electronics

MDPI

Article

A Fast and Accurate Few-Shot Detector for Objects with Fewer Pixels in Drone Image

Yuxuan Gao, Runmin Hou, Qiang Gao and Yuanlong Hou *

School of Mechanical Engineering, Nanjing University of Science and Technology, Nanjing 210094, China; gaoyuxuan1994@njust.edu.cn (Y.G.); riluo1102@hotmail.com (R.H.); gaoq0916@sina.com (Q.G.)
* Correspondence: houyuanlong1964@163.com

Abstract: Unmanned aerial vehicles (UAVs) are important in modern war, and object detection performance influences the development of related intelligent drone application. At present, the target categories of UAV detection tasks are diversified. However, the lack of training samples of novel categories will have a bad impact on the task. At the same time, many state-of-the-arts are not suitable for drone images due to the particularity of perspective and large number of small targets. In this paper, we design a fast few-shot detector for drone targets. It adopts the idea of anchor-free in fully convolutional one-stage object detection (FCOS), which leads to a more reasonable definition of positive and negative samples and faster speed, and introduces Siamese framework with more discriminative target model and attention mechanism to integrate similarity measures, which enables our model to match the objects of the same categories and distinguish the different class objects and background. We propose a matching score map to utilize the similarity information of attention feature map. Finally, through soft-NMS, the predicted detection bounding boxes for support category objects are generated. We construct a DAN dataset as a collection of DOTA and NWPU VHR-10. Compared with many state-of-the-arts on the DAN dataset, our model is proved to outperform them for few-shot detection tasks of drone images.

Keywords: object detection; drone; small object; few-shot; anchor-free; Siamese framework; attention mechanism

check for updates

Citation: Gao, Y.; Hou, R.; Gao, Q.; Hou, Y. A Fast and Accurate Few-Shot Detector for Objects with Fewer Pixels in Drone Image. *Electronics* **2021**, *10*, 783. https://doi.org/10.3390/electronics10070783

Academic Editor: Daniel Gutiérrez Reina

Received: 5 March 2021
Accepted: 22 March 2021
Published: 25 March 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Object detection has played a more and more important role in drone-based applications. Traditional object detection methods, such as the histogram of oriented gradients descriptor (HOG) [1] and the deformable part model (DPM) [2], perform well when detecting a certain specific object. However, they have poor performance when detecting multiple classes of objects. Since neural network framework was proposed, the development of object detection has progressed very fast. Some state-of-the-art object detectors (such as Faster R-CNN [3], YOLO [4], SSD [5], etc.) have been proposed with high accuracy on conventional detection datasets (PASCAL VOC 2007, PASCAL VOC 2012, COCO, etc.). Nevertheless, object detection for drone images suffers a lot due to the particularity of drone images. The unmanned aerial vehicle (UAV) images are different from images in generic datasets in following points.

Diversity of scale: Drone images are taken from different altitudes, and the sizes of targets are different, even for similar targets.

Different direction of perspective: Aerial images are generally viewed from high altitude, but most of the conventional data sets are from the ground level perspective, so the pattern of the same object is different.

Densely arranged small targets problem: Many targets in aerial images are small targets (dozens or even a few pixels), which results in a small amount of target information.

High background complexity: Drone image has a large field of vision. The field of vision may contain a variety of backgrounds, which will cause strong interference to object detection.

Therefore, for object detection task of drone-based intelligent application, a special dataset like NWPU VHR-10 [6] is necessary for training an ideal object detector. However, such a special dataset often suffers from the problem of a small number of labeled images. Especially when we need to detect a new category which has not been learned before, it is difficult to retrain a qualified detector through a few shots of new class targets. Therefore, it is a challenging and significant task to promote few-shot learning for UAV target detection. Few-shot learning is an important branch of object detection, as it is a method which quickly establishes the cognition of novel categories by only a few examples of targets. In recent years, some works for few-shot object detection [7–10] have been proposed with the drawback that they require fine-tuning to be able to detect new categories. It is difficult to directly apply the related progress to detection tasks of novel categories until [11] proposed a general-purpose few-shot object detector, through the well-designed Attention-RPN, Multi-Relation Detector, and contrastive training strategy, the network can squeeze out the matching relationship between targets by training on a high-diversity dataset FSOD, and can carry out reliable detection of novel categories without fine-tuning. It inspires us to train the model to learn a general matching relationship to distinguish objects of the same category from those of different categories, instead of learning the details of each category separately. This enables our model to have better generalization ability for novel categories. However, we argue that the model proposed in [11] ignores the background appearance information during inference just like the popular Siamese paradigm, and the impact of this drawback is magnified in the task of UAV image detection, which aims to detect small targets in complex background. Hence, we use the idea of [12] for reference which inspires us to introduce background information to iteratively optimize the target model. In addition, the network trained on the conventional dataset cannot be used as a backbone for UAV image detection, because the features they learned do not match drone-based object detection task [13], which contains a large number of small objects with fewer pixels instead of multi-pixel objects with clear appearance characteristics. To obtain optimal feature embeddings for a discriminative target model, we design a backbone to extract features for small objects (see Section 4.1). It contains fewer parameters, which makes it suitable for training with fewer samples. Moreover, it extracts the first feature map from the shallow layer of the network, which is conducive to preserving the pixel information of small objects.

In reality, the real-time performance of models greatly limits their applications. Hence, we attach great importance to the inference speed of our model. In most state-of-the-arts of few-shot detection or object tracking, two-stage model are generally used, for example, reference [11] uses Faster-RCNN network as its weight-shared framework, which uses an anchor as reference and contains RPN for region proposal. This complex two-stage model is obviously not as fast as a one-stage model. We decide to use the idea of one-stage model in [14] to design the query branch in our model. Reference [14] proposed a state-of-the-art anchor-free one-stage model FCOS which defines positive and negative samples by distinguishing whether each point on the feature falls into the ground truth, and it was proven that FCOS was faster than its anchor-based counterparts while obtaining comparable detection accuracy with other state-of-the-arts. At the same time, as verified in [15], FCOS outperforms an anchor-based state-of-the-art RetinaNet [16] mainly because the definition of positive and negative samples of FCOS is more reasonable than that of RetinaNet. The anchor-free method does not need to set anchor related parameters, and can receive a more balanced number of positive and negative samples. In most cases, detection accuracy of one-stage models is usually behind that of two-stage models, and one of the main reasons is due to the class imbalance problem [17]. The anchor-free one-stage framework solves this problem, as it can reduce the accuracy gap between one-stage models

and two-stage models, and greatly improve the inference speed of the model. Inspired by this, we adopt the mechanism of anchor-free to design the query branch of our model.

In addition, we find that although there is no large area occlusion between objects in drone images, when objects are densely and inclined arranged in an image, there will be a large overlap between the rectangular detection bounding boxes. To solve this problem, in the post-processing stage, we utilize Soft-NMS [18] to retain accurate predicted bounding boxes with high intersection over union (IoU) when dealing with densely arranged targets, which improves our model significantly in terms of recall.

We divide our experiments into four modules. The first module compares our proposed model and other state-of-the-art few-shot models trained from scratch on a DAN dataset. The second module is an ablation study running on DAN datasets to validate the improvements in our model. In the last module, we explore the influence of different factors on object detection of our model and other practical models, such as image resolution, object density, and orientation. The images with densely arranged objects are selected from an UCAS-AOD dataset [19], which is not learned by our model to validate the effectiveness. Through experiments, we find that our model is more accurate when dealing with drone image datasets compared with other state-of-the-art few-shot detectors. It is due to the backbone and feature extraction, designed especially for small objects, and the whole feature map of support image is utilized to introduce background information which leads to stronger discriminating ability. In addition, since our model leverages an anchor-free one-stage model as the query branch and there are fewer parameters in the backbone, the inference speed is faster than other state-of-the-art few-shot detectors.

Our new few-shot detector is expected to have the following advantages: (1) Our model can detect novel categories without retraining and fine-tuning. (2) As the query branch is designed based on an anchor-free one-stage model, it is faster than the existing few-shot detectors. (3) Our model can be used to detect densely arranged small objects in remote sensing images with great accuracy.

2. Related Works

Since the main contribution of this paper is an anchor-free one-stage model for few-shot drone image detection, in this section, we briefly introduce two aspects related to our work: General object detection and few-shot detection.

General object detection. Object detection is a key technology of computer vision. In early years, object detection was usually formulated as a sliding window classification problem using handcrafted features like HOG and DPM. With the development of deep learning network (DNN), many classical backbone networks have been proposed for image classification, such as AlexNet [20], VGG16 [21], GoogLeNet [22], Darnet19 [4], etc., which has made CNN-based methods more and more popular. Most object detection models can be divided into two categories: Two-stage model and one-stage model. Compared with one-stage model, two-stage model has one more step to generate proposals by region proposal network (RPN). RPN filters out many negative locations to solve the class imbalance problem and provides refined anchors for the next classification and regression, which makes two-stage models usually have higher detection accuracy, but slower inference speed than one-stage models. Gishick et al. [23] used selective search to generate region proposals in R-CNN. Afterwards, following the structure of R-CNN, Fast R-CNN [24] introduced a RoI pooling layer to extract the region proposals generated by selective search from a shared feature map. Considering the enormous time cost of selective search, in Faster R-CNN [3], RPN replaced selective search to generate refined anchors for detection. Reference [25] proposed an approach based on multi-scale balanced sampling (MB-RPN) to solve the problem of difficult matching of small objects and detecting multi-scale objects, and it had high accuracy on DOTA dataset. Considering the high efficiency, the one-stage approach tends to be the first choice for the development of intelligence detection applications. Redmon et al. [4] proposed an extremely fast one-stage model YOLO, which used a single feed-forward neural network to directly predict object classes and locations

at the same time. Afterwards, YOLOv2 [26] improved YOLO in several aspects, i.e., batch normalization, high resolution classifier, dimension clusters, etc. Another classical one-stage model SSD [5] made a breakthrough in multi-scale object detection by introducing a multi-scale feature map and detecting objects of different sizes on the feature layer of the corresponding scale. DSSD [17] introduced additional context into SSD by combining a deconvolutional high-level feature map with a high-resolution, low-level feature map to improve the accuracy. DSOD [27] used DenseNet as the backbone to enable the training objective to supervise the optimization of parameters of earlier layers, hence realizing a model that can be trained from scratch. RefineDet [28] introduced the anchor refinement module (ARM) on the basis of one-stage model to filter out negative anchors to reduce search space for the classifier and coarsely adjust the locations of anchors for the subsequent regressor. Reference [29] proposed a seven-layer convolutional lightweight real-time detector SSD7-FFAM for embedded devices, which applied a novel feature fusion and attention mechanism to alleviate the impact of decreasing the number of convolutional layers, and it performed well on NWPU VHR-10. Nevertheless, since the accuracy of these one-stage model trails that of two stage methods, improving the detection accuracy of one-stage model is still an enormous challenge in objection detection.

Few-shot detection. Few-shot detection refers to learning the target object just from a few training samples. References [30–32] attempted to achieve few-shot learning by obtaining a general prior, which is shared across different categories. References [33–35] proposed the use of distance measure for few-shot learning. An increasingly popular solution for few-shot learning is meta-learning, which refers to design a strategy to guide the supervised learning in each task, so that the model has the ability of learning to learn. In this field, a Siamese network is proposed in [36], which is composed of two weight shared networks, which are used to extract the features of support image and query image, respectively, and the model can judge whether there is an object of support category in the query image by comparing two feature maps. Vinyals et al. [33] proposed Matching Network to learn the task of finding the most similar class for the target among a small set of labeled images. Prototypical Network [34] and Relation Network [35] use distance measure to realize classification. Ravi and Larochelle [37] proposed an LSTM meta-learner, which was dedicated to learning a general agent to guide parameter optimization. Similar to [37] in optimization for fast adaptation, Model-Agnostic Meta-Learning (MAML) [38] performed well in detecting novel categories by optimizing a task-agnostic network. In recent years, some works for few-shot object detection [7–10] have been proposed. However, they learn category-specific feature embedding and require fine-tuning to be able to detect novel category. It is difficult to directly apply the related progress to detection tasks of novel categories until [11] proposed a general-purpose few-shot object detector, through the well-designed Attention-RPN, Multi-Relation Detector, and contrastive training strategy, the network can squeeze out the matching relationship between targets by training on a high-diversity dataset FSOD, and can carry out reliable detection of novel categories without fine-tuning. It inspires us to train the model to learn a general matching relationship to distinguish objects of the same category from those of different categories, instead of learning the details of each category separately.

3. Network Architecture and Detection Pipeline

The overall architecture of our network is showed in Figure 1. Specifically, our model consists of multiple branches, where one branch is for query set and others are for support set. In order to simplify the presentation diagram, we only draw one support branch that contains a novel category. We build a weight shared backbone to extract feature maps for both support set and query set. We find that the objects in the drone image are generally small and do not need a large receptive field to detect objects. Therefore, in order to extract better features for small objects, three feature maps are extracted from the shallow layers of the backbone. However, the operation of our model is different from that of most Siamese-based models, which do the same processing to the feature maps and then directly

compare them or calculate correlation value. In our model, after extracting three feature maps, the processing of the two branches is completely different.

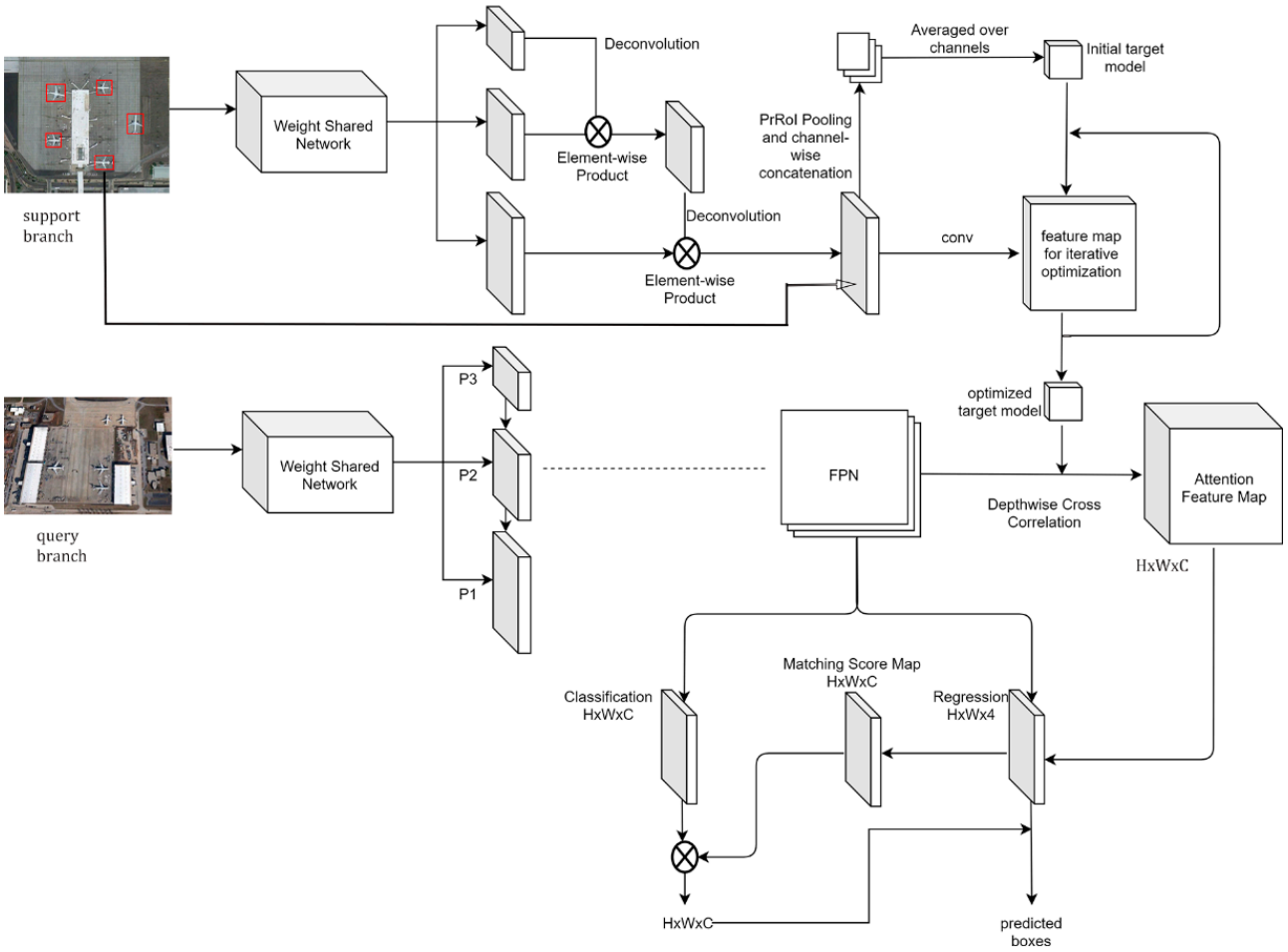


Figure 1. Network overall architecture. The feature maps of the support image are combined by deconvolution and element-wise product while that of support image are reserved to form FPN. The combined feature map of support is used to extract the initial target model and iteratively optimize the target model. The final target model is utilized to compute the similarity map with the feature maps from the query set as the attention feature map $H \times W \times C$, which helps our model match the categories in support from the regression results. For simplicity, only one support branch is shown. C represents the number of novel categories to be detected, that is, the number of support branches. After performing the depth-wise cross correlation between the final target model of each support branch and the feature map obtained from the query branch, an attention feature map of $H \times W \times 1$ is obtained. Additionally, after conducting channel-wise concatenation on these attention feature maps, a final attention feature map of $H \times W \times C$ is obtained. The attention feature map is then utilized to get the matching score map $H \times W \times C$, which indicates the probability of each pixel in the regression bounding boxes belonging to the category in support by calculating the average probability of the pixels belonging to each category within the regression boxes. The first result of classification $H \times W \times C$ is obtained by repeating a map of $H \times W \times 1$, which indicates the probability of the pixel belonging to a foreground on the channel C times. The final result of classification which indicates the probability of each pixel belonging to each category is obtained by conducting element-wise product between the first result of classification and the matching score map. Afterwards, this final result of classification map is utilized in turn to calculate the score of each regression bounding box on each category for post-processing.

For the support branch, our goal is to obtain an optimal target model to represent the support category regardless of its size. Reference [17] introduced a deconvolutional layer to adjust high-level feature map to the size of low-level feature map and achieved good accuracy by utilizing a deconvolutional layer and element-wise product method. Thus, we combine these feature maps through a step-by-step deconvolution and element-wise

product operation. As a drone image often contains many target objects, we take the precise RoI pooling (PrRoI Pooling) [39] features for these target objects from the combined feature map and concatenate them in channels. Next, by averaging over channels, we obtain the initial target model for the support category.

A problem is that the initial target model contains only the information of support category and ignores the use of background information, which makes the model unable to identify when the background is similar to the support category. Therefore, in order to utilize background information, we introduce the feature map obtained by processing the combined feature map with 1×1 convolution for iterative optimization. A correlation value will be obtained by calculating the depth-wise cross correlation between the initial target model and the feature map for iterative optimization. Furthermore, we introduce an annotation map according to the spatial distance between the pixel position and the center of the annotation box. Specifically, we use a method similar to multi-dimensional Gaussian distribution to determine our annotation map. Thus, we iteratively optimize the target model by reducing the gap between the cross-correlation map and the labelled value map.

For the query branch, we use the three feature maps P1, P2, P3, which are produced by top-down connecting the extracted layers from the backbone to form a feature pyramid network (FPN) [40], so that our model can regress different objects from three different scales, respectively. Two vectors $H \times W \times C$ (C = 1 for one class in support set) and $H \times W \times 4$ are obtained from these feature maps through two branches for classification and regression bounding boxes, respectively. C represents the number of novel categories to be detected, that is, the number of support branches. Since our model is based on learning a large number of different categories and then detecting new categories of objects, the features we extract can be understood as features extracted according to the general rules of objects. Therefore, the vector $H \times W \times C$ is obtained by repeating a map of $H \times W \times 1$ which indicates the probability of the pixel belonging to a foreground on the channel C times and the vector $H \times W \times 4$ represents the predicted bounding boxes (offsets of left, top, right, and bottom) of each pixel on the feature map. Then, we utilize the optimized target model obtained from support set and attention mechanism to help us determine whether the regression bounding boxes obtained belongs to the category in the support image. To be specific, we produce an attention feature map by computing the similarity between the final target model of support and the feature map of the query by depth-wise cross correlation. The attention feature map then is utilized to get the matching score map, which indicates the probability of each pixel in the regression bounding boxes belonging to the category in support by combining the information of regression bounding boxes, and then the probability that each pixel in the feature map belongs to the category in support is calculated by combining the probability that each pixel in the feature map belongs to a foreground in query image which finally returns to guide the filtering of regression bounding boxes in the post-process stage.

4. Model Details

4.1. Backbone Network

We design a Swish-DenseNet as the backbone to extract feature maps. The dense block consists of a 1×1 convolution layer and 3×3 convolution layer cycles. Among them, a 1×1 convolution layer is also called a bottleneck, which is used to reduce dimensions and combine features of different channels. DenseNet inserts a transition layer between each two adjacent dense blocks, which includes a 1×1 convolution layer to compress dimensions, and a size 2×2, stride 2 average pooling layer to change the resolution of the feature map, so that a feature map with a different scale can be obtained from each dense block. DenseNet is a compact and effective backbone network, which is preferable for our detector to learn from scratch.

In our experiments on DAN dataset, we utilize a modified DenseNet (growth rate k = 24) as the feature extractor. It consists of the initial convolution layers and four dense blocks with three transition layers between each two adjacent dense blocks. As proven by

the heatmaps in [41], the first few convolution layers of DNN contain more information of small objects, while the deep layers contain strong semantic features, but less information of small objects. We argue that the first feature layer should be located in the front of the network as much as possible. Therefore, in order to capture more information of small objects, we use only two convolution layers without a max pooling layer as the initial convolution layers and set the number of layers of dense blocks at the front of the network to be small. The specific configuration is shown in Table 1.

Table 1. Swish-DenseNet architecture (growth rate k = 24 in each dense block).

Layer	Parameters	Output (Input 640 × 640 × 3)
Convolution	3 × 3 conv, stride 2	320 × 320 × 64
Convolution	3 × 3 conv, stride 2	320 × 320 × 128
Dense block 1	$\begin{bmatrix} 1 \times 1\ conv \\ 3 \times 3\ conv \end{bmatrix} \times 4$	160 × 160 × 224
Transition layer	1 × 1 conv, stride 1 2 × 2 average pooling, stride 2	160 × 160 × 112 80 × 80 × 112
Dense block 2	$\begin{bmatrix} 1 \times 1\ conv \\ 3 \times 3\ conv \end{bmatrix} \times 6$	80 × 80 × 256
Transition layer	1 × 1 conv, stride 1 2 × 2 average pooling, stride 2	80 × 80 × 128 40 × 40 × 128
Dense block 3	$\begin{bmatrix} 1 \times 1\ conv \\ 3 \times 3\ conv \end{bmatrix} \times 12$	40 × 40 × 416
Transition layer	1 × 1 conv, stride 1 2 × 2 average pooling, stride 2	40 × 40 × 208 20 × 20 × 208
Dense block 4	$\begin{bmatrix} 1 \times 1\ conv \\ 3 \times 3\ conv \end{bmatrix} \times 12$	20 × 20 × 496

In addition, most neural networks use ReLUs as the activation functions. However, there is a hidden problem of the dying ReLUs [42]. Since its gradient in the negative range is 0, unreasonable parameter initialization and a large update of parameters in back propagation will make activation values of some neurons negative, meaning these neurons may never be activated, and the corresponding parameters cannot be updated. In this case, ReLUs collapse to a constant function and "die", effectively removing their contribution from the model, and that is what we call the dying ReLUs issue. In our work, we use a Swish unit [43] proposed by Google Brain as the activation function. The Swish activation function is defined as:

$$Swish(x) = x * Sigmoid(x) \quad (1)$$

where x denotes the input of the activation, and $Sigmoid(x)$ equals $1/(1 + e^{-x})$.

Reference [43] proved that Swish outperforms ReLU on deeper models by experiments. Swish is unbounded above and bounded below like ReLU, whereas it is smooth, non-monotonic, and unsaturated, which alleviates the dying neuron problem and gradient vanishing. Meanwhile, the simplicity of Swish and its similarity to ReLU make it easy to replace ReLUs with Swish units [43].

4.2. Feature Fusion and FPN

For the support branch, in order to obtain a feature map that can express the image in multiple scales, we utilize deconvolution and element-wise product operation to fuse high-resolution, low-semantic features and low-resolution, high-semantic features. To be specific, we use deconvolutional layers to adjust a high-level feature map to the size of a low-level feature map and then combine them by element-wise product. This feature map is used to extract RoI features from the image and to optimize the target model iteratively. This enables our final target model to have better representation of the support category at multiple scales.

For the query branch, we reserve the three feature maps P1, P2, P3, which are produced by top-down connecting the extracted layers from the backbone to form a feature pyramid network (FPN). Specifically, P1 is the shallowest feature, P2 is obtained by combining deconvolutional P1 and second extracted feature layer with element-wise product operation, and similar to P2, P3 is the combined feature map of P2 and the last extracted feature layer. This FPN enables our model to regress different objects from three different scales, respectively.

4.3. Initialization and Iterative Optimization of Target Model

After obtaining the combined feature map of the support image, we use PrRoI pooling to extract the feature maps of labeled boxes for objects of the same support category. These feature maps are then combined by channel-wise concatenation and we take the average value over channels as the initial target model. To be specific, we denote the feature map of the object i as $A^i \in t^{S\times S\times C}$ and the number of objects labeled in the support image as N, so the initial target model B can be formulated as

$$B_{h,w,c} = \frac{1}{N}\sum_{i=1}^{N} A^i_{h,w,c} \tag{2}$$

where h, w, c represent the abscissa, ordinate, and channel of the pixel position, respectively.

As the initial target model only contains the information of support category, ignoring the utilization of background information, it is not discriminative when the background is similar to the support category. In order to introduce background information to generate a more discriminative target model, we use the feature map of whole image for iteratively optimizing the target model. For example, if we denote the feature map of the whole image for iterative optimization as W, then a cross-correlation map M can be formulated as

$$M_{h,w,1} = \frac{1}{C}\sum_{i,j,c} B_{i,j,c}\cdot W_{h+i-1,w+j-1,c}, \quad i,j \in \{1,\ldots,S\} \tag{3}$$

where the target model B serves as a kernel to slide on the whole feature map W in a depth-wise cross correlation manner [44], and we take the average value over channels as the cross-correlation map, which indicates the similarity of the pixel region to the support category. Furthermore, we introduce an annotation map according to the spatial distance between the pixel position and the center of the annotation box. Specifically, we use a method similar to multi-dimensional Gaussian distribution to determine our annotation map, where we make the center of the label box take the highest value 1, the value at the border corresponds to the value at the standard deviation for each labeled bounding box in the support image, the pixel region of background takes 0, and the value of the pixel closer to the center is closer to 1. In detail, for a labeled bounding box $(\overline{x}, \overline{y}, a, b)$, where $(\overline{x}, \overline{y})$ is the center of the bounding box, a and b are the length and width of bounding, respectively, the value of a pixel region (x, y) within this bounding box can be calculated as

$$\begin{aligned} G_{x,y,1} &= \frac{1}{\sqrt{2\pi}\cdot\frac{a}{2}}\exp[-\frac{(x-\overline{x})^2}{2\cdot(\frac{a}{2})^2}]\cdot\frac{1}{\sqrt{2\pi}\cdot\frac{b}{2}}\exp[-\frac{(y-\overline{y})^2}{2\cdot(\frac{b}{2})^2}] + v \\ &= \frac{2}{ab\pi}\exp\left\{-2[\frac{(x-\overline{x})^2}{a^2} + \frac{(y-\overline{y})^2}{b^2}]\right\} + v \end{aligned} \tag{4}$$

where v is used to compensate the value at the center to 1. Thus, we iteratively optimize the target model by reducing the gap between the cross-correlation map M and the annotation map G.

4.4. Attention Feature Map

Attention feature map indicates the similarity information between each pixel region in the feature map of query image and the category of support. Here, we obtain the

attention feature map by calculating the depth-wise cross correlation between the final target model and the feature map of the query image. Similar to formula (3), if we denote the final target model as $X \in t^{S\times S\times C}$ and feature map of query as $Y \in t^{H\times W\times C}$, for one category in support set, the attention feature map Z can be represented as

$$Z_{h,w,1} = \frac{1}{C}\sum_{i,j,c} X_{i,j,c} \cdot Y_{h+i-1,w+j-1,c}, \quad i,j \in \{1,\ldots,S\} \tag{5}$$

where each channel represents a category to be detected, that is, an input support branch.

4.5. Matching Score Map

In our work, we propose a map to transform the classification of foreground pixels and background pixels from query branch into the classification of target and non-target pixels. We first extract the regression bounding box $t^* = (l^*, t^*, r^*, b^*)$ for each pixel region on the feature map of query, and then take the mean value of the similarity values of pixel regions within each bounding box on the attention feature map as the matching score of the pixel region in the query feature map for the support category. Here, l^*, t^*, r^*, and b^* are the distance from the location of each pixel of feature map to the four sides of the regression bounding box [14]. To be specific, for a pixel region (x, y) on the feature map with the stride = s of FPN, we denote a regression bounding box as $t^* = (l^*, t^*, r^*, b^*)$, then the coordinates of the left-top and right-bottom corners of the corresponding region of the regression bounding box on the attention feature map are (lt_x, lt_y) and (rb_x, rb_y), which can be formulated as:

$$\begin{aligned} lt_x &= \frac{xs+\frac{s}{2}-l^*}{s}, lt_y = \frac{ys+\frac{s}{2}-t^*}{s}, \\ rb_x &= \frac{xs+\frac{s}{2}+r^*}{s}, rb_y = \frac{ys+\frac{s}{2}+b^*}{s}. \end{aligned} \tag{6}$$

Thus, we define our matching score map as follows:

$$Match_{x,y,c} = \frac{1}{N_b}\sum_{\mathrm{i}=lt_x}^{rb_x}\sum_{j=lt_y}^{rb_y} Z_{i,j,c} \tag{7}$$

where N_b equals $(rb_x - lt_x) \times (rb_y - lt_y)$, and the different channels on matching score map indicate the matching scores of different support categories.

5. Training Strategy

5.1. Dataset and Image Processing

In order to learn from scratch and ensure diversity of training categories, we need a lot of data for training to make the parameters of our model work properly, but the actual data is not as much as required. In NWPU VHR-10, there are only 650 images with 10 classes of labeled objects. The images of NWPU VHR-10 are an order of magnitude less than that of generic detection datasets. To make up for this, we construct a DAN dataset as a collection of DOTA [45] and NWPU VHR-10, which consists of 15 representative categories, i.e., soccer ball field, helicopter, swimming pool, roundabout, large vehicle, small vehicle, bridge, harbor, ground track field, basketball court, tennis court, baseball diamond, storage tank, ship, and plane.

We find that a large number of small objects in the DAN dataset are not labeled, resulting in a small number of available samples, which will make detectors miss the detection of small targets. Secondly, the lack of diversity of their context background makes it difficult to detect small targets in other backgrounds. We use data augmentation that focuses on small objects. We copy and paste the small objects to the position that does not overlap with the existing objects in the images to increase the diversity of the location of the small target, and ensure that the objects appear in an appropriate context. Before pasting the target to the new location, we make a random transformation. We scale the target size

to 80~120% and rotate the angle by ±15 degrees. We only consider the objects that are not occluded, because the discontinuous samples with occluded areas will be distorted.

5.2. Loss Function

We leverage the two-way contrastive training strategy of [11] to enable our model to identify objects of the same category from objects of different categories. Specifically, for each query image q_c, we randomly choose one support image s_c with objects of the same category and one support image s_n with objects of other different categories to construct a training triplet (q_c, s_c, s_n). In the query image of the triplet, only the objects of category c are labeled as positive, while other objects and background are labeled as negative. For this triplet, our model should not only match the same category objects between (q_c, s_c), but also distinguish objects with different classes between (q_c, s_n). Therefore, we design the training loss function as follows:

$$L(q_c, s_c, s_n) = L_{match}(q_c, s_c) + \alpha L_{match}(q_c, s_n) + \lambda L_{reg}(q_c, s_c) \tag{8}$$

where $L_{matching}(q_c, s_c)$ is focal loss for offsetting the impact of class imbalance and makes the model pay more attention to hard examples by adjusting the weights, $L_{matching}(q_c, s_n)$ is the binary cross-entropy loss, and $L_{reg}(q_c, s_c)$ is the IOU loss as in [46]. In addition, we add the weighting factors α and λ, where the former is used to down-weight the matching loss of (q_c, s_n), and the latter is used to adjust the weight of L_{reg}. In our work, α is set to 0.5, and λ is set to 1 by cross validation.

5.3. Post-Processing

Since most objects overlap a lot with other adjacent objects in images with densely arranged objects, many correct detection results are filtered out by conventional NMS due to large IoUs. When an IoU above the threshold appears, conventional NMS sets the confidence score of the bounding box with lower confidence to zero to remove the redundant bounding boxes. Therefore, we use Soft-NMS rather than conventional NMS to process the detection results. Soft-NMS retains the correct results by reducing the lower confidence score instead of zeroing it when there is an IOU above the threshold. Specifically, for the bounding box b_i, if the IoU between b_i and another bounding box b_j, which has higher confidence score than b_i, is greater than the defined threshold T, the confidence score c_i will be recalculated according to the following equation:

$$c_i = \begin{cases} c_i, & IoU(b_i, b_j) < T \\ c_i(1 - IoU(b_i, b_j)), & IoU(b_i, b_j) \geq T \end{cases} \tag{9}$$

where T is set to 0.5 as in most other works.

6. Implementation Details and Evaluation Metrics

We implement our model based on a Tensorflow framework. Our detector is trained from scratch with a computer running Ubuntu 18.04 LTS. Stochastic gradient descent (SGD) is performed on Nvidia GeForce GTX 1060 with 8 GB GPU memory. The experiments utilize CUDA v10.0, cuDNN v7.5.0, and Tensorflow-gpu-1.13 to accelerate computation. Considering that too many training iterations may damage performance by making the model over-fit, we take 80 epochs to train this model. We use the momentum method to optimize SGD. We train the model with an initial learning rate of 0.0002, momentum of 0.9, and weight decay of 0.0005. In addition, we use sub-batch method to solve the memory overflow problem caused by large batch size. Our model takes batch size 32, image size 640×640 pixels as input.

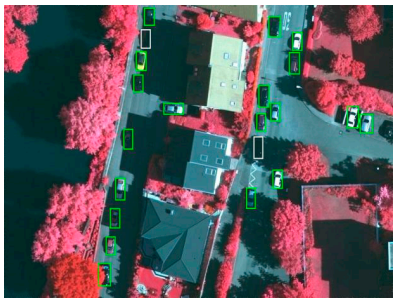
We divide our experiments into three modules to validate the effectiveness of our model and explore the factors that may affect the detection performance. We choose vehicle, storage tank, and plane as the novel class to conduct experiments. Since our purpose is to create a model which can learn from scratch on an unconventional dataset, not using a pre-

trained model is the premise of all our experiments. The first module includes performing our proposed model and other state-of-the-art few-shot detectors on a DAN dataset and then comparing their results. Their object detection performances are measured by two evaluation metrics on DAN base classes and DAN novel classes, respectively, which are mean Average Precision (mAP) and speed (FPS). The second module includes investigating the validity of different components in our model. Several controlled experiments on DAN datasets are conducted for the ablation study. In the last module, we test our model on the images with a single influence factor, image resolution and object density, to find out how these factors affect our model, respectively. To be specific, we first down-sample the images to a series of lower resolutions (0.8×, 0.6× and 0.4×) and then train our model with each resolution and measure their predictions on the mAP at IoU 0.5 metric. On the other hand, we collect images with densely arranged small objects in UCAS-AOD to form a test set in order to evaluate the performance of our model on detection of densely arranged small objects.

7. Experimental Result and Analysis

7.1. Performance Comparison with Other Few-Shot Detectors on DAN Dataset

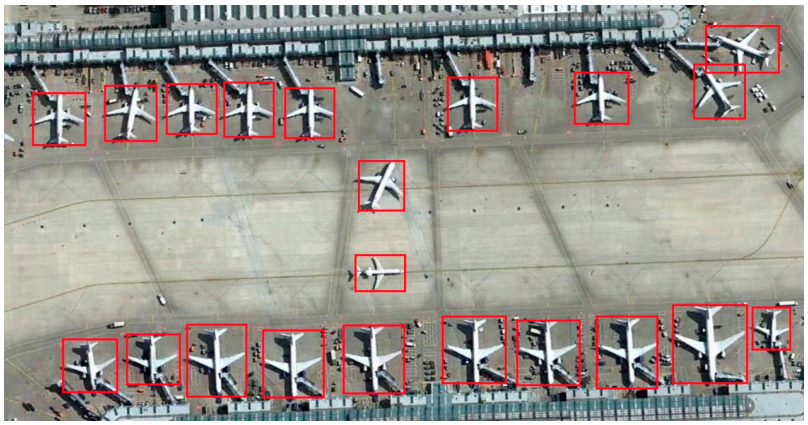
The result of the performance comparison between our model and other state-of-the-art few-shot detectors is shown in Table 2. All models are trained and tested on a DAN dataset with the same pre-process and post-process setting. Since our model is designed for unconventional dataset with a large number of small targets and applies one-stage model as template in query branch, through experiments, it appears that our proposed model has better performance not only on detection accuracy, but also on inference speed compared to other few-shot detectors. There are some examples of detection results predicted by our proposed model (see Figure 2).



(**a**) Vehicle



(**b**) Storage tank



(**c**) Plane

Figure 2. Some examples of detection results predicted by our proposed model. Image (**a**–**c**) are the detection results of vehicle, storage tank, and plane. The white boxes in the picture (**a**) indicates the targets of missed detection.

Table 2. Performance comparison between our proposed model and other state-of-the-art few-shot methods. Input image size is 640 × 640 and predictions are measured at 0.5 IoU metric.

Model	Novel Classes (mAP)	Base Classes (mAP)	Speed (FPS)
RepMet	19.3	53.7	20.4
YOLO-Few-shot	21.2	55.2	25.8
Meta R-CNN	30.6	57.3	14.9
Proposed model	39.9	67.4	27.2

7.2. Ablation

The result of the ablation study is shown in Table 3. Specifically, we use a consistent setting in evaluation for fair comparison. All models are trained and tested on a DAN dataset. Our complete proposed model achieves mAP 67.4 and 39.9 when tested on DAN base classes and DAN novel classes, respectively. Then, we remove some components of the model to observe the detection performance in order to analyze the effect of the components, and "√" indicates which components are included. Obviously, our designed backbone for extracting feature for small objects, Swish activation function, and feature map for iterative optimization improve detection performance. Our designed backbone for extracting features for small objects improves the mAP of our model on DAN base classes and DAN novel classes by 5.7% and 9.1%, respectively, because our designed backbone makes use of skip connection to simplify the learning objective and enable the model to apply a deeper network structure, thus improving the effectiveness of feature extraction. It is preferable for tasks where the model needs to be trained from scratch due to fewer parameters. Furthermore, since feature maps are extracted from shallow layers, the information of small objects are relatively complete, which helps to express small objects in DAN dataset. Swish activation function improves the mAP of our model on DAN base classes and DAN novel classes by 0.2% and 0.1%, respectively, which proves that it is better than ReLU in our work. It alleviates the dying neuron problem and gradient vanishing. The feature map for iterative optimization also brings improvement on detection performance. It increases the mAP of our model on DAN base classes and DAN novel classes by 3.3% and 4.1%, respectively, because it introduces information of background when generating a final target model, which makes the model more discriminative in distinguishing foreground and background.

Table 3. Effectiveness of different components on DAN base classes and DAN novel classes. Input image size is 640 × 640 and predictions are measured on the mAP at 0.5 IoU metric. Experiments start with a basic SSD model. Our complete proposed model achieves mAP 67.4 and 39.9 when tested on DAN base classes and DAN novel classes, respectively, and "√" indicates which components are included.

		Proposed Model		
Our designed backbone for extracting features for small objects		√	√	√
Swish activation function			√	√
Feature map for iterative optimization				√
DAN base class mAP	58.2	63.9	64.1	67.4
DAN novel class mAP	26.6	35.7	35.8	39.9

7.3. Impact of Image Resolution and Object Density

In this module, we test the impact of image resolution and object density on object detection, and compare their impact on our model and well-trained models.

Image resolution impact: In this experiment, we down-sample the images to a series of lower resolutions (see Figure 3). Afterwards, we train our model and Meta R-CNN with each resolution and measure their predictions on the mAP at IoU 0.5 metric. Results are illustrated in Table 4.



(**a**) Scale 0.4 (**b**) Scale 0.6



(**c**) Scale 0.8 (**d**) Original

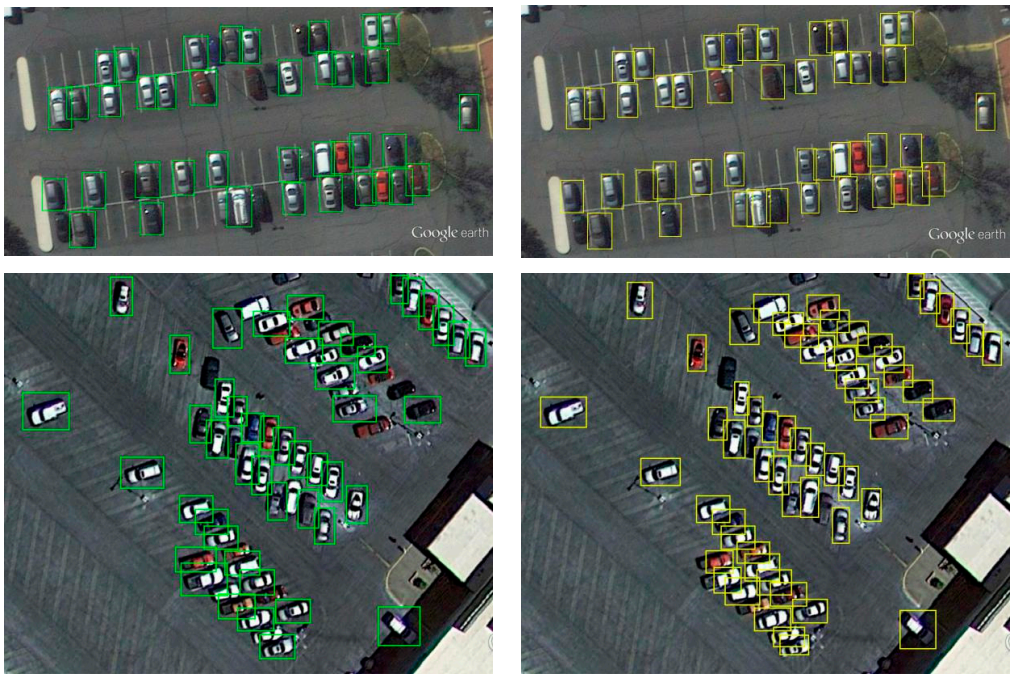
Figure 3. Images of different resolutions and the change trend of AP of different objects and mAP with the increase of resolution. Images (**a**–**c**) are a series of down-sampled images. Image (**d**) is the original picture.

Table 4. Impact of resolution on performance of Meta R-CNN and our model when detecting novel classes.

Image Resolution	Meta R-CNN				Proposed Model			
	AP			mAP	AP			mAP
	Vehicle	Storage Tank	Plane		Vehicle	Storage Tank	Plane	
Original	23.5	31.3	37.0	30.6	36.1	33.9	49.7	39.9
0.8x	22.9	28.3	35.2	28.8	34.4	32.5	43.2	36.7
0.6x	18.0	26.9	30.7	25.2	33.7	32.2	42.4	36.1
0.4x	9.4	23.6	17.1	16.7	26.1	28.4	32.3	28.9

Obviously, image resolution is very important to the accuracy of target detection. From Figure 3, we can see that an image at low resolution looks more blurred and lacks a lot of detailed features, which makes small objects more difficult to identify. As shown in Table 4, the detection accuracy of the small objects (vehicle, storage tank, and plane) in the picture decreases sharply with the decrease of resolution. At the same time, when the resolution of the image is reduced to 0.4×, the detection accuracy of Meta R-CNN for vehicle, storage tank, and plane are reduced by 14.1%, 7.7%, and 19.9%, while that of our model is reduced by 10%, 5.5%, and 17.4%. We can find that our model is more robust to the change of image resolution in the detection of small objects than Meta R-CNN.

Object density impact: In order to eliminate the influence of missing detection of the detector itself, we compare the model trained by adding vehicle class into base classes with a practical method, YOLOv3, which directly learns vehicle class. We select some images with densely arranged vehicles from an UCAS-AOD dataset, and use YOLOv3 and our model trained on a DAN dataset to detect the targets. Figure 4 shows some detection results of YOLOv3 and our model, respectively.



(**a**) YOLOv3 (**b**) Proposed Model

Figure 4. Detection results of YOLOv3 and our model in densely arranged objects detection. The pictures in the first column are the result of YOLOv3, which shows the detection result in green boxes; while the pictures in the second column are the results of our model, which uses yellow boxes to display the detection results. In the first row of pictures, the cars are arranged vertically, and the adjacent bounding boxes overlap a small area; in the second row of pictures, there are many cars with oblique orientation, and the adjacent bounding boxes overlap a large area.

Obviously, a lower rate of missed detection indicates that our model has better performance in the detection of densely arranged objects. This is especially true when detecting objects with different sizes and oblique orientations. As in this case, the predicted bounding boxes of two adjacent objects have a high IoU, which leads to the traditional NMS filtering out the prediction with lower confidence score, resulting in missed detection. Soft-NMS, utilized in our model, preserves the correct prediction by reducing the lower confidence score rather than setting it to zero when there is an IOU above the threshold.

8. Conclusions

The research on few-shot detection is of great significance, because it can not only reduce the cost of manual annotation, but also help to realize the diversification of detection targets. Although many detectors have been proposed, there are few real-time few-shot detection methods for UAV image targets. In this paper, we propose a novel few-shot object detector especially for special datasets with fewer labeled images and small objects. We design a special Swish-DenseNet as our backbone for feature extraction, which enable our model to be trained from scratch and have more effective feature maps. We introduce the feature map for iterative optimization to make use of background information to generate a more discriminative target model. Unlike most of the state-of-the-arts of few-shot detection, we utilize a one-stage model FCOS as template in the query branch rather than a two-stage model, thus making our model have higher inference speed. In addition, we leverage a matching score map to transform the classification of foreground and background from the query branch into the classification of target and non-target pixels, which integrates the information from the support branch and query branch. In addition, we use Soft-NMS to alleviate the missed detection problem when dealing with densely arranged targets. The experimental result on the DAN dataset shows that our proposed model has

better performance than other state-of-the-art few-shot models and maintains the high efficiency of one-stage model, which enable it to be applied on applications with real-time requirements. In the future, we will try to apply it to the target detection of UAV aerial photography through the cloud server.

Author Contributions: Formal analysis, Y.G.; Investigation, Y.G.; Methodology, Y.G. and Y.H.; Supervision, R.H., Q.G. and Y.H.; Writing—original draft, Y.G.; Writing—review and editing, Y.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (51805264).

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Dalal, N.; Triggs, B. Histograms of Oriented Gradients for Human Detection. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 20–25 June 2005.
2. Felzenszwalb, P.F.; Girshick, R.B.; McAllester, D.; Ramanan, D. Object Detection with Discriminatively Trained Part-Based Models. *IEEE Trans. Pattern Anal. Mach. Intell.* **2010**, *32*, 1627–1645. [CrossRef] [PubMed]
3. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *39*, 1137–1149. [CrossRef] [PubMed]
4. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-time Object Detection. *arXiv* **2016**, arXiv:1506.02640v5.
5. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.; Berg, A.C. SSD: Single Shot MultiBox Detector. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016.
6. Cheng, G.; Han, J.; Zhou, P.; Guo, L. Multi-class geospatial object detection and geographic image classification based on collection of part detectors. *ISPRS J. Photogramm. Remote Sens.* **2014**, *98*, 119–132. [CrossRef]
7. Chen, H.; Wang, Y.; Wang, G.; Qiao, Y. Lstd: A low-shot transfer detector for object detection. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 7–12 February 2018.
8. Kang, B.; Liu, Z.; Wang, X.; Yu, F.; Feng, J.; Darrell, T. Few-shot object detection via feature reweighting. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019.
9. Karlinsky, L.; Shtok, J.; Harary, S.; Schwartz, E.; Aides, A.; Feris, R.; Giryes, R.; Bronstein, A.M. Repmet: Representative-based metric learning for classification and few-shot object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019.
10. Yan, X.; Chen, Z.; Xu, A.; Wang, X.; Liang, X.; Lin, L. Meta r-cnn: Towards general solver for instance-level low-shot learning. In Proceedings of the International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019.
11. Fan, Q.; Zhuo, W.; Tang, C.K.; Tai, Y.W. Few-Shot Object Detection with Attention-RPN and Multi-Relation Detector. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 14–19 June 2020.
12. Bhat, G.; Danelljan, M.; Van Gool, L.; Timofte, R. Learning Discriminative Model Prediction for Tracking. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27 October–2 November 2019.
13. Zheng, L.; Tang, M.; Chen, Y.; Wang, J.; Lu, H. Learning Feature Embeddings for Discriminant Model Based Tracking. In Proceedings of the 2020 European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020.
14. Tian, Z.; Shen, C.; Chen, H.; He, T. FCOS: Fully convolutional one-stage object detection. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27 October–2 November 2019.
15. Zhang, S.; Chi, C.; Yao, Y.; Lei, Z.; Li, S.Z. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 14–19 June 2020.
16. Lin, T.; Goyal, P.; Girshick, R.; He, K.; Dollar, P. Focal Loss for Dense Object Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**. [CrossRef]
17. Fu, C.; Liu, W.; Ranga, A.; Tyagi, A.; Berg, A.C. DSSD: Deconvolutional Single Shot Detector. *arXiv* **2017**, arXiv:1701.06659v1.
18. Bodla, N.; Singh, B.; Chellappa, R.; Davis, L.S. Improving Object Detection with One Line of Code. *arXiv* **2017**, arXiv:1704.04503v2.
19. Zhu, H.; Chen, X.; Dai, W.; Fu, K.; Ye, Q.; Jiao, J. Orientation robust object detection in aerial images using deep convolutional neural network. In Proceedings of the IEEE International Conference on Image Processing IEEE, Quebec City, QC, Canada, 27–30 September 2015.
20. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. In *NIPS*; Curran Associates Inc.: Morehouse Lane, NY, USA, 2012.
21. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2015**, arXiv:1409.1556.

22. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. *arXiv* **2014**, arXiv:1409.4842v1.
23. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. *arXiv* **2014**, arXiv:1701.06659v1.
24. Girshick, R. Fast R-CNN. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 11–18 December 2015.
25. Yu, H.; Gong, J.; Chen, D. Object Detection Using Multi-Scale Balanced Sampling. *Appl. Sci.* **2020**, *10*, 6053. [CrossRef]
26. Redmon, J.; Farhadi, A. YOLO9000: Better, Faster, Stronger. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 6517–6525.
27. Shen, Z.; Liu, Z.; Li, J.; Jiang, Y.; Chen, Y.; Xue, X. DSOD: Learning Deeply Supervised Object Detectors from Scratch. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 1937–1945.
28. Zhang, S.; Wen, L.; Bian, X.; Lei, Z.; Li, S.Z. Single-Shot Refinement Neural Network for Object Detection. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4203–4212.
29. Li, Q.; Lin, Y.; He, W. SSD7-FFAM: A Real-Time Object Detection Network Friendly to Embedded Devices from Scratch. *Appl. Sci.* **2021**, *11*, 1096. [CrossRef]
30. Fei-Fei, L.; Fergus, R.; Perona, P. One-shot learning of object categories. *IEEE Trans. Pattern Anal. Mach. Intell.* **2006**, *28*, 594–611. [CrossRef] [PubMed]
31. Lake, B.M.; Salakhutdinov, R.; Tenenbaum, J.B. One-shot learning by inverting a compositional causal process. In *Advances in Neural Information Processing Systems*; Massachusetts Institute of Technology Press: Cambridge, MA, USA, 2015.
32. Lake, B.M.; Salakhutdinov, R.; Tenenbaum, J.B. Human-level concept learning through probabilistic program induction. *Science* **2015**, *350*, 1332–1338. [CrossRef] [PubMed]
33. Vinyals, O.; Blundell, C.; Lillicrap, T.; Kavukcuoglu, K.; Wierstra, D. Matching networks for one shot learning. In Proceedings of the Advances in Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016; pp. 3630–3638.
34. Snell, J.; Swersky, K.; Zemel, R.S. Prototypical networks for few-shot learning. In Proceedings of the Neural Information Processing Systems (NIPS), Long Beach, CA, USA, 4–9 December 2017.
35. Sung, F.; Yang, Y.; Zhang, L.; Xiang, T.; Torr, P.H.; Hospedales, T.M. Learning to compare: Relation network for few-shot learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2018, Salt Lake City, UT, USA, 18–23 June 2018.
36. Koch, G.; Zemel, R.; Salakhutdinov, R. Siamese neural networks for one-shot image recognition. In Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 7–9 July 2015.
37. Ravi, S.; Larochelle, H. Optimization as a model for few-shot learning. In Proceedings of the International Conference on Computer Vision, Toulon, France, 24–26 April 2017; Volume 2.
38. Finn, C.; Abbeel, P.; Levine, S. Model agnostic meta-learning for fast adaptation of deep networks. In Proceedings of the International Conference on Machine Learning, Sydney, Australia, 6–11 August 2017.
39. Jiang, B.; Luo, R.; Mao, J.; Xiao, T.; Jiang, Y. Acquisition of Localization Confidence for Accurate Object Detection. *arXiv* **2018**, arXiv:1807.11590v1.
40. Lin, T.; Dollar, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature Pyramid Networks for Object Detection. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 936–944.
41. Xie, W.; Qin, H.; Li, Y.; Wang, Z.; Lei, J. A Novel Effectively Optimized One-Stage Network for Object Detection in Remote Sensing Imagery. *Remote Sens.* **2019**, *11*, 1376. [CrossRef]
42. Arnekvist, I.; Carvalho, J.F.; Kragic, D.; Stork, J.A. The effect of Target Normalization and Momentum on Dying ReLU. *arXiv* **2020**, arXiv:2005.06195v1.
43. Ramachandran, P.; Zoph, B.; Le, Q.V. Swish: A Self-Gated Activation Function. *arXiv* **2017**, arXiv:1710.05941v2.
44. Li, B.; Wu, W.; Wang, Q.; Zhang, F.; Xing, J.; Yan, J. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019.
45. Xia, G.; Bai, X.; Ding, J.; Zhu, Z.; Belongie, S.; Luo, J.; Datcu, M.; Pelillo, M.; Zhang, L. A Large-Scale Dataset for Object Detection in Aerial Images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 19–21 June 2018; pp. 3974–3983.
46. Yu, J.; Jiang, Y.; Wang, Z.; Cao, Z.; Huang, T. Unitbox: An advanced object detection network. In Proceedings of the 24th ACM International Conference on Multimedia, Amsterdam, The Netherlands, 15–19 October 2016; pp. 516–520.