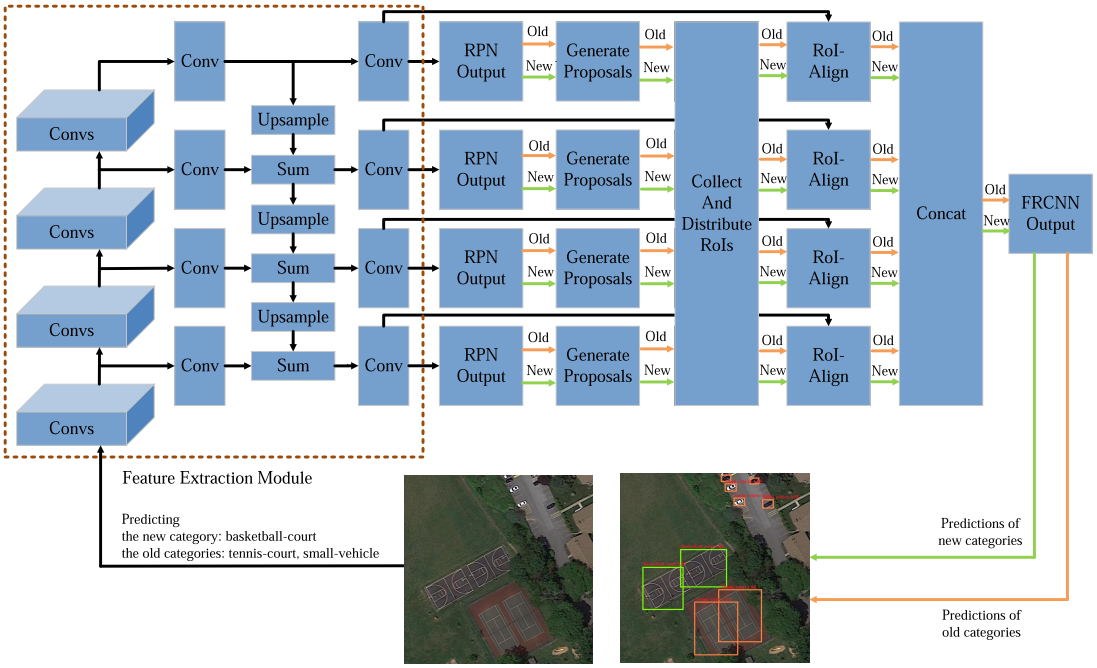


Fig. 2. Overall architecture of the FPN.

traditional methods perform bounding box regression with horizontal coordinates [5]

$$t_x = (x - x_a)/w_a, \quad t_y = (y - y_a)/h_a$$
$$t_w = \log(w/w_a), \quad t_h = \log(h/h_a). \tag{1}$$
$$t'_x = (x' - x_a)/w_a, \quad t'_y = (y' - y_a)/h_a$$
$$t'_w = \log(w'/w_a), \quad t'_h = \log(h'/h_a) \tag{2}$$

where x, y, w, and h denote the two coordinates of the box center, width, and height, respectively. Variables x, x_a, and x' denote the predicted box, anchor box, and ground-truth box respectively (likewise for y, w, and h). This can be thought of as bounding-box regression from an anchor box to a nearby ground-truth box. However, it will suppress the detection of densely arranged objects with arbitrary orientations over the horizontal line during the NMS postprocessing. Therefore, based on the horizontal representation, we use (x, y, w, h, θ) to represent arbitrary-oriented rectangle, where θ is defined as the acute angle to the x-axis, ranging in $[-\pi/2, 0]$. When employing rotated coordinates, we need to calculate the intersection of the union (IoU) between skew bounding boxes in the NMS step. To deal with this problem, [31] proposes an implementation of skew IoU computation with thought to triangulation. Here, the skew IoU computation-based rotation nonmaximum suppression (R-NMS) is used as an alternative postprocessing operation. For the regression of the rotation bounding box, we follow the same way used in the aforementioned horizontal regression and further introduce regression for θ [14]

$$t_\theta = \theta - \theta_a, \quad t'_\theta = \theta' - \theta_a. \tag{3}$$

Likewise, θ, θ_a, and θ' denote the predicted box, anchor box, and ground-truth box, respectively.

C. Branch Expansion

In order to adapt the model to detect the new classes, we could expand a new branch to detect the objects of new classes. Analogously to [19], we add sibling layers for the new classes in the last fully-connected layers for object classification and bounding box regression. Especially, as displayed in Fig. 1, the RPN output and FRCNN output expand a new branch, i.e., fully-connected layers for regressing and classifying objects of the new classes. The new layers are initialized randomly in the same way as the corresponding layers in FPN. On the other hand, the object distributions between the old and the new tasks may be largely different. Creating a separate branch could help the model to keep its distinct distribution.

D. Knowledge Distillation

In addition to adding the new branch for detecting the objects of new classes, we want to preserve the detection ability of the old task and detect the old classes in the old branch. With only the images of new classes, the outputs of the old branch should approximate the outputs of the frozen model in both the RPN output and the FRCNN output. For the RPN output, we first record outputs of the frozen model including classification output $P_{\text{frozen\_RPN}}$ and regression output $B_{\text{frozen\_RPN}}$. Combining $P_{\text{frozen\_RPN}}$ with $B_{\text{frozen\_RPN}}$, the frozen model performs nonmaximum suppression (NMS) and generates $\text{RoIs}_{\text{frozen}}$. The RoI align layer transforms $\text{RoIs}_{\text{frozen}}$ into the fixed size and feeds them into the FRCNN output. When updating the model, we enforce classification output $P_{\text{old\_RPN}}$ and the regression output $B_{\text{old\_RPN}}$ of the old branch in the RPN output to approach the recorded $P_{\text{frozen\_RPN}}$ and $B_{\text{frozen\_RPN}}$, respectively. The smooth L_1 loss is used to

measure the discrepancy, and the approximation loss in the RPN output is defined as

$$\begin{aligned}\text{Loss}_{\text{old\_RPN}} &= \text{smooth } L_1\ (|P_{\text{frozen\_RPN}} - P_{\text{old\_RPN}}|) \\ &\quad + \text{smooth } L_1\ (|B_{\text{frozen\_RPN}} - B_{\text{old\_RPN}}|). \end{aligned} \tag{4}$$

The smooth L_1 loss is a robust L_1 loss that is less sensitive to outliers than the L_2 loss, which is defined as

$$\text{smooth } L_1(x) = \begin{cases} 0.5x^2, & \text{if } |x| < 1, \\ |x| - 0.5, & \text{otherwise.} \end{cases} \tag{5}$$

Similarly, in the FRCNN output, we enforce the outputs of the old branch in the training model to mimic the corresponding outputs in the frozen model. The FRCNN output receives RoIs as its input to perform fine-grained classification and regression. There exist three different choices of RoIs that fed into the old branch of the FRCNN output in the current training model: $\text{RoIs}_{\text{frozen}}$ from the frozen model, RoIs_{old} from the old branch, and RoIs_{new} from the new branch. In the experiments, we evaluate the different RoIs selection methods and find that $\text{RoIs}_{\text{frozen}}$ is the best among them. The classification output $P_{\text{frozen\_FRCNN}} = [\hat{p}_1, \hat{p}_2, \ldots, \hat{p}_m]$, where m stands for the number of the old classes, and the regression output $B_{\text{frozen\_FRCNN}}$ of the FRCNN output in the frozen model is also recorded. With $\text{RoIs}_{\text{frozen}}$ as input, the old branch of the FRCNN output produces the classification output $P_{\text{old\_FRCNN}} = [p_1, p_2, \ldots, p_m]$ and the regression output $B_{\text{old\_FRCNN}}$. We modify the softmax probabilities $\hat{p}_i$ and p_i into

$$\hat{p}_i' = \frac{\hat{p}_i^{1/T}}{\sum_i^m \hat{p}_i^{1/T}}$$

and

$$p_i' = \frac{p_i^{1/T}}{\sum_i^m p_i^{1/T}}$$

respectively, where T is the hyperparameter of temperature. Such modification could increase the weight of smaller logit values and encourage the network to better encode similarities among classes. The cross-entropy loss is applied to these modified softmax probabilities, and we acquire the following cross-entropy distillation loss:

$$\text{CE\_dist}\ (P_{\text{frozen\_FRCNN}}, P_{\text{old\_FRCNN}}) = -\sum_i^m \hat{p}_i' \log p_i' \tag{6}$$

where $P_{\text{frozen\_FRCNN}}$ and $P_{\text{old\_FRCNN}}$ are the classification outputs. The approximation loss in the FRCNN output is defined as

$$\begin{aligned}&\text{Loss}_{\text{old\_FRCNN}} \\ &= \text{smooth } L_1\ (|B_{\text{frozen\_FRCNN}} - B_{\text{old\_FRCNN}}|) \\ &\quad + \text{CE\_dist}\ (P_{\text{frozen\_FRCNN}}, P_{\text{old\_FRCNN}}). \end{aligned} \tag{7}$$

The overall approximation loss used for preserving the knowledge of the old task is

$$\text{Loss}_{\text{old}} = \text{Loss}_{\text{old\_RPN}} + \text{Loss}_{\text{old\_FRCNN}}. \tag{8}$$

E. Multitask Training

Multitask learning is used to complete the first task of detecting the objects of new classes and the second task of preserving the detection ability of the old classes. Following the commonly used setting in FPN, the loss of the first task used for training is the combination of cross-entropy loss and smooth L_1 loss, respectively, from the RPN output and the FRCNN output of the expanded new branch

$$\begin{aligned}\text{Loss}_{\text{new}} &= \text{cross} - \text{entropy}\ (P_{\text{new\_RPN}}, P_{\text{new\_GT}}) \\ &\quad + \text{smooth } L_1\ (|B_{\text{new\_RPN}} - B_{\text{new\_GT}}|) \\ &\quad + \text{cross} - \text{entropy}\ (P_{\text{new\_FRCNN}}, P_{\text{new\_GT}}) \\ &\quad + \text{smooth } L_1\ (|B_{\text{new\_FRCNN}} - B_{\text{new\_GT}}|) \end{aligned} \tag{9}$$

where $P_{\text{new\_GT}}$ and $B_{\text{new\_GT}}$ represent the one-hot label and the bounding box of the ground truth on the new task. $P_{\text{new\_RPN}}$, $P_{\text{new\_FRCNN}}$, $B_{\text{new\_RPN}}$, and $B_{\text{new\_FRCNN}}$ are classification and regression outputs of the new branch in the RPN output and the FRCNN output, respectively.

The loss of the second task for training the old branch in the model is defined in (8). Therefore, the total loss is defined as

$$\text{Loss}_{\text{total}} = \lambda_o \text{Loss}_{\text{old}} + \text{Loss}_{\text{new}}. \tag{10}$$

where λ_o is the hyperparameter that balances the significance between the old and new tasks. The standard stochastic gradient descent (SGD) with momentum is applied to optimize the model parameters. Notably, the RPN output and the FRCNN output share the feature extraction module, and SGD optimizes all parameters of the updating model including the feature extraction module, the RPN output, and the FRCNN output. During training, we first train the parameters of the new branch both in the RPN output and the FRCNN output to converge while fixing the rest parameters. We refer to this step as the warm-up step. After the warm-up step, the whole parameters in the model are further optimized. The experiments will verify the effectiveness of the warm-up step. The overall training procedure is summarized in Algorithm 1.

IV. EXPERIMENTS

In this section, we conduct experiments to evaluate the proposed method on two remote sensing image data sets: DOTA [11] and DIOR [12]. In the ablation study, some crucial factors in the training process are evaluated, e.g., the choice of RoIs used for calculating the outputs of the old branch in the FRCNN output and the choice of loss functions used for measuring the distance between two probability distributions in the RPN output. In comparative experiments, two state-of-the-art incremental object detection methods in [19] and [20] are used as reference methods. Our method is implemented on the Detectron platform.[1] Regarding the training algorithm, we empirically set the batch size as 2, the momentum as 0.9, and the number of iterations as 60 000. The initial learning rate is 0.0025 and multiplies 0.1 every 20 000 iterations. The detection results are recorded when IoU between the predicted

[1] https://github.com/facebookresearch/Detectron

Algorithm 1 Proposed Incremental Training Algorithm for Object Detection

Input:
X_n: images of the new task
P_n, B_n: the label and the bounding box of the ground truth on the new task
CNN_{frozen}: the frozen model trained on the old task
Output: Optimal parameters θ^* of the updating model $CNN_{updating}$
Initialize:
P_{frozen}, B_{frozen}: classification outputs and regression outputs of the frozen model $CNN_{frozen}(X_n)$ with X_n as input
$RoIs_{frozen}$: RoIs produced from the RPN Output in the frozen model $CNN_{frozen}(X_n)$
Warm-up Step:
Randomly initialize parameters of the new branch in the $CNN_{updating}$
repeat
  Using P_n and B_n, training parameters of the new branch and fixing the rest parameters in the $CNN_{updating}$
until convergence
Training Step:
repeat
  Updating whole parameters θ of the $CNN_{updating}$ by $\arg\min_\theta(\lambda_o Loss_{old} + Loss_{new})$
until convergence
return θ^*

bounding box and the annotated bounding box is equal to or greater than 0.5. Unless otherwise specified, the following experiments employ the horizontal bounding box, as the compared incremental object detection methods are based on the horizontal box regression, and only the DOTA data set contains the rotated box annotations. However, we give a separate experiment on DOTA to compare the horizontal and rotated bounding boxes for object location.

A. Remote Sensing Data Sets

DOTA [11] contains 2806 images and 188 282 instances annotated with horizontal and rotated bounding boxes, divided into the training set (1411 images), the validation set (458 images), and the test set (937 images). The image size ranges from 800 × 800 to 4000 × 4000, and we crop it into patches with a size of 800 × 800 in our experiments. DOTA includes 15 categories: plane, baseball-diamond (BD), bridge, ground-track-field (GTF), small-vehicle (SV), large-vehicle (LV), ship, tennis-court (TC), basketball-court (BC), storage-tank (ST), soccer-ball-field (SBF), turntable (TA)/roundabout, harbor, swimming-pool (SP), and helicopter (HC). In our class-incremental setting, the first eight classes are taken as the old classes for the old task, while the latter seven classes are the new classes belonging to the new task. The training set is used to train our model, while the validation set is used as our testing set as the label of the original test set is unavailable.

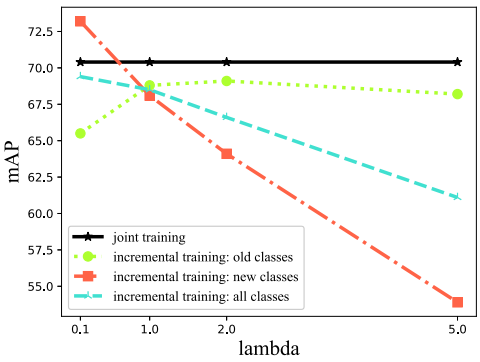


Fig. 3. mAP performance on the DIOR data set as the value of λ_o changes.

DIOR [12] includes 11 738 images with the size of 800 × 800, divided into the training set (5862 images), the validation set (5853 images), and the test set (11 738 images). DIOR annotates 190 288 instances with horizontal bounding boxes and incorporates 20 classes: airplane, baseball-field (BD), bridge, GTF, vehicle, ship, TC, airport, chimney, dam, BC, ST, harbor, expressway toll station (ETS), expressway service area (ESA), golf course (GC)/golf field, overpass, stadium, train station (TS), and windmill (WM). The first ten classes are used for the old task, while the latter ten classes are regarded as the new classes of the new task. We combine the training set and the validation set to train our model and test the model on the test set.

B. Ablation Study

1) Different Values of λ_o: The hyperparameter λ_o in loss function (10) balances the importance of the old and new tasks. We perform experiments on the DIOR data set with the different values of λ_o to analyze its influence on the old and new classes. The results shown in Fig. 3 demonstrate that, when the weight of old classes is little ($\lambda_o = 0.1$), the new classes are easier to learn, but the old classes are easily forgotten. When the weight of old classes is large ($\lambda_o = 2, 5$), it impedes learning the new classes but well preserves the detection capability on old classes. As λ_o becomes to be larger, the performance of new classes decreases quickly, but the performance of old classes increases slowly. When $\lambda_o = 1$, the average results of old and new classes approximate the corresponding joint-training counterparts. $\lambda_o = 1$ is a good tradeoff between learning new classes and preventing catastrophic forgetting, so, in all other experiments in this article, λ_o is fixed to 1.

2) Different Incremental Learning Strategies: When encountering the class-incremental problem, we have several strategies to update the old model and its parameters. Fig. 4 shows the four typical incremental learning methods, where θ_s represents parameters of the shared feature extraction module, θ_o stands for parameters of the output branch for old classes, and θ_n denotes parameters of the output branch for new classes. Case 1 only fine-tunes the parameters of the new branch, θ_n, while the rest parameters stay unchanged. In addition to θ_n, case 2 further fine-tunes the parameters of the shared feature extraction module, θ_s. Case 3 updates whole

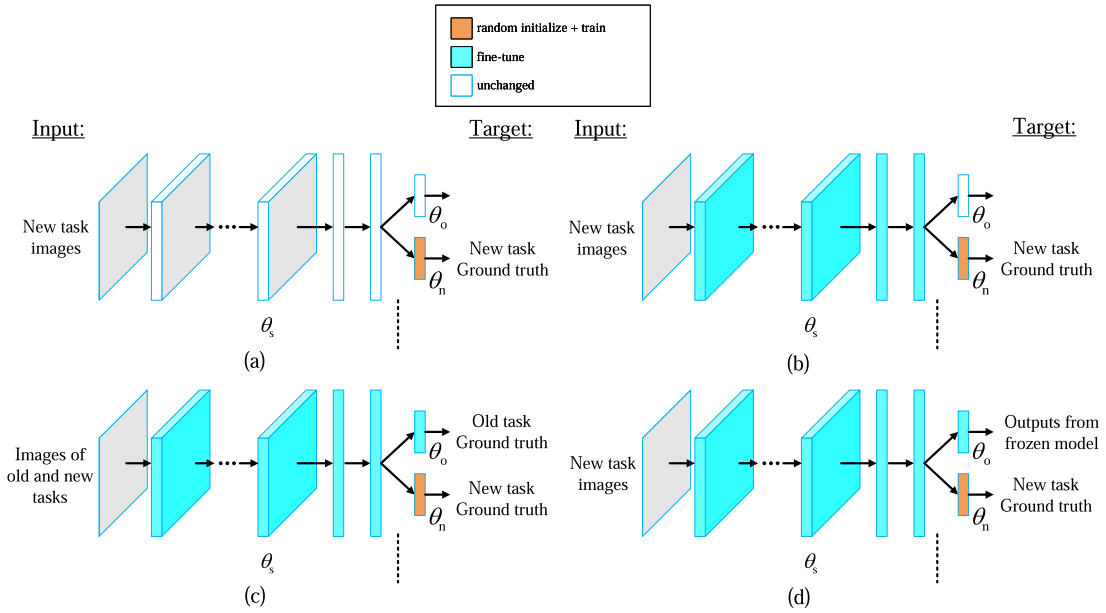


Fig. 4. Different training strategies of class-incremental learning. θ_s represents the parameters of the shared feature extraction module, θ_o stands for the parameters of the output branch for old classes, and θ_n denotes the parameters of the output branch for new classes. Case 1: only the parameters of the new branch are updated. Case 2: the parameters of the feature extraction module and the new branch are updated. Case 3: the whole parameters are updated with joint-training. Case 4: the whole parameters are updated with the proposed incremental learning method.

TABLE I
DETECTION RESULTS (MAP) WITH DIFFERENT CLASS-INCREMENTAL LEARNING STRATEGIES

Datasets	DOTA		DIOR	
Categories	Old 8	New 7	Old 10	New 10
Case 1	69.5	20.9	69.2	28.4
Case 2	1.8	60.0	15.6	**72.4**
Case 3	**69.8**	**60.8**	**69.4**	71.3
Case 4	69.2	60.7	68.8	68.1

TABLE II
DETECTION RESULTS (MAP) WITH DIFFERENT ROIS FED INTO THE OLD BRANCH IN THE FRCNN OUTPUT

Datasets	DOTA		DIOR	
Categories	Old 8	New 7	Old 10	New 10
$RoIs_{frozen}$	**69.2**	**60.7**	68.8	**68.1**
$RoIs_{old}$	66.2	58.3	68.3	68.1
$RoIs_{new}$	68.3	58.6	**68.9**	67.1

[1] $RoIs_{frozen}$ is produced from the RPN Output in the frozen model;
[2] $RoIs_{old}$ are extracted from the old branch in the RPN Output;
[3] $RoIs_{new}$ are extracted from the new branch in the RPN Output.

parameters from scratch by utilizing the training samples of old and new classes. With the proposed incremental learning algorithm, case 4 trains all parameters with outputs from the frozen model and the training samples of the new classes. The results are displayed in Table I, from which we can find that our incremental detection method without the training samples of old classes (Case 4) achieves similar results as the method of retraining model from scratch (Case 3), and both Cases 1 and 2 have very poor detection performance for old classes or new classes. Case 3, also represented as joint-training, is time-consuming due to learning from scratch and requires massive storage due to keeping all training samples from both old and new classes in hand. However, it is the upper bound for the class-incremental learning problem, so we used it as a baseline method in the later comparative experiments.

3) Different Regions of Interest: To preserve the detection capability learned on the old task, the outputs of the old branch should approximate the outputs of the frozen model with the only images of new classes. When computing the outputs of the old branch in the FRCNN output, we need to decide which aforementioned RoIs ($\text{RoIs}_{\text{frozen}}$, RoIs_{old}, and RoIs_{new}) to be used as the input. Table II displays the detection results of using different RoIs. In Table II, $\text{RoIs}_{\text{frozen}}$ achieves the best results compared with RoIs_{old} and RoIs_{new}. Therefore, we adopt $\text{RoIs}_{\text{frozen}}$ extracted from the frozen model as the input to the old branch of the FRCNN output in our experiments.

4) Effectiveness of the Warm-Up Step: In our training process, before we start to update the whole parameters of the model, the warm-up step first trains the parameters of the new branch both in the RPN output and in the FRCNN output to converge with the rest parameters fixed. To verify the effectiveness of the warm-up step, we replace it with Gaussian initialization. Notably, the numbers of total iterations of two initialization methods are set to be equal. Their results are exhibited in Table III, from which we can observe that the

TABLE III

DETECTION RESULTS (MAP) WITH DIFFERENT INITIALIZATIONS OF PARAMETERS IN THE NEW BRANCH AT IOU = 0.5

Datasets	DOTA		DIOR	
Categories	Old 8	New 7	Old 10	New 10
Warm-up	69.3	**61.5**	**69.2**	**69.2**
Random	**69.4**	60.9	69.1	68.8

TABLE IV

DETECTION RESULTS (MAP) WITH DIFFERENT LOSSES FOR MEASURING THE DISTANCE BETWEEN TWO PROBABILITY DISTRIBUTIONS IN THE RPN OUTPUT

Datasets	DOTA		DIOR	
Categories	Old 8	New 7	Old 10	New 10
Smooth L_1	69.2	**60.7**	**69.2**	**69.2**
Cross-Entropy	**69.5**	60.3	69.1	69.2

warm-up step helps to reach slightly better performance in most cases.

5) Different Losses for Measuring the Distance Between Two Probability Distributions in the RPN Output: Considering the binary classification in the RPN output, the sigmoid function is usually used to indicate the probabilities of the candidate proposals. To measure the distance between two probability distributions of the frozen model and the old branch of the incremental model, two popular loss functions, smooth L_1 loss and cross-entropy loss, are compared. Table IV displays the experimental results. From Table IV, it can be seen that two forms of losses achieve similar results, and smooth L_1 loss is slightly better than cross-entropy loss. Therefore, in our experiments, we adopt a smooth L_1 loss.

6) Algorithm Stability: To validate the stability of the proposed algorithm, we randomly assign old and new classes from the DIOR data set and repeat class-incremental experiments five times independently. The detection results (mAP) of old, new, and entire classes, respectively, are shown in Table V. Although the results on old and new classes vary to some extent along with the assignment of old and new classes, the results on entire classes are very similar, and their average mAP approximates that of the joint-training method.

C. Incremental Detection With Rotated Box Regression

To tackle the problems of arbitrary orientation and dense arrangement, we further implement the proposed incremental learning algorithm based on the rotated bounding box. Especially, we replace the horizontal coordinates with the rotated bounding box in the output layer and perform the rotation detection described in Section III-B. Since DIOR lacks the rotated box annotations, we conduct the comparative experiments on the DOTA data set. Table VI shows the experimental results, in which MAPs are calculated with the horizontal box annotations and the rotated box annotations, respectively, for horizontal and rotated detections. Fig. 5 shows an example of

TABLE V

DETECTION RESULTS (MAP) BY RANDOMLY ASSIGNING OLD AND NEW CLASSES FROM DIOR

	Old Classes	New Classes	All Classes
1st	68.8	68.1	68.5
2nd	69.3	65.4	67.4
3rd	74.0	63.7	68.9
4th	74.4	64.3	69.4
5th	73.4	66.7	70.1
Mean±Std	72.0±2.4	65.6±1.6	68.9±0.9
Joint-training	69.4	71.3	70.4

TABLE VI

COMPARATIVE DETECTION RESULTS (MAP) BETWEEN ROTATED AND HORIZONTAL BOUNDING BOXES

Bounding Box	Learning stategy	DOTA	
		old 8	new 7
Horizontal	incremental	69.2	60.7
	joint-training	69.8	60.8
Rotated	incremental	71.3	54.0
	joint-training	**71.8**	**62.9**

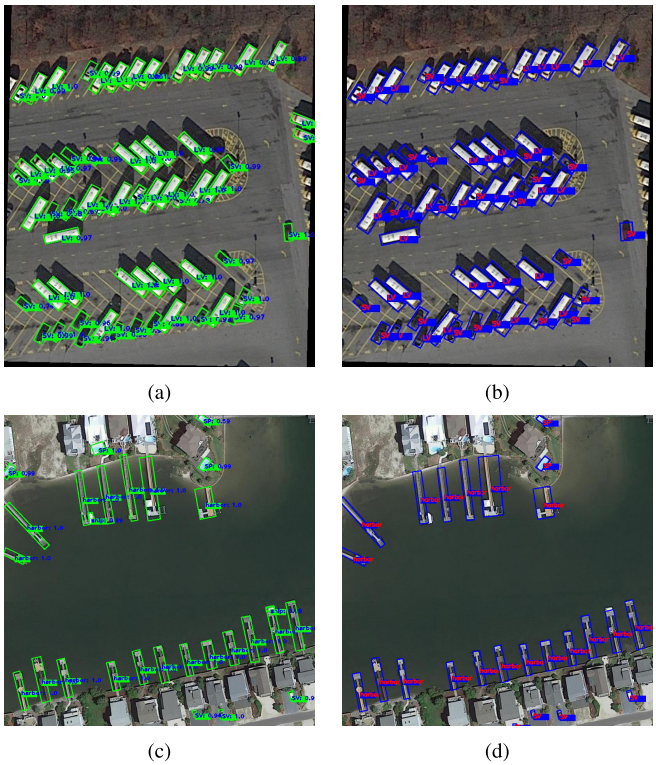


Fig. 5. Detection results of the proposed method with rotated bounding box on DOTA. (a) and (b) Results of the old task on LV and SV. (c) and (d) Results of the new task on harbor. (a) and (b) Rotated detection. (b) and (d) Ground truth.

incremental detection with the rotated bounding box, where two old classes, LV and SV, are arbitrarily orientated and closely arranged, and a new class harbor also appears in

TABLE VII
COMPARATIVE DETECTION RESULTS (MAP) OF PEER METHODS AT IOU = 0.5. DIFF MEANS THE DIFFERENCE OF EACH APPROACH FROM ITS CORRESPONDING JOINT-TRAINING RESULTS

Methods	Basic Architecture	Learning stategy	DOTA		DIOR	
			old 8	new 7	old 10	new 10
Fast-IL [19]	Fast R-CNN	incremental	26	13	31	19.4
		joint-training	26.8	22.6	33.5	34.1
		Diff	-0.8	-9.6	-2.5	-14.7
Faster-IL [20]	Faster R-CNN	incremental	36.1	26.4	36.7	47.0
		joint-training	41.5	26.9	47.4	47.7
		Diff	-5.4	-0.5	-10.7	-0.7
FPN-IL-our	FPN	incremental	**69.2**	**60.7**	**68.8**	**68.1**
		joint-training	**69.8**	**60.8**	**69.4**	**71.3**
		Diff	-0.6	-0.1	-0.6	-3.2

TABLE VIII
DETECTION RESULTS ON EACH OLD CLASS OF DOTA

Methods	Learning stategy	Old 8 categories								mAP
		Plane	BD	Bridge	GTF	SV	LV	Ship	TC	
Fast-IL [19]	incremental	48.5	26.7	2.9	31.4	3.1	20.7	16.8	57.5	26
	joint-training	50.7	28.3	4.4	32.4	3.1	20.7	15.7	58.9	26.8
	Diff	-2.2	-1.6	-1.5	-1	0	0	+1.1	-1.4	-0.8
Faster-IL [20]	incremental	57.9	36.9	18.1	25.4	14.0	29	28.5	79.2	36.1
	joint-training	66.0	47.5	31.0	21.0	15.1	39.6	31.2	80.9	41.5
	Diff	-8.1	-10.6	-12.9	+4.4	-1.1	-10.6	-2.7	-1.7	-5.4
FPN-IL-our	incremental	**79.6**	**66.2**	**46.6**	**59.1**	**53.7**	**75.2**	**81.9**	**91.2**	**69.2**
	joint-training	**80.9**	**69.1**	**49.4**	**57.4**	**53.8**	**76.4**	**80.4**	**90.8**	**69.8**
	Diff	-1.3	-2.9	-2.8	+1.7	-0.1	-1.2	+1.5	+0.4	-0.6

arbitrary orientations. It can be found that the detection with the rotated bounding box is suitable for detecting objects with arbitrary orientation and dense distribution.

D. Comparison With State-of-the-Art Methods

In order to show the advantage of the proposed method over other related works in class-incremental object detection, two state-of-the-art methods [19], [20] are chosen for evaluation. Compared with our approach, the first approach (abbreviated as Fast-IL) [19] uses the traditional EdgeBoxes or MCG method for generating class-agnostic RoIs, and the second approach (abbreviated as Faster-IL) [20] reduces the discrepancy of the feature map before the classifier layer in RPN between the frozen model and the training model to preserve the old detection capability. The proposed method adopts FPN as the backbone structure (abbreviated as FPN-IL-our), as FPN can detect objects of various scales in different layers of the feature pyramid. It evolves the separate branches for the old and new tasks and computes the discrepancies of the classification and regression outputs in RPN between the frozen model and the training model. We carefully follow the specifications from these compared articles for their settings and parameters. Table VII records the detection results of all three compared methods, and Tables VIII, IX, X, and XI further elaborate on the results on each class.

In Table VII, concerning the joint-training outcomes, Faster R-CNN is better than Fast R-CNN by introducing RPN, and FPN reaches the best results. When adapting to the class-incremental problem, Fast-IL preserves the detection ability on the old task but misses many objects on the new task. Conversely, Faster-IL fails to maintain the detection capability on the old task but detects well on the new task compared with its corresponding joint-training results. In general, Faster-IL outperforms Fast-IL on both of the old and new tasks. The proposed method, FPN-IL-our, not only attains the best results but also is the closest to its joint-training counterpart, which demonstrates its effectiveness of incremental learning. As observed from Table VIII, Fast-IL and Faster-IL detect poorly on SV and LV. However, FPN-IL-our improves greatly on these two categories, especially for SV, which verifies that FPN-IL-our is more robust to different scales of targets. We can find similar detection results of the vehicle in Table X. Another finding in Table VIII and X is that FPN-IL-our detects better than Fast-IL and Faster-IL on ship that arranges densely and closely.

The proposed method is superior to other compared methods due to two factors: robustness to various object scales and withstanding to catastrophic forgetting. We exemplify these two factors and illustrate them in Figs. 6 and 7. Fig. 6(a)–(d) show the detection results of the old task on LV and SV in

TABLE IX
DETECTION RESULTS ON EACH NEW CLASS OF DOTA

Methods	Learning stategy	New 7 categories							mAP
		BC	ST	SBF	TA/RA	Harbor	SP	HC	
Fast-IL [19]	incremental	12.2	14.6	22.5	6.2	17.4	13.2	4.7	13
	joint-training	23.5	18.1	28.2	13.1	36.7	20.3	18.2	22.6
	Diff	-11.3	-3.5	-5.7	-6.9	-19.3	-7.1	-13.5	-9.6
Faster-IL [20]	incremental	18.3	29.1	19.8	30.2	42.4	35.4	9.7	26.4
	joint-training	14.8	32.4	14.8	36.7	45.1	36.1	8.1	26.9
	Diff	+3.5	-3.3	+5	-6.5	-2.7	-0.7	+1.6	-0.5
FPN-IL-our	incremental	**52.1**	**68.5**	**56.0**	**68.7**	**70.6**	**62.7**	**45.9**	**60.7**
	joint-training	**56**	**71.4**	**58.1**	**69**	**74.2**	**61.4**	**35.7**	**60.8**
	Diff	-3.9	-2.9	-2.1	-0.3	-3.6	+1.3	+10.2	-0.1

TABLE X
DETECTION RESULTS ON EACH OLD CLASS OF DIOR

Methods	Stategy	Old 10 categories										mAP
		Airplane	BD	Bridge	GTF	Vehicle	Ship	TC	Airport	Chimney	Dam	
Fast-IL [19]	incremental	22.4	46.7	7	36.3	6.5	9.1	35.5	45.4	66.7	34.2	31
	joint-training	22.4	49	10.2	43	7	7.8	38.7	52.6	69.7	35	33.5
	Diff	0	-2.3	-3.2	-6.7	-0.5	+1.3	-3.2	-7.2	-3	-0.8	-2.5
Faster-IL [20]	incremental	23.0	61.9	9.5	50.1	15.8	29.2	73.3	36.5	64.0	3.3	36.7
	joint-training	37.0	60.9	18.2	56.4	28.1	41.8	75.7	57.1	68.6	30.0	47.4
	Diff	-14.0	+1.0	-8.7	-6.3	-12.3	-12.6	-2.4	-20.6	-4.6	-26.7	-10.7
FPN-IL-our	incremental	**47.6**	**71.5**	**46.1**	**83.1**	**54.1**	**85.7**	**85.0**	**74.8**	**77.7**	**61.8**	**68.8**
	joint-training	**55.1**	**70.7**	**43.7**	**83**	**53.7**	**85.6**	**84.9**	**75.7**	**78.5**	**62.7**	**69.4**
	Diff	-7.5	+0.8	+2.4	+0.1	+0.4	+0.1	+0.1	-0.9	-0.8	-0.9	-0.6

the DOTA data set. Although the sizes of targets in these two old classes are largely different, our method still obtains the best results. Various object scales also appear in Fig. 6(e)–(h). Our method not only detects each instance of BC from the new task but also discovers each instance of TC and SV from the old task. Similar findings can be observed from Fig. 7 on the DIOR data set. Fig. 7(a)–(d) display the detection results on three old classes: baseball-field (BD), ground-track-field (GTF), and TC. It is obvious that our method successfully avoids catastrophic forgetting and is more robust to different object scales compared with other methods. As observed from Fig. 7(e)–(h), our model successfully locates each instance of ST from the new task, which is densely and closely arranged.

V. DISCUSSION

In the experiments, we first verify several pivotal factors of the proposed method and then compare our method with two state-of-the-art incremental object detection approaches on two commonly used remote sensing data sets. To overcome problems of objects appearing in arbitrary orientations and cluttered arrangements, we further replace the horizontal bounding box with the rotated bounding box in the proposed incremental learning algorithm. The results have demonstrated the effectiveness of our proposed method. However, our work in this article focuses on how to deal with the incremental learning problem without the training samples of the old classes, so the other techniques for further improving the detection performance are retained for future study.

The problem that we address here is also related to few-shot learning. Few-shot learning aims to adapt the model to recognize unseen novel classes using very few training samples, while the model's recognition performance on the old classes is not considered. Comparatively, class-incremental learning learns new classes and preserves prior knowledge of old classes. Class-incremental learning not only needs to achieve competitive performance in new classes like few-shot learning but also tries to avoid catastrophic forgetting. When the number of training samples of new classes is very few for class-incremental learning, few-shot learning is also required. Therefore, the combination of class-incremental learning and few-shot learning has broad applications, which learns new classes with a small number of training examples and preserves the previously learned knowledge of old classes without catastrophic forgetting [44], [45]. In our future work, the incremental few-shot learning in remote sensing image analysis will be a possible topic.

On the other hand, class-incremental learning under the fully supervised setting requires manually annotated bounding boxes around each object of new classes. It is generally

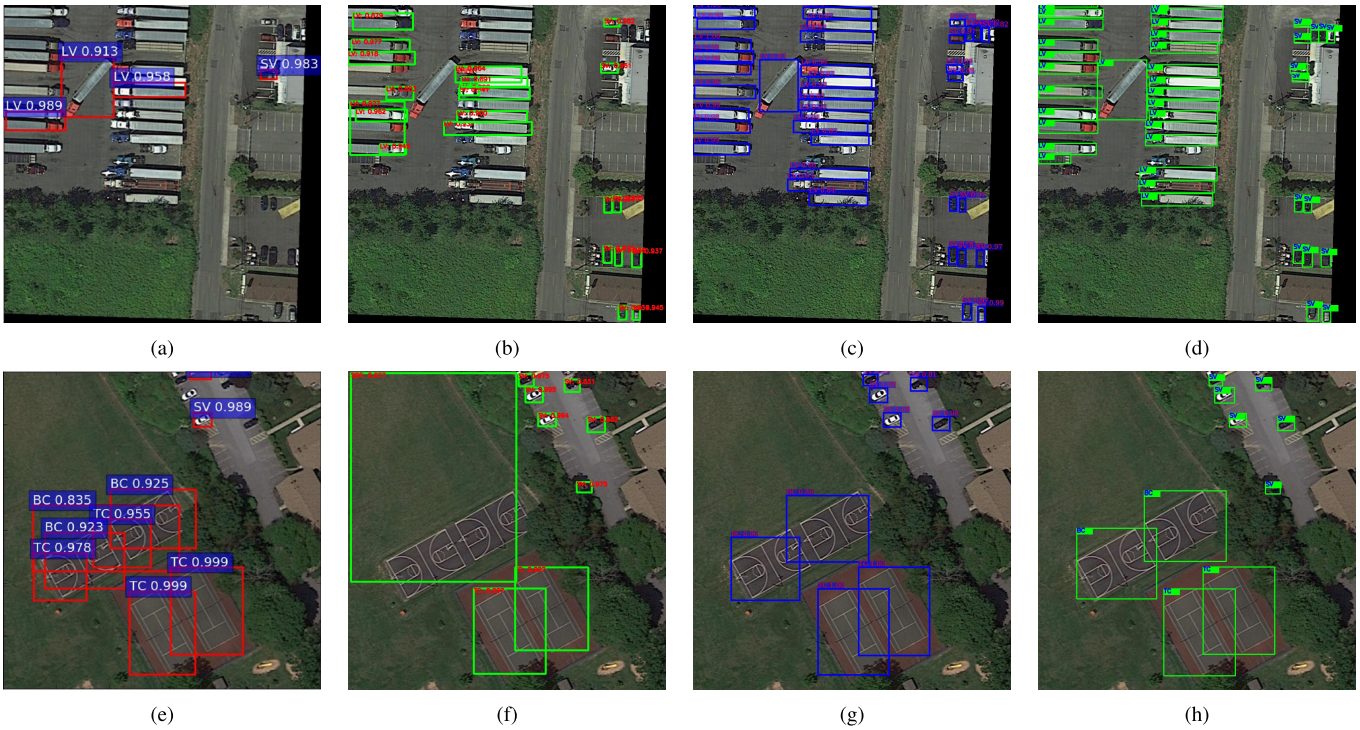


Fig. 6. Detection results on DOTA with three different methods. (a)–(d) Results of the old task on LV and SV. (e)–(h) Results on BC of the new task, SV of the old task, and TC of the old task (a) Fast-IL. (b) Faster-IL. (c) FPN-IL-our. (d) Ground truth. (e) Fast-IL. (f) Faster-IL. (g) FPN-IL-our. (h) Ground truth.

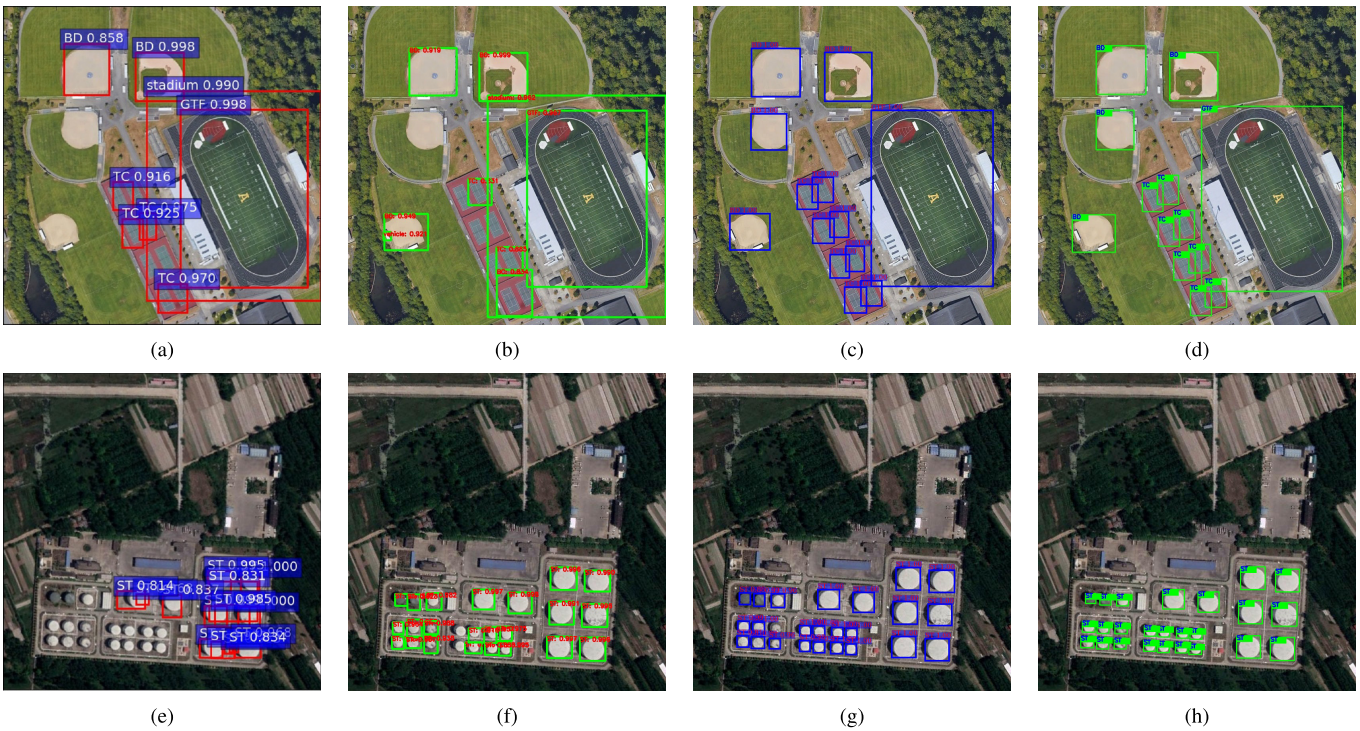


Fig. 7. Detection results on DIOR with three different methods. (a)–(d) Results of the old task on baseball-field (BD), GTF, and TC. (e)–(h) Results of the new task on storage-rank (ST). (a) Fast-IL. (b) Faster-IL. (c) FPN-IL-our. (d) Ground truth. (e) Fast-IL. (f) Faster-IL. (g) FPN-IL-our. (h) Ground truth.

time-consuming, expensive, and, sometimes, even unreliable to obtain such accurate manual annotation. With the aid of studies of weakly supervised object detection in remote sensing images [46], [47], a more feasible way is to employ class-incremental detection with image-level annotations, which is also a possible topic in our feature work.

TABLE XI
DETECTION RESULTS ON EACH NEW CLASS OF DIOR

Methods	Stategy	New 10 categories										mAP
		BC	ST	Harbor	ETS	ESA	GC	Overpass	Stadium	TS	WM	
Fast-IL [19]	incremental	33.7	10.3	12.5	23.8	23.5	39.7	12.8	27.5	1.4	9.2	19.4
	joint-training	58.2	13	27.9	33.2	43.7	66.9	20.7	51.1	5.4	21.4	34.1
	Diff	-24.5	-2.7	-15.4	-9.4	-20.2	-27.2	-7.9	-23.6	-4	-12.2	-14.7
Faster-IL [20]	incremental	70.0	29.3	17.9	33.4	54.0	62.8	43.2	48.1	38.0	72.9	47.0
	joint-training	70.4	29.8	16.4	37.9	53.8	64.3	42.8	47.2	39.5	74.9	47.7
	Diff	-0.4	-0.5	+1.5	-4.5	+0.2	-1.5	+0.4	+0.9	-1.5	-2.0	-0.7
FPN-IL-our	incremental	**86.8**	**58.3**	**59.1**	**66.9**	**73.3**	**71.0**	**60.1**	**69.2**	**49.6**	**86.8**	**68.1**
	joint-training	**87.2**	**69.8**	**63.2**	**65.9**	**80.4**	**81.6**	**57.2**	**61.2**	**58.3**	**88.3**	**71.3**
	Diff	-0.4	-11.5	-4.1	+1.0	-7.1	-10.6	+2.9	+8.0	-8.7	-1.5	-3.2

VI. CONCLUSION

In this article, we present a new class-incremental object detection approach under a deep learning frame, which learns to detect the new classes and maintains the ability to recognize the old classes in the absence of the original training data of the old classes. To overcome the problem of catastrophic forgetting, we expand the new branch for classification and regression of the objects in new classes and transfer the distilled knowledge from the frozen model learned from the old classes into the old branch. Using the multitask learning strategy, the total loss is minimized that balances the interplay between detecting the new classes and preserving the detection capability of the old classes. The proposed incremental training algorithm can be easily extended to other powerful deep learning-based detectors. Comparative experiments on the two remote sensing data sets demonstrate the superiority of our method, especially for the objects with various scales, arbitrary orientations, and dense distributions.

REFERENCES

[1] G. Cheng and J. Han, "A survey on object detection in optical remote sensing images," *ISPRS J. Photogramm. Remote Sens.*, vol. 117, pp. 11–28, Jul. 2016.

[2] T.-Y. Lin *et al.*, "Microsoft coco: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 740–755, doi: 10.1007/978-3-319-10602-1_48.

[3] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, Jun. 2010.

[4] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.

[5] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.

[6] T. 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.*, Jul. 2017, pp. 936–944.

[7] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6517–6525.

[8] W. Liu and *et al.*, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 21–37.

[9] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2999–3007.

[10] 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.

[11] G.-S. Xia *et al.*, "DOTA: A large-scale dataset for object detection in aerial images," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3974–3983.

[12] K. Li, G. Wan, G. Cheng, L. Meng, and J. Han, "Object detection in optical remote sensing images: A survey and a new benchmark," *ISPRS J. Photogramm. Remote Sens.*, vol. 159, pp. 296–307, Jan. 2020.

[13] X. Yang *et al.*, "Automatic ship detection in remote sensing images from Google Earth of complex scenes based on multiscale rotation dense feature pyramid networks," *Remote Sens.*, vol. 10, no. 1, p. 132, 2018.

[14] X. Yang *et al.*, "SCRDet: Towards more robust detection for small, cluttered and rotated objects," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 8231–8240.

[15] A. Gepperth and B. Hammer, "Incremental learning algorithms and applications," in *Proc. Eur. Symp. Artif. Neural Netw.*, 2016, pp. 357–368.

[16] K. James *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proc. Nat. Acad. Sci. USA*, vol. 114, no. 13, pp. 3521–3526, Mar. 2017.

[17] Z. Li and D. Hoiem, "Learning without forgetting," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 12, pp. 2935–2947, Dec. 2018.

[18] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "ICaRL: Incremental classifier and representation learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 5533–5542.

[19] K. Shmelkov, C. Schmid, and K. Alahari, "Incremental learning of object detectors without catastrophic forgetting," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 3420–3429.

[20] Y. Hao, Y. Fu, Y.-G. Jiang, and Q. Tian, "An end-to-end architecture for class-incremental object detection with knowledge distillation," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jul. 2019, pp. 1–6.

[21] L. Chen, C. Yu, and L. Chen, "A new knowledge distillation for incremental object detection," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2019, pp. 1–7.

[22] D. Li, S. Tasci, S. Ghosh, J. Zhu, J. Zhang, and L. Heck, "RILOD: Near real-time incremental learning for object detection at the edge," in *Proc. 4th ACM/IEEE Symp. Edge Comput.*, Nov. 2019, pp. 113–126.

[23] W. Li, Q. Wu, L. Xu, and C. Shang, "Incremental learning of single-stage detectors with mining memory neurons," in *Proc. IEEE 4th Int. Conf. Comput. Commun. (ICCC)*, Dec. 2018, pp. 1981–1985.

[24] L. Guan, Y. Wu, J. Zhao, and C. Ye, "Learn to detect objects incrementally," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2018, pp. 403–408.

[25] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.

[26] M. McCloskey and N. J. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," *Psychol. Learn. Motivat.*, vol. 24, pp. 109–165, Dec. 1989.

[27] I. J. Goodfellow, M. Mirza, D. Xiao, A. Courville, and Y. Bengio, "An empirical investigation of catastrophic forgetting in gradient-based neural networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2014.

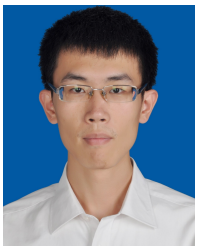
[28] R. French, "Dynamically constraining connectionist networks to produce distributed, orthogonal representations to reduce catastrophic interference," in *Proc. Annu. Cognit. Sci. Soc. Conf.*, vol. 5, 1994, pp. 207–220.

[29] Y. LeCun *et al.*, "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, Dec. 1989.

[30] G. E. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*. [Online]. Available: http://arxiv.org/abs/1503.02531
[31] J. Ma *et al.*, "Arbitrary-oriented scene text detection via rotation proposals," *IEEE Trans. Multimedia*, vol. 20, no. 11, pp. 3111–3122, Nov. 2018, doi: 10.1109/TMM.2018.2818020.
[32] W. Liu, L. Ma, and H. Chen, "Arbitrary-oriented ship detection framework in optical remote-sensing images," *IEEE Geosci. Remote Sens. Lett.*, vol. 15, no. 6, pp. 937–941, Jun. 2018.
[33] Z. Zhang, W. Guo, S. Zhu, and W. Yu, "Toward arbitrary-oriented ship detection with rotated region proposal and discrimination networks," *IEEE Geosci. Remote Sens. Lett.*, vol. 15, no. 11, pp. 1745–1749, Nov. 2018.
[34] Z. Tian, C. Shen, H. Chen, and T. He, "FCOS: Fully convolutional one-stage object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9626–9635.
[35] C. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 391–405.
[36] P. Arbelaez, J. Pont-Tuset, J. Barron, F. Marques, and J. Malik, "Multiscale combinatorial grouping," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 328–335.
[37] L. Burzzone, D. Fernandez Prieto, and R. Cossu, "An incremental learning classifier for remote-sensing images," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jun./Jul. 1999, pp. 2483–2485.
[38] A. Vetrivel, N. Kerle, M. Gerke, F. Nex, and G. Vosselman, "Towards automated satellite image segmentation and classification for assessing disaster damage using data-specific features with incremental learning," in *Proc. GEOBIA : Solutions Synergies*, Sep. 2016, pp. 1–5.
[39] S. Dang, Z. Cao, Z. Cui, Y. Pi, and N. Liu, "Open set incremental learning for automatic target recognition," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 7, pp. 4445–4456, Jul. 2019.
[40] J. Zhang, L. Chen, C. Wang, L. Zhuo, Q. Tian, and X. Liang, "Road recognition from remote sensing imagery using incremental learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 11, pp. 2993–3005, Nov. 2017.
[41] S. Dang, Z. Cui, Z. Cao, and N. Liu, "SAR target recognition via incremental nonnegative matrix factorization," *Remote Sens.*, vol. 10, no. 3, p. 374, 2018.
[42] R. Roscher, B. Waske, and W. Forstner, "Incremental import vector machines for classifying hyperspectral data," *IEEE Trans. Geosci. Remote Sens.*, vol. 50, no. 9, pp. 3463–3473, Sep. 2012.
[43] H. Zheng, H. Zhang, X. Bai, and H. Zhao, "query expansion for VHR image detection," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, Jul. 2011, pp. 205–208.
[44] X. Tao, X. Hong, X. Chang, S. Dong, X. Wei, and Y. Gong, "Few-shot class-incremental learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 12183–12192.
[45] M. Ren, R. Liao, E. Fetaya, and R. Zemel, "Incremental few-shot learning with attention attractor networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 5275–5285.
[46] X. Yao, X. Feng, J. Han, G. Cheng, and L. Guo, "Automatic weakly supervised object detection from high spatial resolution remote sensing images via dynamic curriculum learning," *IEEE Trans. Geosci. Remote Sens.*, early access, May 18, 2020, doi: 10.1109/TGRS.2020.2991407.
[47] J. Han, D. Zhang, G. Cheng, L. Guo, and J. Ren, "Object detection in optical remote sensing images based on weakly supervised learning and high-level feature learning," *IEEE Trans. Geosci. Remote Sens.*, vol. 53, no. 6, pp. 3325–3337, Jun. 2015.

Jingzhou Chen received the B.E. degree in computer science and technology from Sichuan University, Chengdu, China, in 2016. He is pursuing the Ph.D. degree with the College of Computer Science, Zhejiang University, Hangzhou, China.

His research interests include machine learning, pattern recognition, and remote sensing image processing.



Shihao Wang received the B.E. degree in software engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2016, and the M.E. degree in computer science and technology from Zhejiang University, Hangzhou, China, in 2019.

His research interests include computer vision, pattern recognition, and machine learning.

Ling Chen received the B.S. and Ph.D. degrees in computer science from Zhejiang University, Hangzhou, China, in 1999 and 2004, respectively.

He is an Associate Professor with the College of Computer Science, Zhejiang University. His research interests include ubiquitous computing and data mining.

Haibin Cai received the B.Eng. and M.S. degrees from the National University of Defense Technology, Changsha, China, in 1997 and 2004, respectively, and the Ph.D. degree from the Donghua University, Shanghai, China, in 2008.

He is a Professor with the Software Engineering Institute, East China Normal University, Shanghai, China. His research interests include the Internet of Things, crowdsensing, trustworthy computing, and mobile Computing.

Yuntao Qian (Member, IEEE) received the B.E. and M.E. degrees in automatic control from Xi'an Jiaotong University, Xi'an, China, in 1989 and 1992, respectively, and the Ph.D. degree in signal processing from Xidian University, Xi'an, in 1996.

From 1996 to 1998, he was a Post-Doctoral Fellow with the Northwestern Polytechnical University, Xi'an. Since 1998, he has been with the College of Computer Science, Zhejiang University, Hangzhou, China, where he became a Professor in 2002. From 1999 to 2001, 2006, 2010, 2013, 2015 to 2016, and 2018, he was a Visiting Professor with Concordia University, Montreal, Canada, Hong Kong Baptist University, Hong Kong, Carnegie Mellon University, Pittsburgh, PA, USA, the Canberra Research Laboratory of NICTA, Canberra, ACT, Australia, Macau University, Macau, China, and Griffith University, Brisbane, QLD, Australia. His research interests include machine learning, signal and image processing, pattern recognition, and hyperspectral imaging.

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