

Article

Deep Learning Based Electric Pylon Detection in Remote Sensing Images

Sijia Qiao ¹, Yu Sun ¹ and Haopeng Zhang ^{1,2,3,*} 

¹ Department of Aerospace Information Engineering, School of Astronautics, Beihang University, Beijing 102206, China; Qsjxyz@buaa.edu.cn (S.Q.); 17374136@buaa.edu.cn (Y.S.)

² Beijing Key Laboratory of Digital Media, Beijing 102206, China

³ Key Laboratory of Spacecraft Design Optimization and Dynamic Simulation Technologies, Ministry of Education, Beijing 102206, China

* Correspondence: zhanghaopeng@buaa.edu.cn; Tel.: +86-10-6171-6978

Received: 5 May 2020; Accepted: 5 June 2020; Published: 8 June 2020



Abstract: The working condition of power network can significantly influence urban development. Among all the power facilities, electric pylon has an important effect on the normal operation of electricity supply. Therefore, the work status of electric pylons requires continuous and real-time monitoring. Considering the low efficiency of manual detection, we propose to utilize deep learning methods for electric pylon detection in high-resolution remote sensing images in this paper. To verify the effectiveness of electric pylon detection methods based on deep learning, we tested and compared the comprehensive performance of 10 state-of-the-art deep-learning-based detectors with different characteristics. Extensive experiments were carried out on a self-made dataset containing 1500 images. Moreover, 50 relatively complicated images were selected from the dataset to test and evaluate the adaptability to actual complex situations and resolution variations. Experimental results show the feasibility of applying deep learning methods to electric pylon detection. The comparative analysis can provide reference for the selection of specific deep learning model in actual electric pylon detection task.

Keywords: electric pylon detection; deep learning; remote sensing image

1. Introduction

Electricity is one of the most crucial energy supports for economic development and technology progress. Furthermore, the stability of electricity supply is an essential requirement for regional development. In the entire power system, electric network is an important link to transfer the electric energy from the power plants with concentrated distribution to individual power users with scattered distribution [1]. In other words, this component of the power system most closely connects with the urban power supply. To monitor the performance of the electric network, electric pylons, which play the role of undertaking and guiding wires, need to be monitored frequently to ensure normal operation.

However, with the popularization of electricity and the increasing complexity of electric network, user residence expresses the trend of enlargement and diversification. Considering that current distribution of electric pylons contains the characteristics of large quantity, wide span, diverse appearance and complex surrounding terrain, traditional field inspections relying on manpower require large resource consumption but receive low time efficiency. Field inspections based on unmanned aerial vehicles (UAVs) may show better performance [2,3]. However, this approach proves to be difficult to realize real-time monitoring requirements in the face of large area, and is susceptible to the influence of the surrounding tall buildings. In contrast, satellite remote sensing monitoring has a large monitoring area and proves to be efficient and less influenced by surroundings,

which has been applied to global environmental observation [4]. Therefore, this paper focuses on electric pylon detection in high-resolution remote sensing images captured by satellites.

Furthermore, artificial interpretation in high-resolution remote sensing images turns out to be a significant amount of hard work. However, in addition to the influence of characteristics of the electric pylon itself, there exists inevitable shortcoming in the artificial interpretation compared with the machine recognition, i.e., the visual fatigue. Due to the influence of visual fatigue, continuous artificial interpretation work can significantly reduce the efficiency and accuracy of manual monitoring. Thus, this paper introduces deep learning methods to automatically interpret remote sensing images containing electric pylons, which can significantly improve the comprehensive efficiency of electric pylon detection.

In recent years, target detection methods based on deep learning have become a hot spot in related areas. Such kind of methods has been demonstrated to be effective in the detection of aircraft [5,6], ship [7,8], and condensing tower [9,10], on the basis of a large number of successful experiments. Furthermore, deep-learning-based detectors can adapt to several remote sensing data sources, and has been applied to the target detection tasks of optical [11], infrared [12], LiDAR [13], SAR [14], aerial images [15], etc. With the continuous improvement of deep learning theory and the iterative update of detection algorithm, target detectors based on deep learning have shown superiority over traditional object detection methods.

In this paper, to analyze the feasibility of deep learning methods in the detection of electric pylons in high-resolution remote sensing images, we select 10 state-of-the-art deep-learning-based target detectors to compare the comprehensive performance of these models in the electric pylon detection. To improve training efficiency, the detectors designed and pre-trained on natural images are fine-tuned on the basis of remote sensing images containing electric pylons. Extensive experiments were performed on a self-made remote sensing image dataset.

The rest of this paper is organized as follows. Section 2 shows the related works of our study. Section 3 describes the main content of our work, including the production of the dataset, the selection of deep learning models, and the adjustment of specific parameters in the experiments. Section 4 introduces the experimental process and the test results of each detector. Section 5 presents the comprehensive analysis of the results. Finally, Section 6 makes a conclusion.

2. Related Work

2.1. Object Detection Based on Deep Learning

In recent years, numerous deep-learning-based methods have been proposed to solve object detection problems. These detectors follow similar lines of thought, extract features using Convolutional Neural Network (CNN), and classify the objects and regress the bounding boxes using diverse methods. Deep-learning-based object detection methods are popularized by both two-stage and one-stage detectors.

Girshick et al. proposed R-CNN (Regions with Convolutional Neural Network) [16] as the first two-stage detector. The R-CNN method mainly acts as a classifier, training CNNs end-to-end to classify the proposal regions into object categories or background. SPP-Net (Spatial Pyramid Pooling Network) [17] and Fast R-CNN [18] promote the development of two-stage detectors further. Ren et al. introduced Faster R-CNN [19]. This meaningful detector proposed region proposal network to advance the efficiency of detectors and allow the detector to be trained end-to-end. Since then, scholars have introduced many methods to enhance Faster R-CNN from different points, e.g., Cascade R-CNN [20], Mask R-CNN [21], Grid R-CNN [22], HTC (Hybrid Task Cascade) [23], etc. These detectors generally have relatively superior accuracy, while cost longer running time and need larger memory.

On the other hand, one-stage detectors are popularized by SSD (Single Shot multibox Detector) [24] and YOLO (You Only Look Once) [25–28]. These detectors densely sample from different locations of the image uniformly, extracted features, and then classify and regress bounding-box directly.

They usually have less running time and memory cost, but have lower accuracy until the introduction of focal loss in Retinanet [29]. Focal loss is proved to greatly improve the comprehensive performance of one-stage detectors.

A brief illustration of deep-learning-based detectors is shown in Figure 1.

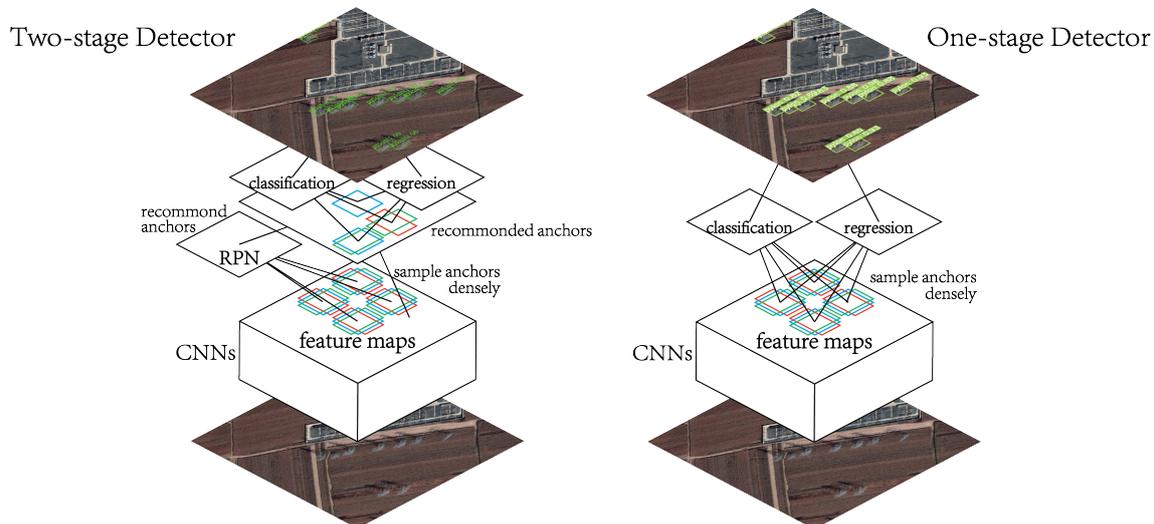


Figure 1. Basic structure of deep-learning-based detection networks from image input to result output: (Left) two-stage detectors; and (Right) one-stage detectors. These detectors firstly extract features using Convolutional Neural Network (CNN), and then densely sampled from different location to get anchors. Difference between two-stage and one-stage is mainly reflected on whether utilizes Region Proposal Network (RPN) [19]. Two-stage detectors classify the anchors recommended by RPN and regress the bounding-box to draw near the ground truth box, while one-stage detectors do classification and regression on all anchors.

For the above methods, the selection of candidate bounding boxes is all based on the prior anchors, and these methods are regarded as the anchor-based methods, which regress pre-set anchors and four variables, $[x,y,w,h]$. In recent years, researchers have paid more attention to anchor-free methods, which use several regression methods, e.g., regressing pixel position, and proposed a number of successful anchor-free detectors such as Corner-Net [30], Center-Net [31], FCOS (Fully Convolutional One-Stage) [32], etc. In certain scenes, anchor-free detectors have the identical accuracy and the superior speed compared to the anchor-based one-stage detectors. Anchor-free methods are growing into a focus of future research.

2.2. Electric Pylon Detection

How to effectively and accurately realize the automatic detection of electric pylons remains an urgent problem to be solved in remote sensing. Matikainen et al. [33] proposed to introduce remote sensing methods to detect power lines. They studied the application of several remote sensing data sources in power lines monitoring, including SAR images, optical satellite images, optical aerial images, thermal images, etc. Various remote sensing images provide a series of new ideas for electric pylon detection, some of which have been greatly improved in recent years, e.g., UAV monitoring [2,3] and detection based on Lidar data [34]. Besides, the detection and tracking of electric pylons in videos is also an important research field of electric pylon monitoring. Tilawat et al. [35] proposed an automatic detection method to locate electric pylons in aerial videos.

Utilizing learning methods is the development trend in the area of target detection and identification. Sampedro et al. [36] proposed a traditional supervised learning method for detecting and classifying transmission towers. In [36], two MLP (multi-layer perceptron) neural networks were trained using HOG (directional gradient histogram) features. The former MLP network was used to

achieve the mission of detection, and the latter network was used to classify different types of towers. Good evaluation results achieved by this model preliminary prove the feasibility of applying learning method to monitor power network working condition.

With the development of deep learning, it has been applied to detect electric pylon in SAR images. For example, Fei and Tan [37] proposed to use deep learning to identify electric tower in high-resolution SAR images. The authors aimed at balancing the precision and efficiency of identification, and, in particular, constructed a two-stage detector by cascading YOLOv2 [26] and VGG [38]. Comparing with YOLOv2, this detector achieved better detection performance with the recall reaching 73.8% in the testing process.

However, there are few works focusing on electric pylon in optical satellite remote sensing images. Therefore, to accelerate the research in this application area, we performed a series of extensive experiments based on high resolution optical remote sensing images and analyzed the advantages and disadvantages of nine state-of-the-art deep-learning-based detectors.

3. Electric Pylon Detection Based on Deep Learning

3.1. EPD Dataset

To study electric pylon detection based on deep learning, we specially collected a high-resolution remote sensing image dataset for electric pylon detection (EPD). Images in our EPD dataset were collected from Google Earth and image productions of Pleiades satellite. Specially, all images in the dataset are processed multi-spectral remote sensing image products, which are widely used in practical detection tasks. Pleiades images are orthoimages, and images from Google Earth are multi-spectral products captured by different sensors. Such multi-source data can better test the generalization ability of deep learning detectors. Figure 2 shows samples from these two sources in our EPD dataset.



Figure 2. Image samples in our EPD dataset. The first and second images were captured by Pleiades satellite, while the third image and fourth images were collected from Google Earth. All image samples in our dataset were obtained from these two sources and image formats are all processed multi-spectral image products.

As shown in Figure 2, the electric pylon targets in high-resolution optical remote sensing images have a variety of features, which bring considerable challenges to actual detection tasks. On the one hand, due to the wide use of pylons, the sizes and specifications of pylons vary greatly. Even at the same spatial resolution, the area occupied by different pylons in the same image may be quite different. On the other hand, due to the wide coverage of the power network, the background environment of the pylons varies greatly. Light and topography also affect the characteristics of electric pylons. The former will impact the appearance color of electric pylons, while the latter will affect the tilt degree of electric pylons with the certain observation angle of satellites.

To test the adaptability of several detectors to the above interference factors, we comprehensively selected the electric pylons targets in different states when making our dataset. EPD dataset contains 1500 images in total: 720 images were captured by Pleiades satellite along Huimao Line in Guangdong Province, China, a main line of power network in south China, while the remaining images were

collected from Google Earth to further improve the representativeness of the dataset by expanding the source of samples. The spatial resolution of images in EPD dataset is 1 m/pixel.

Moreover, to test and evaluate the adaptability of the detectors in face of actual situations, we specially selected 50 relatively complex images from EPD dataset comprising a complex test subset called EPD-C. Twenty images in subset EPD-C were from the production of Pleiades satellite while the remaining 30 images were from Google Earth. One criterion for selecting images to EPD-C is the interference of the background, such as the similarity between the background and the target or interfering objects in the image containing certain similar characteristics with the target to be detected. Another criterion includes the particularity of the target to be detected, such as the large scale variation or unique characteristics. The details of the complex test set EPD-C are summarized in the Table 1.

Table 1. Details of the complex test subset EPD-C.

20 Images from Pleiades Satellite		
Features and Background	Number of Images	Number of Targets
green fields	2	8
multicolored fields	2	5
mountains	2	2
towns+multicolored fields	2	4
towns+mountains	4	6
mountains+multicolored fields	2	3
lakes	2	6
complex terrain	4	4
30 Images from Google Earth		
Features and Background	Number of Images	Number of Targets
frame architectures	6	32
multicolored fields	6	14
shadows	1	2
highways	2	5
special electric pylons	3	8
small targets	4	27
large size variation	2	18
complex terrain	6	15

Figure 3 shows two samples in the complex test subset EPD-C. We can see that the left image from Pleiades satellite has a colorful field background, similar to the color of electric pylons. The right image in Figure 3 is from Google Earth, and its background contains crisscrossed roads and framed buildings. It would be harder to detect electric pylons in these two images. There are totally 159 electric pylons in EPD-C. Thus, the construction of EPD-C can help to better evaluate electric pylon detection performance in complex situations.

Moreover, these two samples in Figure 3 also indicate that, in addition to the characteristics of electric pylons themselves, the surrounding background also brings challenges to the detection task, which is mainly reflected in the background color and interference targets. Due to the light-colored frame structure of electric pylon target, light background and frame structure buildings can significantly interfere with detection results.

Particularly, we regard the remaining 1450 images in EPD dataset excluding EPD-C as a standard subset named EPD-S, which involves more than 3000 electric pylons. EPD-S subset was used to train detectors and perform random experiments.



Figure 3. Image samples in the complex test subset EPD-C. The **left** one was captured by Pleiades satellite, where the detection difficulty mainly lies in the similarity of color characteristics between the background and electric pylon targets. The **right** one was collected from Google Earth, where the detection difficulty mainly reflects on interference from frame structure buildings.

3.2. Deep Learning Detectors for Comparison

We selected 10 popular state-of-the-art deep-learning-based detectors, namely Faster R-CNN [19], Cascade R-CNN [20], Grid R-CNN [22], Libra R-CNN [39], Retinanet [29], YOLOv3 [27], YOLOv4 [28], Retinanet FreeAnchor [40], FCOS [32], and Retinanet FSAF (Feature Selective Anchor-Free) [41], containing four two-stage models and six one-stage models. From the perspective of whether to use anchor, eight detectors are anchor-based, one detector is anchor-free, and one is specially an anchor-based detector with an anchor-free branch. We selected these detectors based on the following reasons. Firstly, these 10 detectors are popular deep learning models proposed in the last five years. Their performance can almost stand for the ability of state-of-the-art deep-learning-based detectors to solve electric pylon detection. Secondly, these models have already been applied in many other remote sensing tasks and have obtained meaningful achievements. Lastly, these models cover the main research directions of deep-learning-based object detection network, such as two-stage/one-stage and anchor-based/anchor-free, making our study more comprehensive and the experimental results more credible. Some details of the detectors we studied in this paper are reported in Table 2. For more detailed introduction to how each detector works, please refer to the respective citations.

Table 2. Detectors based on deep learning studied in this paper. Eight detectors use ResNet101 [42] + FPN [43] as the backbone, while the other two detectors of YOLO series use Darknet-53 [27] and CSPDarknet-53 [28] as the backbone, respectively. ResNet101 refers to a deep residual network with 101 layers. FPN refers to feature pyramid networks. Darknet-53 refers to a deep residual network with 53 layers and CSPDarknet adds a CSPNet [44] structure on the basis of Darknet-53.

Detectors	Backbone	Category
Faster R-CNN [19]	ResNet101+FPN	two-stage, anchor-based
Cascade R-CNN [20]	ResNet101+FPN	two-stage, anchor-based
Grid R-CNN [22]	ResNet101+FPN	two-stage, anchor-based
Libra R-CNN [39]	ResNet101+FPN	two-stage, anchor-based
Retinanet [29]	ResNet101+FPN	one-stage, anchor-based
YOLOv3 [27]	Darknet-53	one-stage, anchor-based
YOLOv4 [28]	CSPDarknet-53	one-stage, anchor-based
Retinanet FreeAnchor [40]	ResNet101+FPN	one-stage, anchor-based
FCOS [32]	ResNet101+FPN	one-stage, anchor-free
Retinanet FSAF [41]	ResNet101+FPN	one-stage, anchor-based with anchor-free

3.2.1. Backbone Network

In deep learning networks for object detection task, the structure utilized to extract features is called the backbone network. As a relatively new network structure with excellent performance in various tasks, deep residual network (ResNet) [42] is very suitable to be the backbone network of object detection models. In this paper, most networks use ResNet101 (ResNet with 101 layers) as the backbone network, except YOLOv3 and YOLOv4, which, respectively, use Darknet-53 and CSPDarknet-53 as the backbone network. Besides, considering that feature pyramid networks (FPN) [43] has a good performance in solving multi-scale problems, we combine FPN with ResNet to further improve network performance.

As shown in Figure 4, the output of the last four stages (C_2 – C_5) out of total five stages of ResNet, are input again to the FPN network. The output of the backbone network is the fused feature map of ResNet and FPN. We record the outputs as P_2 – P_5 . The P_5 layer is obtained by 1×1 convolution of C_5 . P_4 layer is obtained by fusion of P_5 layer and C_4 layer after upper sampling. P_3 and P_2 are computed by a similar operation as P_4 . The numbers of channels inputting each layer are 256, 512, 1024, and 2048, respectively, and the number of output channels is 256. Each layer has half the size of the previous layer. Meanwhile, FPN increases the P_6 layer obtained by the secondary down sampling of C_5 layer, which adds a feature layer on a larger scale. Besides, the structures of Darknet-53 and CSPDarknet-53 are detailed in Section 3.2.7 and Section 3.2.8, respectively.

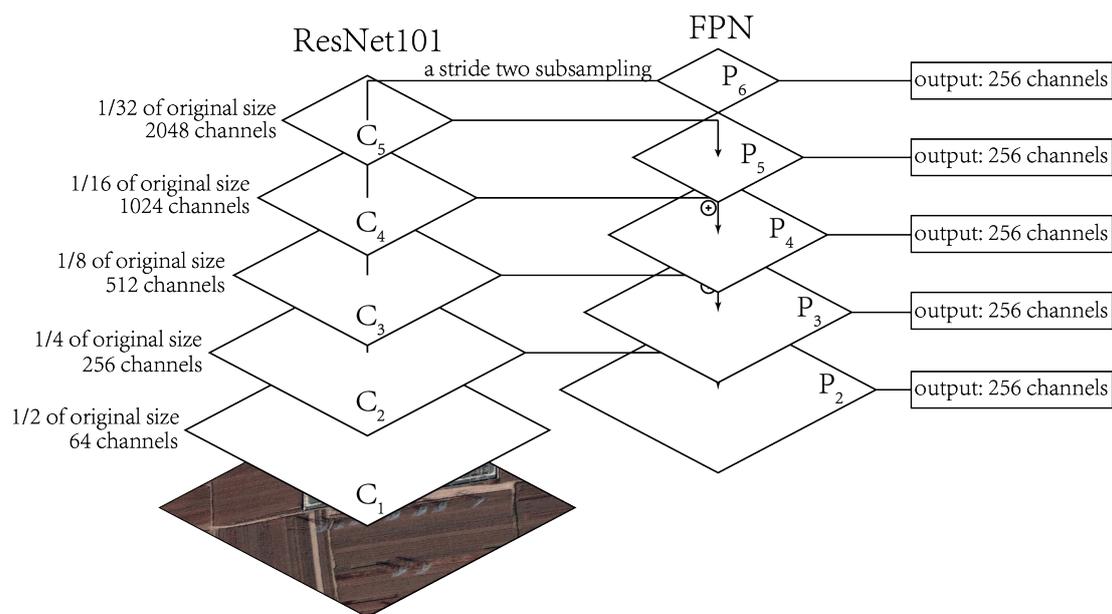


Figure 4. Structure of ResNet101 [42] FPN [43]. The left part shows the structure of ResNet101, which utilizes as the bottom-up pathway. ResNet utilizes a residual learning framework, deepening neural networks by shortcut connections. ResNet outputs five stages of feature maps, C_1 – C_5 , which have features of different scales. The sizes and channels of C_1 – C_5 are shown on their left. The right part shows the top-down structure of FPN. FPN is a component to acquire and merge multi-scale features. \oplus means up-sampling coarser-resolution feature maps and merging it with the corresponding bottom-up map. P_2 – P_6 which have 256 channels are the output of FPN, imported to different detectors.

3.2.2. Faster R-CNN

Faster R-CNN [19] creatively merges Region Proposal Network (RPN) to Fast R-CNN [18]. An RPN is a fully convolutional network, trained end-to-end to generate high-quality candidate boxes for detection by Fast R-CNN. RPN accelerates the region proposal and reduces the running time of two-stage detectors significantly. The structure of Faster R-CNN is shown in Figure 5.

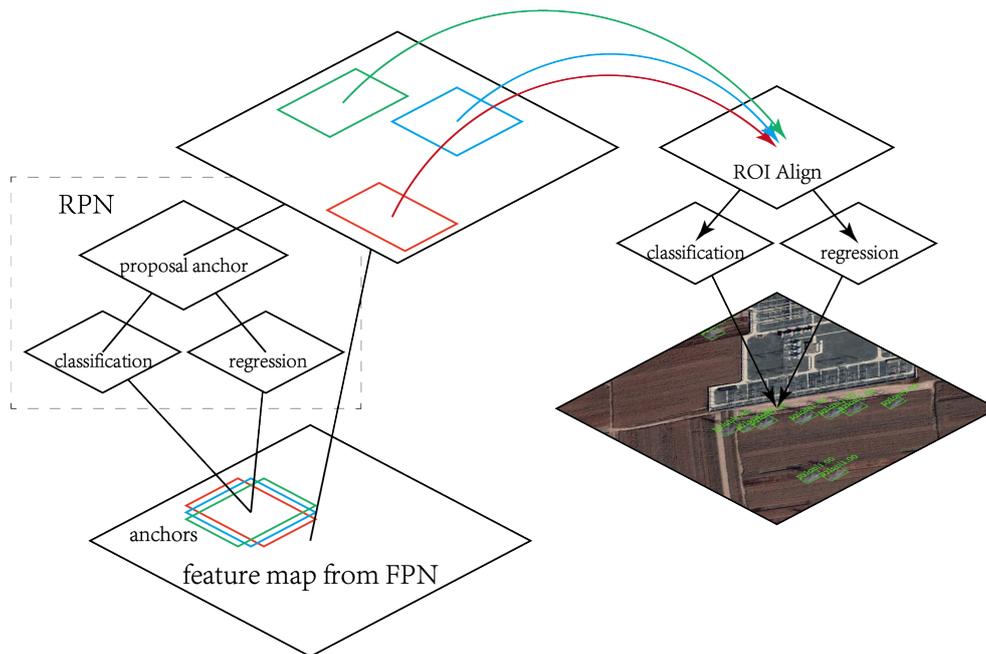


Figure 5. Structure of Faster R-CNN [19]. The input of Faster R-CNN is the output of FPN [43]. Tetragons with different colors represent different anchors. RPN (Region Proposal Network) is a fully convolutional network, classifying the anchors to foreground/background and regressing the bounding-box sketchily. ROI is the abbreviation of region of interesting and ROI Align layer [21] is used to reconcile the size of feature maps.

Due to the utilization of FPN, the feature maps input to RPN have multi-scale characteristic. The scale of anchors in each feature map is 8^2 , equivalent to generate five scales of anchors (32^2 , 64^2 , 128^2 , 256^2 , and 512^2) in the original image. The aspect ratios of anchors are 1:1, 2:1, and 1:2. We generated 15 kinds of anchor totally.

Faster R-CNN usually uses ROI pooling layer to reconcile the size of feature maps input into two fully connected (FC) layers. However, in this paper, we utilized a modified method of ROI pooling, i.e., ROI Align [21], as shown in Figure 5. ROI Align layer outputs a feature map with a shape of $7 \times 7 \times 256$, and the FC layer outputs 1024 channels.

In RPN, we utilized Cross Entropy Loss (CE Loss) [19] with sigmoid function when calculating classification loss and Smooth L1 Loss [19] with parameter $\beta = 1/9$ when calculating bounding-box regression loss. In R-CNN, we utilized CE Loss with SoftMax method when calculating classification loss and Smooth L1 Loss with parameter $\beta = 1$ when calculating bounding-box regression loss. We added these losses together as the final loss.

Faster R-CNN adopts the parameterizations of four coordinates $[x, y, w, h]$ to regress from an anchor box to a nearby ground-truth box. In R-CNN, we utilized mean values 0 and variances $[0.1, 0.1, 0.2, 0.2]$ to normalize the four coordinates.

When training RPN, we assigned positive and negative labels to anchors following respective citation [19]. We sampled anchors as well. In particular, if the maximum IoU (Intersection over Union) between a ground-truth box and any anchor was lower than 0.3, we ignored entire anchors. To reduce redundant anchor, we adopted non-maximum suppression (NMS) [45]. We set the IoU threshold for NMS as 0.7, and calculated 2000 proposal regions with the highest scores per image.

When training R-CNN, we simply assigned positive and negative labels with 0.5 as a threshold, and we utilized ground truth as a positive sample. We randomly sampled 512 anchors with the positive and negative ratio 1:3.

When testing, we utilized NMS as well. In RPN, we also set the IoU threshold for NMS as 0.7, and calculated 1000 proposal regions per image. In R-CNN, we set the IoU threshold for NMS as 0.7 and calculated 100 proposal regions per image.

3.2.3. Cascade R-CNN

Faster R-CNN [19] utilizes 0.5 as the IoU threshold when defining positive and negative samples. A low threshold, e.g., 0.5, usually produces noisy detection in positive samples, but a high threshold, e.g., 0.7, usually leads positive samples to exponentially vanish and causes detection performance to degrade. Cascade R-CNN [20] was proposed to address the problem. As shown in Figure 6, Cascade R-CNN cascades a sequence of detectors with increasing IoU thresholds. The detectors are trained stage by stage. The output of a detector is a superior distribution for training the next higher quality detector.

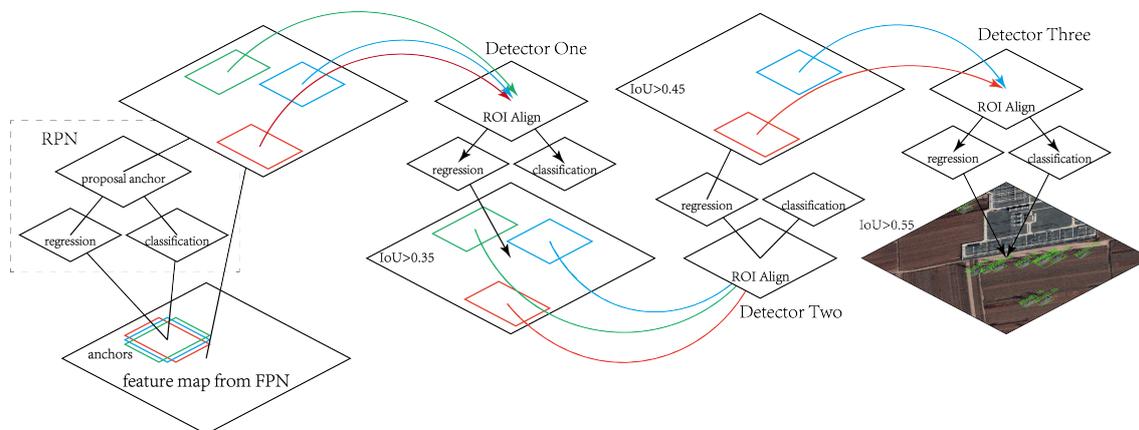


Figure 6. Structure of Cascade R-CNN [20]. The input of Cascade R-CNN is the output of FPN [43]. Tetragons with different colors represent different anchors, while the tetragons on different feature maps with the same color represent the same anchors. RPN (Region Proposal Network) is a fully convolutional network, classifying the anchors to foreground/background and regressing the bounding-box sketchily. ROI is the abbreviation of region of interesting and ROI Align layer [21] is used to reconcile the size of feature maps. IoU is the abbreviation of Intersection over Union, whose details are introduced in Section 4.2. Detectors 1 and 2 use different IoU thresholds and the positive ones are imported to next detector, as shown by the lines of different colors.

We cascaded three detectors with the IoU thresholds 0.35, 0.45, and 0.55, respectively. We selected these IoU thresholds to calculate more anchors and improve the ability of finding small objects. The detectors randomly sampled 1024, 512, and 512 anchors, respectively. The positive and negative ratio for each detector was 1:3. For the three detectors, we normalized the coordinates $[x, y, w, h]$ with mean values 0 and variances $[0.1, 0.1, 0.2, 0.2]$, $[0.05, 0.05, 0.1, 0.1]$, and $[0.033, 0.033, 0.067, 0.067]$, respectively. For each detector, we utilized the same loss function as Faster R-CNN, and the weights of three detectors are $[0.25, 1, 0.5]$. Other parameters of Cascade R-CNN are identical to Faster R-CNN.

3.2.4. Grid R-CNN

Grid R-CNN [22] substitutes a grid guided localization mechanism for precise object detection instead of traditional regression based methods. The detector designs a spatial information fusion module to utilize the inner spatial correlation and calibrate the location of grid points. The fusion feature maps are obtained by adding correlative grid feature maps processed by convolution layers together. Grid R-CNN extends region mapping to cover all the target grid points of positive proposal as well.

The authors also presented a better and faster version of Grid R-CNN, Grid R-CNN Plus. The major update is the proposed grid point specific representation. Grid R-CNN Plus solves the problem that the ground truth label is obliged to a small region on the supervision map by shifting a biased distribution to a normalized one. More details of Grid R-CNN Plus can be found in respective citations [46].

The structure of Grid R-CNN used in this paper is shown in Figure 7. We established Grid Head mainly following [46]. The loss function of Grid Head is CE Loss with sigmoid function, and the weight of CE Loss is 15. In addition, we utilized Group Normalization (GN) [47] during calculating the convolution of grid features. Other parameters of Grid R-CNN are identical to Faster R-CNN.

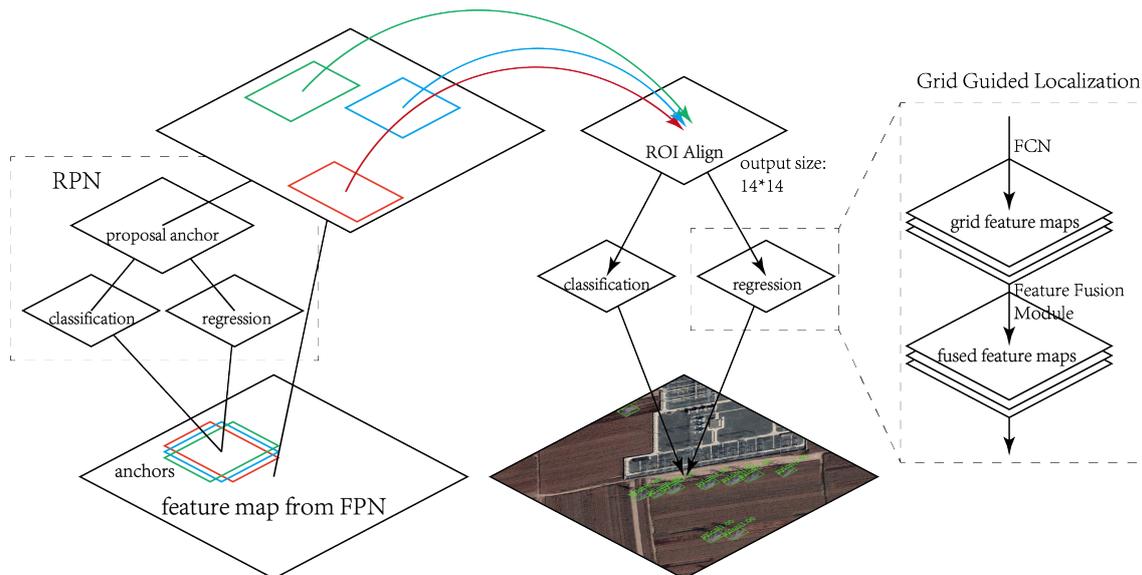


Figure 7. Structure of Grid R-CNN [22]. The input of Grid R-CNN is the output of FPN [43]. Tetragons with different colors represent different anchors. RPN (Region Proposal Network) is a fully convolutional network, classifying the anchors to foreground/background and regressing the bounding-box sketchily. ROI is the abbreviation of region of interesting and ROI Align layer [21] is used to reconcile the size of feature maps. Grid Guided Localization [22] adopts a fully convolutional network to obtain grid feature maps and fuses them.

3.2.5. Libra R-CNN

Pang et al. proposed Libra R-CNN [39]. They divided the object detector training into three stages: (1) sampling regions; (2) extracting features; and (3) recognizing the categories and refining the locations under the guidance of a loss function. Three levels of imbalance exist during the training process, i.e., sample level, feature level, and objective level. Pang et al. proposed three novel components, namely IoU-balanced sampling, balanced feature pyramid, and balanced L1 loss, for reducing these imbalances separately. The structure of Libra R-CNN is shown in Figure 8.

IoU-balanced sampling splits the sampling interval into even bins according to IoU and samples from them uniformly. We split three bins and utilized IoU-balanced sampling in R-CNN. In RPN, we restricted the maximum ratio of negative sample to five. Balanced feature pyramid resizes the feature maps extracted from the former network structure identically, calculates average value, resizes to original size, and superposes with original feature maps. We resized the feature maps to the size of C4, and utilized non-local module [48] to extract features. Balanced L1 loss implements a more balanced training of the detector. We utilized balanced L1 loss with parameter $\alpha = 0.5$, $\gamma = 1.5$, and $\beta = 1.0$ when calculating bounding-box regression loss. We set NMS following Pang et al. [39]. Other parameters of Libra R-CNN are identical to Faster R-CNN.

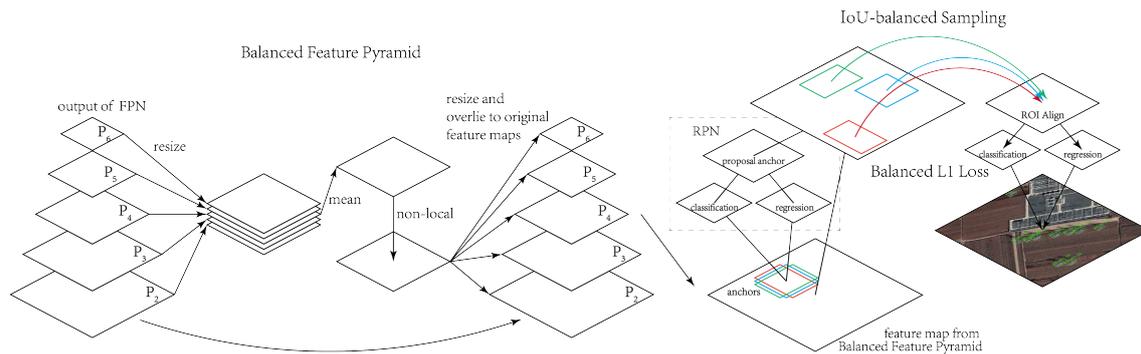


Figure 8. Structure of Libra RCNN [39]. Libra R-CNN updates the output of FPN [43], P_2 – P_6 , with Balanced Feature Pyramid and the updated feature maps are imported to next structure. Tetragons with different colors represent different anchors. RPN (Region Proposal Network) is a fully convolutional network, classify the anchors to foreground/background and regress the bounding-box sketchily. ROI is the abbreviation of region of interesting and ROI Align layer [21] is used to reconcile the size of feature maps. Libra R-CNN utilizes IoU-balanced Sampling to sample the anchors and Balanced L1 Loss to calculate classification loss. Details of Balanced Feature Pyramid, IoU-balanced Sampling, and Balanced L1 Loss can be found in [39].

3.2.6. Retinanet

Retinanet [29] is a well-performing one-stage object detector. One-stage detectors usually extract features and make classification and regression directly. They output multi-channel maps which represent the classification and regression information. Retinanet addresses one of the critical problems that lead to low detection precision, i.e., the extreme foreground–background class imbalance. It proposes a novel loss function, focal loss. Focal loss focuses on a sparse set of hard examples and prevents the vast number of easy negatives from overwhelming the detector during training. The primary structure of Retinanet is shown as Figure 9.

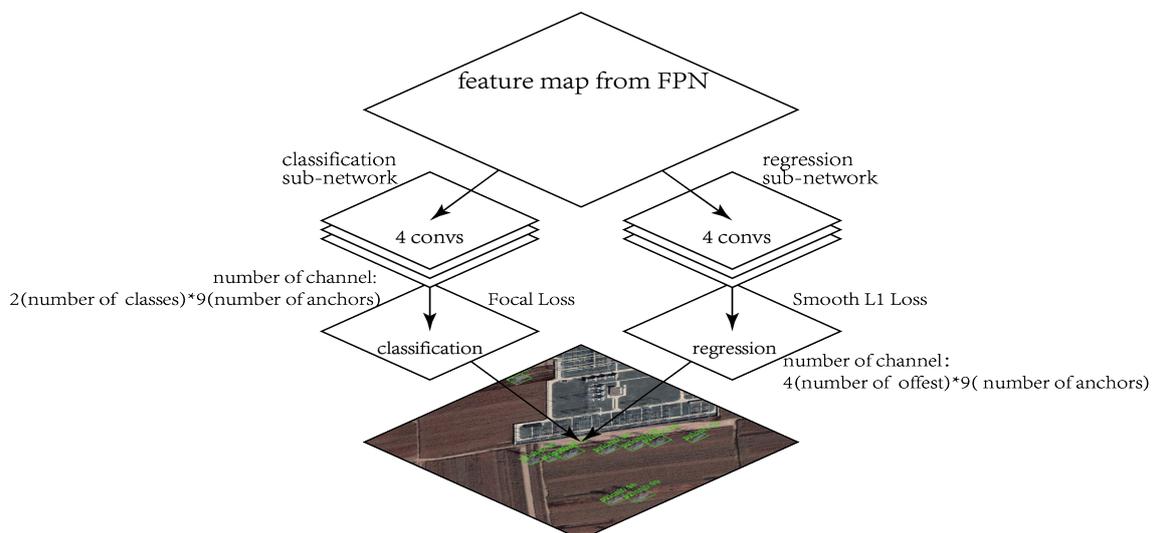


Figure 9. Structure of Retinanet [29]. The input of Retinanet is the output of FPN [43]. Retinanet is divided into classification sub-network and regression sub-network. In this paper, Classification sub-network outputs feature maps with 18 channels, equaling the number of classes multiplied by the number of anchors. Focal Loss is used in classification sub-network. Regression sub-network outputs feature maps with 36 channels, equaling the number of offsets multiplied by the number of anchors. Regression sub-network utilizes smooth L1 loss [19].

The FPN we utilized for Retinanet was slightly different from the one in Section 3.2.1. We constructed a pyramid with levels P_3 through P_7 . P_6 is computed by a 3×3 stride-2 convolution on C_5 , and P_7 is obtained via applying ReLU function [49] followed by a 3×3 stride-2 convolution on P_6 .

We established the size of anchors according to Lin et al. [29]. For focal loss, we utilized it when calculating classification loss, with parameters $\gamma = 2.0$ and $\alpha = 0.25$. For regression loss, we utilized Smooth L1 Loss with parameter $\beta = 1/9$.

During training, we assigned anchors positive with an IoU threshold of 0.5 and negative with 0.4. We also assigned the anchors positive when they had the maximum IoU with a ground-truth box. However, we ignored entire anchors if the maximum IoU was lower than 0.1. We did not normalize $[x, y, w, h]$ in Retinanet. In addition, other parameters are identical to Faster R-CNN.

3.2.7. YOLOv3

You Only Look Once (YOLO) [25] is a typical one-stage object detector proposed by Joseph Redmon in 2016. Afterwards, the author had made two improved versions, including YOLOv2 [26] and YOLOv3 [27]. In the series, YOLOv3 is a relatively new detector and can reach a well performance. Specifically, YOLOv3's network for feature extraction is Darknet-53, referring the residual network idea of ResNet. The basic composition unit of Darknet-53 is "1 \times 1 convolution module + 3 \times 3 convolution module + residual module". Darknet-53 retains the strategy of leaky ReLU layer and batch normalization layer in the former series. Besides, the input images are processed by a total of five times of down sampling. Darknet-53 also refers the idea of the multi-scale feature layers in FPN, and layers of the last three scales are selected as output. For example, the ratio of output feature layers size is 1:2:4 when the size of input image is 1024^2 . Furthermore, to improve the expression of shallow feature layers and make full use of the information of each feature layers, the relatively shallower layer will be superimposed with the deeper feature layer which has been processed by up-sampling. Figure 10 shows the structure of YOLOv3 network.

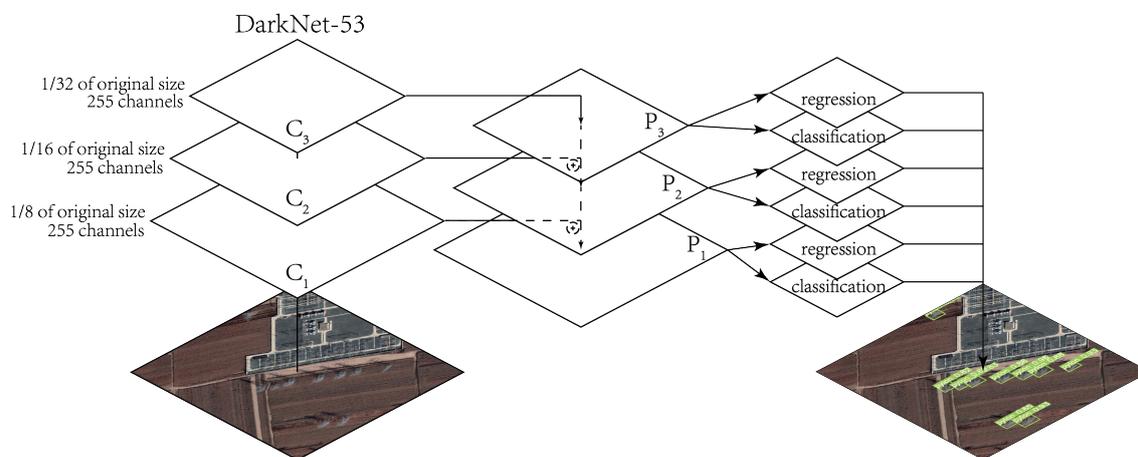


Figure 10. Structure of YOLOv3 [27]. Darknet-53 is a residual network mainly constructed of 1 \times 1 and 3 \times 3 convolution module. C_1 – C_3 refer to the feature layers of the last three scales obtained after five down sampling of the input image in Darknet-53. P_1 – P_3 are obtained from C_1 – C_3 feature layers after superimposition of adjacent layers through upper sampling. \oplus refers to the superimposition operation.

The size of the nine anchors given in [25] was calculated by k-means clustering. We also used k-means clustering on the EPD-S subset, and the sizes of the anchors were modified to 26×22 , 30×41 , 35×60 , 45×29 , 47×48 , 61×86 , 63×35 , 88×55 and 168×140 . Considering the case that a single target may have multiple labels, YOLOv3 replaces the SoftMax classifier with sigmoid classifier. In addition, focal loss was selected as the loss function with $\gamma = 0.8$ and $\alpha = 1.0$.

3.2.8. YOLOv4

Recently, Alexey et al. proposed the latest YOLOv4 [28] on the basis of YOLO series. Compared with YOLOv3, YOLOv4 has better overall performance and can receive well detection result on a single GPU. To reach a fast and accurate detector, the authors carefully sifted and tested the typical algorithm modules commonly used in the deep learning models, and further designed and improved some modules. In particular, the improvement mainly focused on the selection of backbone and combination of several tricks. On the basis of choosing CSPDarknet-53 as the backbone of the detector, the authors added SPP block [17] to extend the receptive field of the model and utilized modified PANet instead of FPN. As for Tricks, the authors selected the most suitable methods for YOLOv4 from the commonly used deep-learning-based detection modules, including choosing Mish as the activation function, DropBlock as the regularization method, etc. Moreover, YOLOv4 utilizes a new method of data augmentation call Mosaic, which expands the data by stitching together four images. To adapt YOLOv4 to single GPU training, the authors also improved several existing methods, including SAM, PANet, Cross mini-Batch Normalization (CMBN), etc. In general, the main structure of YOLOv4 can be summarized as "CSPDarknet-53+SPP+PANet+YOLOv3 Head+Tricks", which is shown in Figure 11.

In addition, we used anchors of the same size in YOLOv4 as YOLOv3, which were obtained through k-means clustering. The sizes of the nine anchors were, respectively, 26×22 , 30×41 , 35×60 , 45×29 , 47×48 , 61×86 , 63×35 , 88×55 , and 168×140 . Except for setting the parameter random to 0, we utilized all the default parameters provided by the author in our experiment.

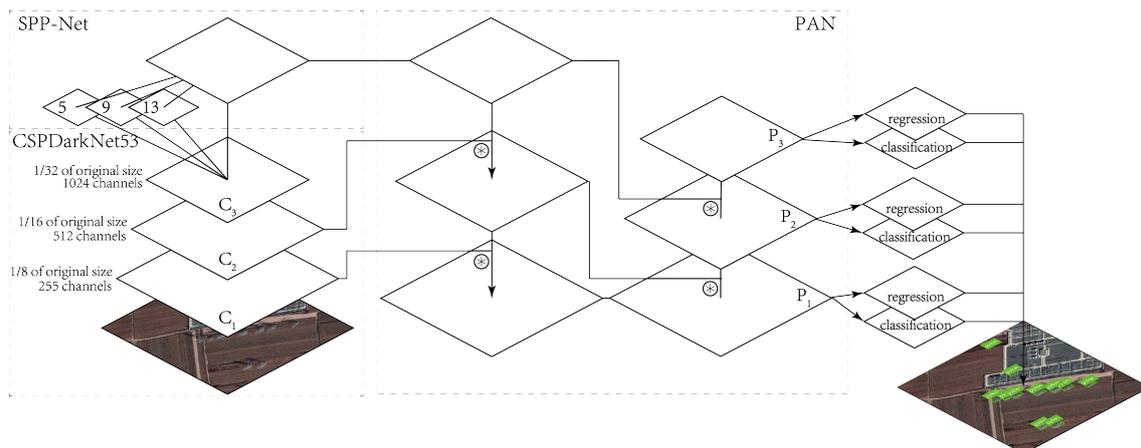


Figure 11. Structure of YOLOv4 [28]. The same as Darknet-53, C_1 – C_3 refer to the feature layers of the last three scales obtained after five down-sampling of the input image. CSPDarknet-53 also uses the structure of CSPNet [44] and the Mish activation function. SPP refers to spatial pyramid pooling, and C_3 layer are disposed with 5×5 , 9×9 , and 13×13 pooling operations, respectively. P_1 – P_3 are obtained from C_1 – C_3 feature layers after aggregation of adjacent layers through two times of down-sampling and up-sampling. \oplus refers to the aggregation operation.

3.2.9. Retinanet FreeAnchor

FreeAnchor [40] utilizes probability theory to guide object–anchor matching, and updates hand-crafted anchor assignment to “free” anchor matching by expressing detector training as a maximum likelihood estimation (MLE) [50] procedure. This method establishes bags of candidate anchors for different objects and learns an object–anchor matching approach. We used FreeAnchor module working jointly with Retinanet (named Retinanet FreeAnchor). The structure of Retinanet FreeAnchor is shown in Figure 12.

We utilized the same loss function as Zhang et al. [40]. The parameter γ of focal loss was 2.0 and α was 0.5. We normalized $[x, y, w, h]$ with mean values 0 and variances $[0.1, 0.1, 0.2, 0.2]$. Other parameters of Retinanet FreeAnchor were identical to Retinanet.

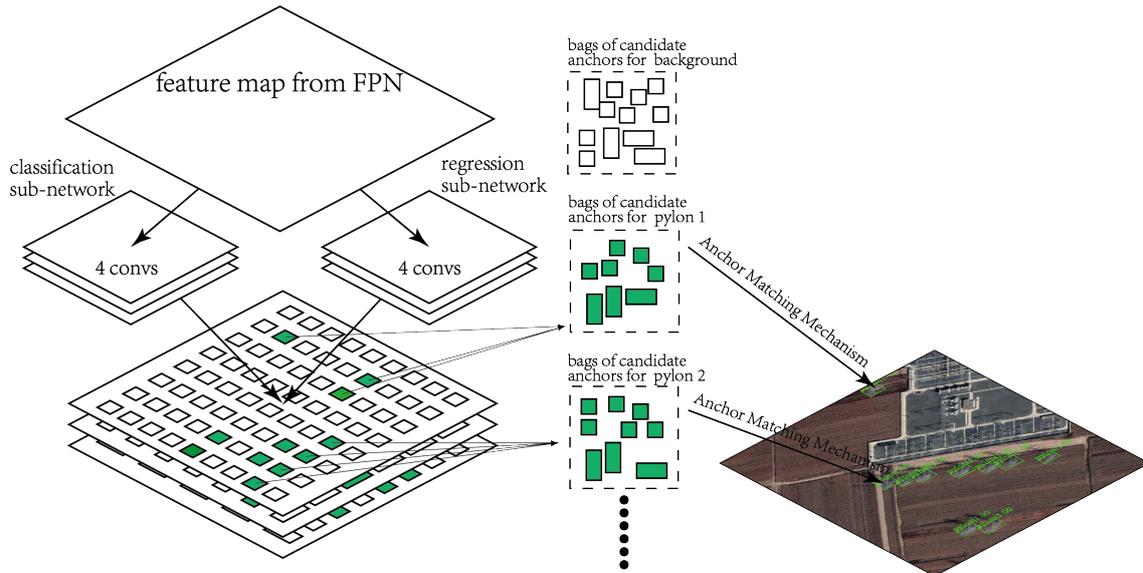


Figure 12. Structure of Retinanet FreeAnchor [40]. Retinanet FreeAnchor has the same input, classification, and regression sub-network as Retinanet [29]. The green-filled tetragons mean anchors containing objects, while the white-filled tetragons means anchors containing nothing. The tetragons put in the same bag of candidate contain the same object. The Anchor Matching Mechanism implements the learning-to-match approach.

3.2.10. FCOS

FCOS [32] is a typically anchor-free detector. Anchor-free detectors do not generate anchors, but gain the classification and regression information directly by convolutions. The structure of FCOS is shown in Figure 13.

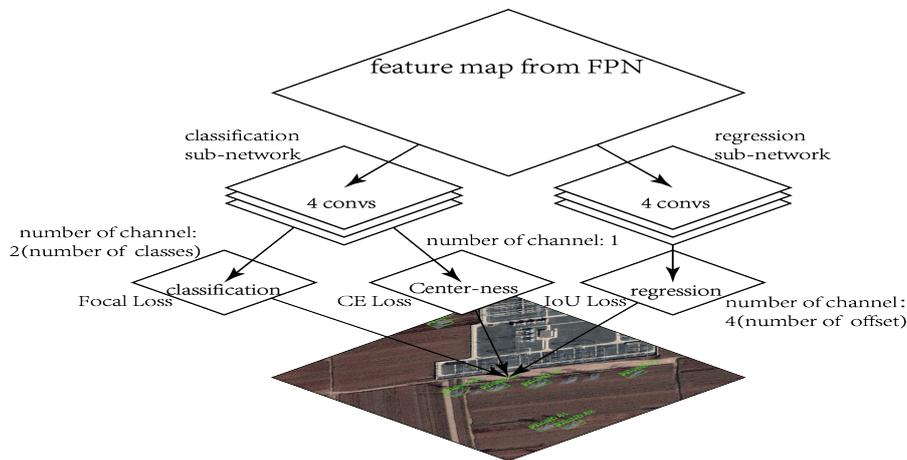


Figure 13. Structure of FCOS [32]. The structure of FCOS is similar with Retinanet [29]. The regression sub-network outputs four channels which present four offsets, while the classification sub-network outputs two channels presenting two categories. IoU loss [51] and focal loss [29] are used as regression and classification loss, respectively. Classification sub-network outputs another channel to restrain the inferior quality outer points, using center-ness loss.

During training, if a point was inside a ground-truth box of an object, we assigned it positive. Otherwise, we assigned it negative. To deal with the problem caused by ground-truth box overlapping, we restricted the range of regression distance in each level of the feature as [32].

For loss function, we utilized the same focal loss as Retinanet and the IoU Loss [51]. In addition, FCOS proposed Center-ness Loss to restrain the inferior quality outer points. We utilized CE Loss with sigmoid function to implement Center-ness Loss. The weights of these three loss functions were equal.

We also used GN with group number 32 in the sub-networks. For other parameters in FCOS, we utilized those recommended by Tian et al. [32]. The rest of parameters were identical to Retinanet.

3.2.11. Retinanet FSAF

FSAF [41] is an anchor-free module. In traditional multi-scale feature extracting networks, e.g., FPN, it is hard to select the best feature level for every object when training. FSAF can be plugged into one-stage detectors with feature pyramid structure. We used FSAF module working jointly with Retinanet (called Retinanet FSAF) as the structure shown in Figure 14.

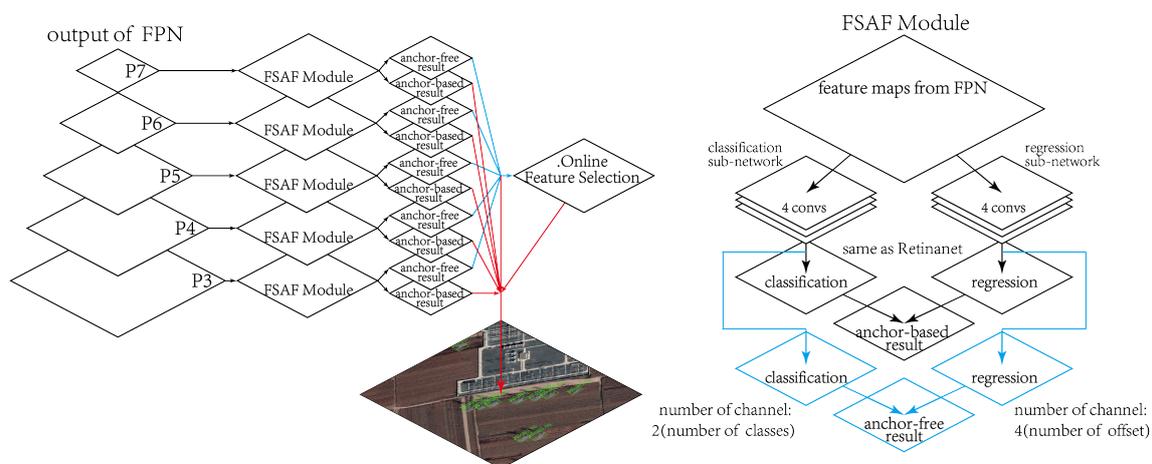


Figure 14. Structure of Retinanet FSAF [41]. Retinanet FSAF combines an anchor-free module, FSAF module, with Retinanet [29]. The right part shows the structure of FSAF module, which has the same anchor-based branch as Retinanet. The blue part shows the structure of anchor-free branch. Regression sub-network outputs four channels which present four offsets, while the classification sub-network outputs two channels which present two categories. IoU loss [51] and focal loss [29] are used as regression and classification loss, respectively. The left part shows the global structure of Retinanet FSAF. Blue lines mean utilizing the results of anchor-free branch to select the best feature levels for training objects, while the red ones represent object detection on the selected feature map. Details of feature map selection can be found in [41].

We established FSAF according to Zhu et al. [41], utilized IoU loss [51] as the regression loss and focal loss as the classification loss, and utilized Online Feature Selection to select the best feature level for training objects. Other parameters of Retinanet FSAF were identical to Retinanet.

3.3. Training Details

3.3.1. Detector Implementation

We built our detectors based on the MMDetection <https://github.com/open-mmlab/mmdetection> toolbox, an open source object detection toolbox based on PyTorch developed by Multimedia Laboratory, The Chinese University of Hong Kong. In particular, the source code of Retinanet FSAF was taken from the community of MMDetection <https://github.com/open-mmlab/mmdetection/pull/675>. For YOLOv3, we built our detector on the basis of PyTorch YOLOv3 software developed by Ultralytics LLC <https://github.com/ultralytics/yolov3>. For YOLOv4, we utilize the code provided on the author's website <https://github.com/AlexeyAB/darknet>.

In addition, all detectors used pre-trained weights to fine-tune, saving a lot of training time. The pre-trained models were trained on ImageNet. Pre-trained weights of ResNet101 are provided

by PyTorch website <https://download.pytorch.org/models/resnet101-5d3b4d8f.pth>, the weights of DarkNet-53 is from the website of the authors of YOLOv3 <https://pjreddie.com/media/files/yolov3.weights>, and the weights of YOLOv4 is from the website of the authors https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v3_optimal/yolov4.conv.137. For transfer learning, we froze shallow parameters of the pre-trained models except for YOLOv3 and YOLOv4. In ResNet101, we did not update the parameters of layer C_1 and C_2 .

3.3.2. Details of Detector Training

For all detectors based on MMDetection, we randomly flipped the input images with a probability of 0.5 for data augmentation. We also normalized the input images using mean values [123.675, 116.28, 103.53] and variances [58.395, 57.12, 57.375] recommended by MMDetection.

For YOLOv3, four data augmentation parameters provided by Ultralytics LLC were utilized to generate more training samples by rotating the angle and adjusting the saturation, exposure and hue. For YOLOv4, four data augmentation parameters provided by the author were also utilized to generate more training samples by rotating the angle and adjusting the saturation, exposure and hue. Moreover, YOLOv4 utilizes a new method of data augmentation called Mosaic, which is introduced in Section 3.2.8.

When training, all detectors based on MMDetection utilized stochastic gradient descent (SGD) with momentum [52] as the optimizer. The momentum was set as 0.9 and the weight decay was 0.0001. YOLOv3 also utilized SGD as the optimizer, with momentum 0.9 and weight decay 0.0005. YOLOv4 utilized Adam [53] as the optimizer, with momentum 0.9 and weight decay 0.0005. We selected the initial learning rate appropriately and set the train epoch big enough to ensure that all models could be thoroughly trained. For learning rate, we utilized learning rate descending strategies. When the epoch reached a certain value of Epoch1 and Epoch2, the learning rate decreased to 1/10 of the original. We used learning rate warm-up method [42] on majority of the detectors. This method could warm up the detectors by using small learning rate at the beginning of training. We utilized linear growth methods, the warm-up iteration was 500, and the warm-up ratio was 1/3. The initial learning rates (lr) and epochs are shown in Table 3. It needs to be emphasized that the parameters used by all detectors are the optimal parameters obtained after adjustment in our opinion, which ensures the fairness and accuracy of performance comparison among different detectors for the task of electric pylon detection in remote sensing images.

Table 3. Parameter settings in training. Initial lr, initial learning rate; Epoch1 and Epoch2: All detectors utilized the STEP lr decline strategy, and the lr decreased to 0.1 times at Epoch1 and Epoch2; Batch size: 1 m and 1–2 m refer to the resolution of the samples in training process.

Detectors	Initial lr	Total Epochs	Epoch1	Epoch2	Whether Warm-Up	Batch Size (1-m Detector)	Batch Size (1–2-m Detector)
Faster R-CNN [19]	0.005	20	13	18	yes	2	4
Cascade R-CNN [20]	0.005	25	13	20	yes	2	4
Grid R-CNN [22]	0.01	20	12	17	yes	2	4
Libra R-CNN [39]	0.005	20	13	18	yes	2	4
Retinanet [39]	0.005	30	20	27	yes	4	4
YOLOv3 [27]	0.01	20	16	18	no	4	16
YOLOv4 [28]	0.0125	25	20	23	yes	4	4
Retinanet FreeAnchor [40]	0.0075	40	28	35	yes	4	4
FCOS [32]	0.001	35	23	32	yes	2	4
Retinanet FSAF [41]	0.005	30	23	28	yes	4	4

4. Experimental Results

4.1. Experimental Settings

To accurately evaluate the performance of electric pylon detectors based on deep learning, we carried out extensive experiments on our self-made EPD dataset described in Section 3.1. We repeated stochastic experiments to reduce the error generated by dataset partition. Particularly, nine detectors were trained and tested on the EPD-S subset that contains 1450 images. In each round of training, the EPD-S subset was randomly divided into 8:1:1, where 8/10 was used to train the detector, 1/10 was used to validate the trained detector, and 1/10 was used to test the performance. The detector performing the best average accuracy (AP) during validation was selected as the final detector, and then its AP gained in testing set was used to evaluate the generalization capability. To improve the accuracy of experimental results, we carried out 10 times the above experimental process of "randomly dividing dataset \rightarrow training detector \rightarrow validating detector \rightarrow testing detector" for each kind of deep learning detectors. The average value of APs gained from 10 times of stochastic tests was reported as the performance evaluation basis of the final detector.

Besides, considering complex real situation, the final detectors trained on the EPD-S subset were also tested on the EPD-C subset. The average value of APs gained from 10 tests was taken as an additional evaluation criterion. Noticing that the EPD-C subset was only used in testing process not in training, testing results on EPD-C subset could be more objective to evaluate the generalization capability of each detector.

For software environment, we utilized Python 3 (3.6.9 and 3.7.5 to run different programs), PYTORCH 1.0, CUDA 10.0, and CUDNN 7.6.4 to run all programs. For hardware environment, we trained and tested the detectors using one NVIDIA GeForce RTX 2080Ti GPU with 10G memory.

It should be noticed that the data and codes of this paper will be released to the public at our website: <https://github.com/qsjxyz/Electric-Pylon-Detection-in-RSI>.

4.2. Index for Evaluation

We use twod popular indicators, i.e., recall and average precision (AP), to evaluate the detection performance of detectors. These indicators are computed based on IoU as

$$IoU = \frac{B_{pred} \cap B_{truth}}{B_{pred} \cup B_{truth}} \quad (1)$$

where B_{pred} represents the predict bounding box and B_{truth} represents ground truth. The bounding box is considered to be correct when IoU surpasses a threshold. We set the value of threshold as 0.5, the same as Yao et al. [9]. When IoU exceeds 0.5, the testing result is a true positive (TP), otherwise it is a false positive (FP). When the detector predicts no target in a location containing a target, it is a false negative (FN). Then, we can calculate two metrics, precision (P) and recall (R), as

$$P = \frac{TP}{TP + FP} \quad (2)$$

$$R = \frac{TP}{TP + FN} \quad (3)$$

The output of detectors contains detections and their confidence scores. After sorting detections by confidence scores from high to low and computing the precision and recall of all detections, we can obtain a precision–recall curve (PRC). AP is the area under PRC.

Besides, to evaluate the practicability of the model more comprehensively, the testing speeds and model sizes of all detectors were also calculated on the same hardware platform and are reported in this paper. It should be noticed that all detectors in the experiments were evaluated based on the same criteria.

4.3. Performance of Detectors

Firstly, we evaluated the performance of detectors at the original resolution 1 m/pixel, and conducted 10 rounds of repeated stochastic experiments for each detector following the specific experimental operation in Section 4.1. The column *Batch size (1m detector)* in Table 3 shows batch sizes set in the training process. Considering the limitation of the memory of the graphics card and the training speed, we selected the maximum batch size for each detector that is available under the condition of hardware environment. Table 4 shows the results of 10 detectors trained with data of 1-m/pixel resolution. Moreover, to analyze the adaptability of detectors to complex environments, we tested each trained detector on the EPD-C subset. Table 5 presents the results.

Table 4. Results on the EPD-S subset. Train resolution, 1 m/pixel; test resolution, 1 m/pixel. AP refers to average precision. Recall, AP, and speed are expressed in the format of mean \pm standard deviation on the basis of 10 rounds of test results.

Detectors	Recall	AP	Speed (Images/s)	Model Size
Faster R-CNN [19]	0.917 \pm 0.018	0.871 \pm 0.041	8.52 \pm 1.00	482.4M
Cascade R-CNN [20]	0.906 \pm 0.022	0.876 \pm 0.028	7.10 \pm 0.22	704.8M
Grid R-CNN [22]	0.931 \pm 0.018	0.877 \pm 0.028	6.43 \pm 0.65	667.4M
Libra R-CNN [39]	0.929 \pm 0.019	0.872 \pm 0.026	8.88 \pm 0.30	484.5M
Retinanet [29]	0.935 \pm 0.013	0.891 \pm 0.012	7.99 \pm 1.03	442.3M
YOLOv3 [27]	0.939 \pm 0.016	0.887 \pm 0.023	10.95 \pm 0.18	246.4M
YOLOv4 [28]	0.898 \pm 0.016	0.887 \pm 0.026	16.21 \pm 0.24	256.0M
Retinanet FreeAnchor [40]	0.939 \pm 0.014	0.893 \pm 0.007	7.70 \pm 0.41	442.3M
FCOS [32]	0.938 \pm 0.023	0.874 \pm 0.020	11.08 \pm 0.60	408.6M
Retinanet FSAF [41]	0.944 \pm 0.011	0.888 \pm 0.013	8.38 \pm 0.29	441.5M

Table 5. Results on the EPD-C subset. Train resolution, 1 m/pixel; test resolution, 1 m/pixel. AP refers to average precision. Recall, AP, and speed are expressed in the format of mean \pm standard deviation on the basis of 10 rounds of test results.

Detectors	Recall rate	AP	Speed (Images/s)
Faster R-CNN [19]	0.697 \pm 0.021	0.654 \pm 0.035	9.62 \pm 0.25
Cascade R-CNN [20]	0.683 \pm 0.020	0.617 \pm 0.022	8.47 \pm 0.17
Grid R-CNN [22]	0.753 \pm 0.017	0.683 \pm 0.008	7.61 \pm 0.12
Libra R-CNN [39]	0.771 \pm 0.011	0.682 \pm 0.008	9.39 \pm 0.22
Retinanet [29]	0.763 \pm 0.028	0.695 \pm 0.018	10.21 \pm 0.23
YOLOv3 [27]	0.744 \pm 0.018	0.644 \pm 0.015	9.35 \pm 0.21
YOLOv4 [28]	0.754 \pm 0.019	0.689 \pm 0.015	12.11 \pm 0.19
Retinanet FreeAnchor [40]	0.752 \pm 0.027	0.687 \pm 0.014	10.09 \pm 0.41
FCOS [32]	0.758 \pm 0.031	0.622 \pm 0.020	11.82 \pm 0.15
Retinanet FSAF [41]	0.754 \pm 0.169	0.683 \pm 0.084	10.65 \pm 0.36

Furthermore, to obtain detectors with multi-resolution features, we trained models for 10 rounds on a mixed resolution dataset by down-sampling from the original EPD dataset. Table 6 shows the specific resolution distribution. The experimental results on mixed resolution EPD-S and EPD-C subsets are illustrated in Table 7 and Table 8, respectively.

Table 6. Data distribution of mixed resolution dataset obtained by down-sampling the images of 1-m resolution in the dataset.

EPD-S Subset		EPD-C Subset	
Resolution	Proportion	Resolution	Proportion
1.0 m	23.7%	1.0 m	32%
1.2 m	12.4%	1.2 m	2%
1.4 m	8.1%	1.4 m	12%
1.6 m	4.4%	1.6 m	8%
1.8 m	3.2%	1.8 m	6%
2.0 m	48.2%	2.0 m	40%

Table 7. Results on mixed resolution EPD-S subset. Train resolution, 1–2 m/pixel; test resolution, 1–2 m/pixel. AP refers to average precision. Recall, AP, and speed are expressed in the format of mean \pm standard deviation on the basis of 10 rounds of test results.

Detectors	Recall	AP	Speed (Images/s)	Model
Faster R-CNN [19]	0.889 \pm 0.022	0.820 \pm 0.031	8.62 \pm 0.27	482.4M
Cascade R-CNN [20]	0.898 \pm 0.024	0.841 \pm 0.040	7.94 \pm 0.24	704.8M
Grid R-CNN [22]	0.919 \pm 0.018	0.851 \pm 0.039	7.32 \pm 0.61	667.4M
Libra R-CNN [39]	0.910 \pm 0.015	0.841 \pm 0.036	9.79 \pm 0.56	484.5M
Retinanet [29]	0.912 \pm 0.017	0.832 \pm 0.032	8.84 \pm 0.27	442.3M
YOLOv3 [27]	0.921 \pm 0.010	0.843 \pm 0.026	14.28 \pm 0.38	246.4M
YOLOv4 [28]	0.892 \pm 0.017	0.879 \pm 0.018	22.66 \pm 0.32	256.0M
Retinanet FreeAnchor [40]	0.915 \pm 0.023	0.841 \pm 0.034	9.77 \pm 0.53	442.3M
FCOS [32]	0.892 \pm 0.019	0.792 \pm 0.020	10.96 \pm 0.43	408.6M
Retinanet FSAF [41]	0.916 \pm 0.017	0.861 \pm 0.032	9.58 \pm 0.43	441.5M

Table 8. Results on mixed resolution EPD-C subset. Train resolution, 1–2 m/pixel; test resolution, 1–2 m/pixel. AP refers to average precision. Recall, AP, and speed are expressed in the format of mean \pm standard deviation on the basis of 10 rounds of test results.

Detectors	Recall	AP	Speed (Images/s)
Faster R-CNN [19]	0.628 \pm 0.032	0.565 \pm 0.031	13.48 \pm 0.58
Cascade R-CNN [20]	0.649 \pm 0.016	0.590 \pm 0.010	10.96 \pm 0.48
Grid R-CNN [22]	0.658 \pm 0.027	0.600 \pm 0.022	9.63 \pm 0.41
Libra R-CNN [39]	0.661 \pm 0.012	0.583 \pm 0.010	13.49 \pm 0.73
Retinanet [29]	0.673 \pm 0.023	0.561 \pm 0.015	14.36 \pm 0.78
YOLOv3 [27]	0.693 \pm 0.017	0.550 \pm 0.025	10.24 \pm 0.39
YOLOv4 [28]	0.743 \pm 0.015	0.648 \pm 0.01	16.21 \pm 0.36
Retinanet FreeAnchor [40]	0.688 \pm 0.045	0.589 \pm 0.016	14.41 \pm 0.78
FCOS [32]	0.671 \pm 0.026	0.525 \pm 0.010	15.25 \pm 0.62
Retinanet FSAF [41]	0.653 \pm 0.026	0.582 \pm 0.013	16.42 \pm 0.50

For the detectors trained under the original resolution 1 m/pixel, we can see that Retinanet FSAF achieves the best recall and Retinanet FreeAnchor performs the best AP on the EPD-S subset. Overall, all detectors perform well in terms of accuracy. However, on the EPD-C subset, the results show that Libra R-CNN and Retinanet get the best recall and AP respectively. FCOS and Cascade R-CNN perform poorly on AP in the face of complex environments, while Faster R-CNN and Cascade R-CNN perform poorly on recall. It means that Libra R-CNN and Retinanet may perform better than other detectors in complex conditions.

For the detectors trained under the mixed resolution 1–2 m/pixel, it can be seen that YOLOv3 obtains the best recall and Retinanet FSAF gains the best AP. Moreover, when we test detectors on the EPD-C subset, YOLOv4 obtains both the best recall and best AP. In general, the AP of FCOS and the recall of Faster R-CNN on the mixed resolution dataset are not well.

Overall, the performance of detectors is satisfactory on the EPD-S subset, but the experimental results on the EPD-C subset are not ideal. It indicates that the detectors could be better trained to fully adapt to the complex environments. Contrasting the models trained under 1-m resolution and 1–2-m resolution, we can find that recalls and APs of various models decline slightly with acceptable drop degree. This suggests that the parameters utilized in the experiment can adapt to the change of resolution to a certain extent.

In terms of memory usage, YOLOv3 performs best with the least model size. This may be because YOLOv3 utilizes Darknet-53 as the backbone network, which contains fewer parameters than ResNet101 used in other detectors. YOLOv4 also has a small model size similar to YOLOv3, utilizing CSPDarknet-53 as the backbone network. FCOS has the smallest model size among the detectors taking ResNet101 as the backbone network. Cascade R-CNN and Grid R-CNN occupy a large amount of memory.

In terms of running speed, YOLOv3, YOLOv4, and FCOS perform well, while Grid R-CNN and Cascade R-CNN have relatively slow speed. This also indicates that higher speed is often accompanied by lower memory occupation. When the actual detection task requires a high speed and low memory detector, we recommend choosing YOLOv3, YOLOv4, or FCOS.

4.4. Robustness against Spatial Resolution

To test the performance of detectors under different spatial resolutions and evaluate the adaptability of detectors to resolution variation, we tested the detectors trained in Section 4.3 on 2- and 4-m resolution testing sets by down-sampling the EPD-C subset. We report the results in Tables 9–12.

Table 9. Results on the EPD-C subset. Train resolution, 1 m/pixel; test resolution, 2 m/pixel. AP refers to average precision. Recall, AP, and speed are expressed in the format of mean \pm standard deviation on the basis of 10 rounds of test results.

Detectors	Recall	AP	Speed (Images/s)
Faster R-CNN [19]	0.640 \pm 0.019	0.611 \pm 0.006	9.92 \pm 0.25
Cascade R-CNN [20]	0.645 \pm 0.024	0.582 \pm 0.010	8.59 \pm 0.17
Grid R-CNN [22]	0.706 \pm 0.029	0.626 \pm 0.033	7.52 \pm 0.34
Libra R-CNN [39]	0.714 \pm 0.016	0.642 \pm 0.017	9.45 \pm 0.22
Retinanet [29]	0.714 \pm 0.029	0.652 \pm 0.033	10.38 \pm 0.17
YOLOv3 [27]	0.687 \pm 0.011	0.591 \pm 0.015	10.15 \pm 0.26
YOLOv4 [28]	0.693 \pm 0.013	0.589 \pm 0.015	14.39 \pm 0.25
Retinanet FreeAnchor [40]	0.712 \pm 0.021	0.635 \pm 0.027	10.30 \pm 0.24
FCOS [32]	0.743 \pm 0.021	0.600 \pm 0.022	10.75 \pm 0.39
Retinanet FSAF [41]	0.722 \pm 0.033	0.632 \pm 0.035	11.10 \pm 0.29

Table 10. Results on the EPD-C subset. Train resolution, 1 m/pixel; test resolution, 4 m/pixel. AP refers to average precision. Recall, AP, and speed are expressed in the format of mean \pm standard deviation on the basis of 10 rounds of test results.

Detectors	Recall	AP	Speed (images/s)
Faster R-CNN [19]	0.365 \pm 0.013	0.343 \pm 0.009	9.89 \pm 0.16
Cascade R-CNN [20]	0.449 \pm 0.027	0.371 \pm 0.014	8.58 \pm 0.15
Grid R-CNN [22]	0.483 \pm 0.034	0.399 \pm 0.022	7.89 \pm 0.15
Libra R-CNN [39]	0.521 \pm 0.028	0.424 \pm 0.024	9.47 \pm 0.33
Retinanet [29]	0.470 \pm 0.022	0.395 \pm 0.010	10.65 \pm 0.10
YOLOv3 [27]	0.404 \pm 0.031	0.240 \pm 0.038	9.51 \pm 0.43
YOLOv4 [28]	0.422 \pm 0.018	0.308 \pm 0.021	17.45 \pm 0.91
Retinanet FreeAnchor [40]	0.486 \pm 0.030	0.388 \pm 0.023	10.52 \pm 0.23
FCOS [32]	0.603 \pm 0.023	0.449 \pm 0.026	11.13 \pm 0.25
Retinanet FSAF [41]	0.487 \pm 0.060	0.394 \pm 0.026	11.26 \pm 0.17

Table 11. Results on the EPD-C subset. Train resolution, 1–2 m/pixel; test resolution, 2 m/pixel. AP refers to average precision. Recall, AP, and speed are expressed in the format of mean \pm standard deviation on the basis of 10 rounds of test results.

Detectors	Recall	AP	Speed (Images/s)
Faster R-CNN [19]	0.593 \pm 0.035	0.537 \pm 0.022	14.42 \pm 0.36
Cascade R-CNN [20]	0.618 \pm 0.016	0.558 \pm 0.021	11.89 \pm 0.28
Grid R-CNN [22]	0.630 \pm 0.035	0.562 \pm 0.031	10.43 \pm 0.27
Libra R-CNN [39]	0.633 \pm 0.018	0.550 \pm 0.018	14.31 \pm 0.27
Retinanet [29]	0.629 \pm 0.022	0.520 \pm 0.023	15.70 \pm 0.24
YOLOv3 [27]	0.663 \pm 0.024	0.525 \pm 0.027	11.82 \pm 0.66
YOLOv4 [28]	0.688 \pm 0.014	0.611 \pm 0.019	23.34 \pm 0.97
Retinanet FreeAnchor [40]	0.632 \pm 0.038	0.549 \pm 0.021	15.68 \pm 0.44
FCOS [32]	0.659 \pm 0.030	0.495 \pm 0.020	15.49 \pm 0.35
Retinanet FSAF [41]	0.591 \pm 0.028	0.542 \pm 0.027	17.40 \pm 0.41

Table 12. Results on the EPD-C subset. Train resolution, 1–2 m/pixel; test resolution, 4 m/pixel. AP refers to average precision. Recall, AP, and speed are expressed in the format of mean \pm standard deviation on the basis of 10 rounds of test results.

Detectors	Recall	AP	Speed (Images/s)
Faster R-CNN [19]	0.489 \pm 0.044	0.427 \pm 0.021	14.47 \pm 0.21
Cascade R-CNN [20]	0.471 \pm 0.034	0.402 \pm 0.024	11.61 \pm 0.31
Grid R-CNN [22]	0.508 \pm 0.024	0.445 \pm 0.019	10.38 \pm 0.37
Libra R-CNN [39]	0.572 \pm 0.029	0.472 \pm 0.020	14.08 \pm 0.51
Retinanet [29]	0.470 \pm 0.040	0.378 \pm 0.027	15.67 \pm 0.46
YOLOv3 [27]	0.590 \pm 0.015	0.402 \pm 0.012	10.46 \pm 0.27
YOLOv4 [28]	0.473 \pm 0.029	0.366 \pm 0.027	27.31 \pm 1.16
Retinanet FreeAnchor [40]	0.468 \pm 0.064	0.371 \pm 0.024	15.58 \pm 0.34
FCOS [32]	0.585 \pm 0.024	0.416 \pm 0.016	15.56 \pm 0.45
Retinanet FSAF [41]	0.460 \pm 0.037	0.410 \pm 0.038	17.62 \pm 0.64

In Table 9, FCOS obtains the best recall, and Retinanet gains the best AP. In Table 10, FCOS performs the best performance in terms of both recall and AP, while YOLOv3 performs very poorly in terms of AP. In Table 11, YOLOv4 obtain the best recall and AP. In Table 12, YOLOv3 gets the best recall, while Libra R-CNN gets the best AP and also performs quite well in terms of recall. For running speed, one-stage detectors are faster than two-stage detectors in general. Among all 10 detectors compared, YOLOv4 can always achieve the fastest running speed; FCOS and Retinanet FSAF can also get a satisfactory running speed. However, Grid R-CNN always runs at a slow speed.

Overall, the detectors have resolution adaptability to some extent. The detectors trained on the 1-m resolution data perform better results on the 2-m resolution test set, while the detectors trained on the mixed resolution data get better results on the 4-m resolution test set. As we select AP as the index to evaluate the adaptability of resolution variation, FCOS trained on the 1-m resolution data and Libra R-CNN trained on the mixed resolution data show better resolution robustness.

Figures 15 and 16 visualize the detection results of each detectors on the EPD-C subset. Four result images are shown for each detector in its row. The first two images are from the same scene with resolutions of 1 and 4 m, respectively, and the background is mountainous, among which there is an island similar to the characteristics of electric pylons in the upper right corner. The background of the other scene with 1- and 4-m resolution images on the right is a power plant which contains relatively dense targets of electric pylons, and the frame structure of the plant itself also has an impact on the detection results.

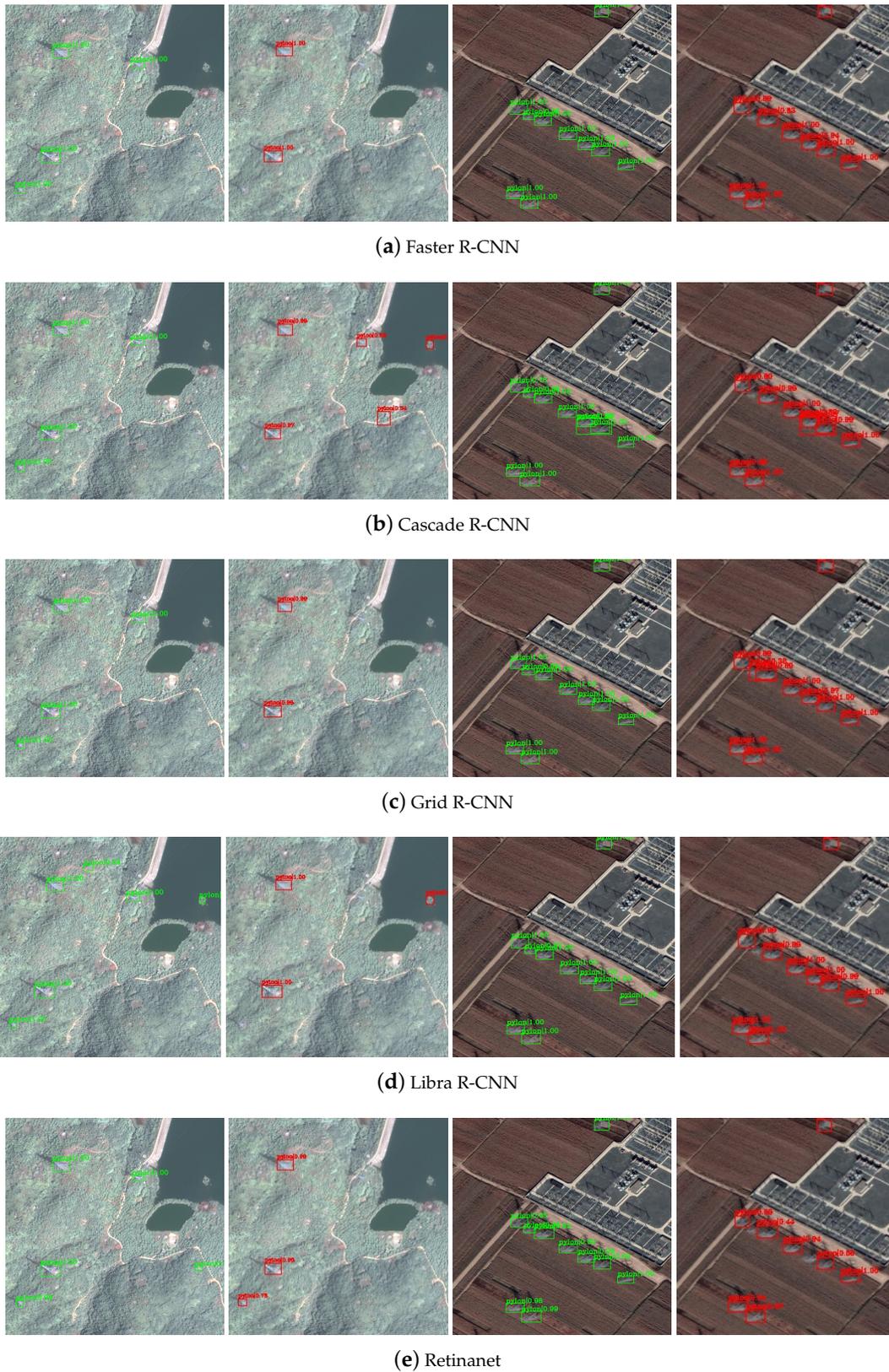


Figure 15. Visualization of experimental results on the EPD-C subset. We utilized detectors obtained from Section 4.3 to test two scenes in the EPD-C subset. For each scene, the left image is in 1-m resolution, while the right one is in 4-m resolution. Rows (a–e) show detection results of Faster R-CNN, Cascade R-CNN, Grid R-CNN, Libra R-CNN, and Retinanet, respectively. We use the green detection boxes in the 1-m images and the red detection boxes in the 4-m images.

Overall, detection results under 1-m resolution is superior to those under 4-m resolution, which shows that all detectors possess resolution adaptability to some extent. Besides, detection results of the scene on the left are more affected by resolution decrease. The reason may be that the background of the left scene is more similar to electric pylons. Furthermore, as shown in Figure 15d, the results of Libra R-CNN on the left scene is not ideal, and the detector even identifies the island as an electric pylon target at 1-m resolution. It is because Libra R-CNN uses the Feature Balance Pyramid to enhance the representation of areas similar to the characteristics of the electric pylons targets, thereby misidentifying the island as the electric pylon. As shown in Figure 16d, the results of FCOS on the right scene is not acceptable. It misses several electric pylon targets. This means that anchor-free detectors may not perform well when detecting densely distributed targets.

5. Discussion

5.1. Analysis of Performance

The analysis of advantages and disadvantages of each detectors based on deep learning can help to select the most suitable detectors to fulfill the electric pylon detection task. As shown in Section 4, Faster R-CNN [19] gets ordinary and acceptable performance in terms of recall, AP, running speed, and model size. On the one hand, Faster R-CNN uses FPN to adapt to the scale change of targets. On the other hand, Faster R-CNN uses RPN to perform better in binary classification. As Faster R-CNN does not have much improvement compared with the conventional two-stage detectors, it performs normally in the actual detection task. Specially, we regard Faster R-CNN as a benchmark for comparison. Cascade R-CNN [20] has the biggest model size and a slower running speed. When testing Cascade R-CNN on a test set with lower resolution, it performs better than Faster R-CNN in general, according to the IoU threshold mentioned in Section 3.2.3.

Grid R-CNN [22] obtains a superior AP and recall, but its model size is big and its running speed is the slowest. Due to its grid point positioning mechanism, Grid R-CNN achieves a good detection result on square targets. Considering that electric pylon targets are similar to square targets, Grid R-CNN performs well in electric pylon detection. However, the feature map size of Grid R-CNN after ROI Align is 14×14 , which is larger than the commonly used 7×7 size. Thus, Grid R-CNN has a slow running speed and occupies a large amount of memory.

Libra R-CNN [39] has a similar model size with Faster R-CNN, but it performs better than Faster R-CNN in other aspects. Libra R-CNN usually gets a relatively faster running speed with a mediocre recall and AP compared to other detectors. Since the method of Libra R-CNN dividing the training process into three stages is not very complex, its detection speed and model size are not much different from Faster R-CNN. Besides, Libra R-CNN uses the Feature Balance Pyramid to enhance the representation of areas similar to the characteristics of targets to be detected.

Retinanet [29] has a similar AP and running speed as Faster R-CNN, but its recall is better and its model size is smaller. Retinanet is a one-stage detector which does not use RPN, so it has a simple structure and small model size. In addition, Retinanet uses focal loss to improve the detection accuracy.

YOLOv3 [27] achieves the smallest model size and a superior resolution applicability contrasting with other detectors, except that the detector trained on 1-m resolution data obtains a low AP on 4-m resolution test set. YOLOv4 [28] also has a small size, performs fast in speed, and has good adaptability to a certain degree of resolution degradation. Both Darknet-53 and CSPDarknet-53 contain fewer layers than Resnet101, thus both YOLOv3 and YOLOv4 have a small model size and perform well in speed. Compared with YOLOv3, YOLOv4 offers better overall detection performance and integrates several typical modules commonly used in deep-learning-based detectors.

Retinanet FreeAnchor [40] and Retinanet FSAF [41] perform well on the EPD-S subset, achieving satisfactory recalls and APs, but they cannot adapt resolution variation favorably. These two detectors both make improvements on the basis of Retinanet, and experimental results show that they both have better performance than Retinanet. The former represents object–anchor matching as a

maximum likelihood estimation (MLE) process and selects the most representative anchor from each object's anchor set, while the latter focuses on how to select the optimal feature layer for object–anchor matching. Results of these two detectors indicate that object–anchor matching is an important part of the electric pylon detector.

FCOS [32] gets a superior performance in terms of running speed, and in most cases its recall is satisfactory, but its AP is not good. FCOS is an anchor-free detector. It has a fast detection speed and performs well in the detection task of small targets and low resolution.

In general, we could not find a detector that has good performance on all aspects. Thus, we need to choose the most suitable one based on the requirements and restrictions in practical situation. FCOS and YOLOv4 could be used to get rapid results, while YOLOv3 and YOLOv4 could be used when the space occupancy is limited. Retinanet FreeAnchor, Retinanet FSAF, and YOLOv4 could be used in conventional environment. YOLOv4 and Retinanet could be used in complex environment when high precision is needed, while Libra R-CNN and YOLOv4 could be used in complex environment to get high recall. If the spatial resolution does not change much, Grid R-CNN, Libra R-CNN, YOLOv4, and Retinanet could be used to gain superior AP; otherwise, we recommend using FCOS as the electric pylon detector.

5.2. Analysis of Resolution Robustness

To discuss resolution robustness, we calculated the average recall and AP of the detectors in each case. The results are shown in Table 13.

Table 13. Average results calculated from Section 4. Recall and AP are obtained from the average calculation of the experimental results of all models under the corresponding training and test resolution.

Train Resolution	Test Resolution	Recall	AP
1 m/pixel	1 m/pixel	0.743	0.666
1 m/pixel	2m/pixel	0.698	0.616
1 m/pixel	4m/pixel	0.469	0.376
1–2 m/pixel	1–2 m/pixel	0.672	0.579
1–2 m/pixel	2m/pixel	0.634	0.545
1–2 m/pixel	4m/pixel	0.509	0.409

As shown in Table 13, if the resolution variation is small, e.g., from 1 to 2 m, the detectors could still gain acceptable accuracy. However, when the resolution varies over a certain limit, e.g., larger than 1 m, the performance of detectors declines rapidly. The detectors trained on mixed resolution data perform relatively better on 4-m test set. It should be noticed that we did not fine-tune the parameters when training variant-resolution, and we can improve the accuracy of detectors by adjusting the parameters such as scales of anchors. That is mainly because detectors trained on variant-resolution data learned further multi-scale features in multi-scale images.

The detectors trained by fixed 1-m resolution or variant-resolution perform barely satisfactory on 4-m test set. On the one hand, there are quite a few objects with the size of less than 20 pixels, and these small objects can be hardly observed even by eyes. On the other hand, the improvement of network structure can hardly completely solve multi-scale problems. Therefore, we need to design new networks to adapt flexible resolution better.

5.3. Application Prospects

With the development of satellite-based and airborne Earth observation, high-resolution remote sensing data can be obtained more and more easily. It is now possible to obtain high-resolution remote sensing data of the regions of interest at low cost and high frequency. Thus, given high-resolution remote sensing data of electric pylons, our deep-learning-based detectors can automatically detect

electric pylons. Considering the good generalization ability of deep-learning-based detectors, our work has significant potential for electric pylon detection, benefiting the management of electric power system.

6. Conclusions

In this paper, we introduce deep learning methods to achieve electric pylon detection in high-resolution remote sensing images. To analyze the comprehensive performance of different detectors under different conditions, we selected 10 state-of-the-art deep-learning-based detectors, and comprehensively compared their performance on a specially made dataset containing 1500 images. The experimental results show the characteristics of each detector in detail and provide the selection criteria when deep-learning-based detectors are applied to actual scene. For conventional detection tasks, YOLOv4 and Retinanet FSAF can achieve relatively good detection accuracy, while the running speed of FCOS and YOLOv4 is fast and can adapt to tasks with real-time processing requirements. In addition, YOLOv3 and YOLOv4 are small in model size and can adapt to the working environment with small memory. For complex detection tasks, Grid R-CNN, YOLOv4, and Retinanet have better comprehensive performance. From the perspective of practical application, Grid R-CNN, Libra R-CNN, YOLOv4, and Retinanet have excellent detection accuracy in tasks with low resolution, and FCOS shows better comprehensive performance in tasks with mixed resolution. It should also be noticed that a detector that performs the best in all conditions has not appeared thus far. Therefore, we need further research on deep-learning-based detectors for electric pylon detection in high-resolution remote sensing images in the future.

Author Contributions: Conceptualization, H.Z.; methodology, S.Q., Y.S., and H.Z.; software, S.Q. and Y.S.; validation, S.Q., Y.S., and H.Z.; formal analysis, S.Q., Y.S., and H.Z.; investigation, S.Q. and Y.S.; resources, H.Z.; data curation, S.Q., Y.S., and H.Z.; writing—original draft preparation, S.Q. and Y.S.; writing—review and editing, H.Z.; visualization, S.Q. and Y.S.; supervision, H.Z.; project administration, H.Z.; and funding acquisition, H.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the National Key Research and Development Program of China (Grant Nos. 2016YFB0501300, 2016YFB0501302, and 2019YFC1510905), the National Natural Science Foundation of China (Grant No. 61501009), and the Fundamental Research Funds for the Central Universities.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Albert, R.; Albert, I.; Nakarado, G.L. Structural vulnerability of the North American power grid. *Phys. Rev. E* **2004**, *69*, 025103. [[CrossRef](#)] [[PubMed](#)]
2. Araar, O.; Aouf, N.; Dietz, J.L.V. Power pylon detection and monocular depth estimation from inspection UAVs. *Ind. Robot Int. J.* **2015**, *42*, 200–213. [[CrossRef](#)]
3. Zhang, R.; Yang, B.; Xiao, W.; Liang, F.; Liu, Y.; Wang, Z. Automatic Extraction of High-Voltage Power Transmission Objects from UAV Lidar Point Clouds. *Remote Sens.* **2019**, *11*, 2600. [[CrossRef](#)]
4. Chuvieco, E. *Earth Observation of Global Change: The Role of Satellite Remote Sensing in Monitoring the Global Environment*; Springer: Alcalá de Henares, Spain, 2008.
5. Zhang, F.; Du, B.; Zhang, L.; Xu, M. Weakly supervised learning based on coupled convolutional neural networks for aircraft detection. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 5553–5563. [[CrossRef](#)]
6. Cai, B.; Jiang, Z.; Zhang, H.; Yao, Y.; Nie, S. Online exemplar-based fully convolutional network for aircraft detection in remote sensing images. *IEEE Geosci. Remote Sens. Lett.* **2018**, *15*, 1095–1099. [[CrossRef](#)]
7. Zou, Z.; Shi, Z. Ship detection in spaceborne optical image with SVD networks. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 5832–5845. [[CrossRef](#)]
8. Yao, Y.; Jiang, Z.; Zhang, H.; Zhao, D.; Cai, B. Ship detection in optical remote sensing images based on deep convolutional neural networks. *J. Appl. Remote Sens.* **2017**, *11*, 042611. [[CrossRef](#)]

9. Yao, Y.; Jiang, Z.; Zhang, H.; Cai, B.; Meng, G.; Zuo, D. Chimney and condensing tower detection based on faster R-CNN in high resolution remote sensing images. In Proceedings of the 2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Fort Worth, TX, USA, 23–28 July 2017; pp. 3329–3332.
10. Zhang, H.; Deng, Q. Deep learning based fossil-fuel power plant monitoring in high resolution remote sensing images: A comparative study. *Remote Sens.* **2019**, *11*, 1117. [[CrossRef](#)]
11. Wu, Y.; Ma, W.; Gong, M.; Bai, Z.; Zhao, W.; Guo, Q.; Chen, X.; Miao, Q. A Coarse-to-Fine Network for Ship Detection in Optical Remote Sensing Images. *Remote Sens.* **2020**, *12*, 246. [[CrossRef](#)]
12. Zhou, M.; Jing, M.; Liu, D.; Xia, Z.; Zou, Z.; Shi, Z. Multi-resolution networks for ship detection in infrared remote sensing images. *Infrared Phys. Technol.* **2018**, *92*, 183–189. [[CrossRef](#)]
13. Nahhas, F.H.; Shafri, H.Z.; Sameen, M.I.; Pradhan, B.; Mansor, S. Deep learning approach for building detection using lidar–orthophoto fusion. *J. Sens.* **2018**, *2018*, 1–12. [[CrossRef](#)]
14. Jiao, J.; Zhang, Y.; Sun, H.; Yang, X.; Gao, X.; Hong, W.; Fu, K.; Sun, X. A densely connected end-to-end neural network for multiscale and multiscene SAR ship detection. *IEEE Access* **2018**, *6*, 20881–20892. [[CrossRef](#)]
15. Sommer, L.W.; Schuchert, T.; Beyerer, J. Fast deep vehicle detection in aerial images. In Proceedings of the 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), Santa Rosa, CA, USA, 24–31 March 2017; pp. 311–319.
16. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
17. He, K.; Zhang, X.; Ren, S.; Sun, J. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 1904–1916. [[CrossRef](#)]
18. Girshick, R. Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 11–18 December 2015; pp. 1440–1448.
19. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*; NIPS: Montreal, QC, Canada, 2015; pp. 91–99.
20. Cai, Z.; Vasconcelos, N. Cascade r-cnn: Delving into high quality object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 6154–6162.
21. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2961–2969.
22. Lu, X.; Li, B.; Yue, Y.; Li, Q.; Yan, J. Grid r-cnn. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 7363–7372.
23. Chen, K.; Pang, J.; Wang, J.; Xiong, Y.; Li, X.; Sun, S.; Feng, W.; Liu, Z.; Shi, J.; Ouyang, W.; et al. Hybrid task cascade for instance segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 4974–4983.
24. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. Ssd: Single shot multibox detector. In *European Conference on Computer Vision*; Springer: Amsterdam, The Netherlands, 2016; pp. 21–37.
25. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
26. Redmon, J.; Farhadi, A. YOLO9000: Better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 7263–7271.
27. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
28. Bochkovskiy, A.; Wang, C.Y.; Liao, H.Y.M. YOLOv4: Optimal Speed and Accuracy of Object Detection. *arXiv* **2020**, arXiv:2004.10934.
29. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal loss for dense object detection. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2980–2988.
30. Law, H.; Deng, J. Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European Conference on Computer Vision (ECCV)*; Springer: Munich, Germany, 2018; pp. 734–750.

31. Zhou, X.; Wang, D.; Krähenbühl, P. Objects as points. *arXiv* **2019**, arXiv:1904.07850.
32. Tian, Z.; Shen, C.; Chen, H.; He, T. Fcos: Fully convolutional one-stage object detection. In Proceedings of the IEEE International Conference on Computer Vision, Seoul, Korea, 27 October–2 November 2019; pp. 9627–9636.
33. Matikainen, L.; Lehtomäki, M.; Ahokas, E.; Hyypä, J.; Karjalainen, M.; Jaakkola, A.; Kukko, A.; Heinonen, T. Remote sensing methods for power line corridor surveys. *ISPRS J. Photogramm. Remote Sens.* **2016**, *119*, 10–31. [[CrossRef](#)]
34. Yermo, M.; Martínez, J.; Lorenzo, O.G.; Vilarino, D.L.; Cabaleiro, J.C.; Pena, T.F.; Rivera, F.F. Automatic detection and characterisation of power lines and their surroundings using lidar data. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2019**, *455*, 1161–1168. [[CrossRef](#)]
35. Tilawat, J.; Theera-Umpon, N.; Auephanwiriyakul, S. Automatic detection of electricity pylons in aerial video sequences. In Proceedings of the 2010 International Conference on Electronics and Information Engineering, Kyoto, Japan, 1–3 August 2010; Voume 1, pp. V1–342.
36. Sampredo, C.; Martinez, C.; Chauhan, A.; Campoy, P. A supervised approach to electric tower detection and classification for power line inspection. In Proceedings of the 2014 International Joint Conference on Neural Networks (IJCNN), Beijing, China, 6–11 July 2014; pp. 1970–1977.
37. Fei, X.; Tan, Y. Electric Tower Target Identification Based on High-resolution SAR Image and Deep Learning. In *Journal of Physics: Conference Series*; IOP: Xi'an, China, 2020; Volume 1453, p. 012117.
38. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
39. Pang, J.; Chen, K.; Shi, J.; Feng, H.; Ouyang, W.; Lin, D. Libra r-cnn: Towards balanced learning for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 821–830.
40. Zhang, X.; Wan, F.; Liu, C.; Ji, R.; Ye, Q. FreeAnchor: Learning to Match Anchors for Visual Object Detection. In *Advances in Neural Information Processing Systems*; NIPS: Montreal, QC, Canada, 2019; pp. 147–155.
41. Zhu, C.; He, Y.; Savvides, M. Feature selective anchor-free module for single-shot object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 840–849.
42. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 27–30 June 2016; pp. 770–778.
43. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125.
44. Wang, C.Y.; Liao, H.Y.M.; Yeh, I.H.; Wu, Y.H.; Chen, P.Y.; Hsieh, J.W. CSPNet: A New Backbone that can Enhance Learning Capability of CNN. *arXiv* **2019**, arXiv:1911.11929.
45. Neubeck, A.; Van Gool, L. Efficient non-maximum suppression. In Proceedings of the 18th International Conference on Pattern Recognition (ICPR'06), Hong Kong, China, 20–24 August 2006; Volume 3, pp. 850–855.
46. Lu, X.; Li, B.; Yue, Y.; Li, Q.; Yan, J. Grid R-CNN Plus: Faster and Better. *arXiv* **2019**, arXiv:1906.05688.
47. Wu, Y.; He, K. Group normalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*; Springer: Munich, Germany, 2018; pp. 3–19.
48. Wang, X.; Girshick, R.; Gupta, A.; He, K. Non-local neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 7794–7803.
49. Nair, V.; Hinton, G.E. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*; Omnipress: Haifa, Israel, 2010; pp. 807–814.
50. White, H. Maximum likelihood estimation of misspecified models. *Econom. J. Econom. Soc.* **1982**, *50*, 1–25. [[CrossRef](#)]
51. Zhou, D.; Fang, J.; Song, X.; Guan, C.; Yin, J.; Dai, Y.; Yang, R. Iou loss for 2D/3D object detection. In Proceedings of the 2019 International Conference on 3D Vision (3DV), Québec City, QC, Canada, 16–19 September 2019; pp. 85–94.

52. Bottou, L. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*; Springer: Berlin, Germany, 2010; pp. 177–186.
53. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.



© 2020 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 (<http://creativecommons.org/licenses/by/4.0/>).