Vehicle detection in aerial images based on lightweight deep convolutional network

Jiaquan Shen\textsuperscript{1,2} | Ningzhong Liu\textsuperscript{1,2} | Han Sun\textsuperscript{1,2}

\textsuperscript{1}College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China
\textsuperscript{2}MIIT Key Laboratory of Pattern Analysis and Machine Intelligence, Nanjing, China

Correspondence
Ningzhong Liu, College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China.
Email: lnz_nuaa@163.com

Abstract

Vehicle detection in aerial images is an interesting and challenging task. Traditional methods are based on sliding-window search and handcrafted features, which limits the representation power and has heavy computational costs. Recent research has shown that deep-learning algorithms are widely used in the field of object detection. However, the deep-learning algorithms still face many difficulties and challenges in the object detection under the aerial scene. Meanwhile, the high computational cost of detection algorithms lead to low-detection efficiency. In this study, we build a fast and accurate lightweight detection framework for vehicle detection in aerial scenes. The proposed detection method improves the expressive ability of detection network and significantly reduces the amount of calculations in the model. Meanwhile, setting suitable anchor boxes according to the size of the object vehicles have been introduced in our model, which also effectively improves the performance of the detection. In addition, we have published a new aerial vehicle image dataset and verified the effectiveness of our method. In the Munich dataset and our dataset, our method achieves 85.8\% and 91.2\% of the mean average precision (mAP), and its detection time is 1.78 and 0.048 s on Nvidia Titan XP. Our results show that the proposed framework achieves significant improvement over several alternatives and state-of-the-art schemes with higher accuracy and less detection time.

1 INTRODUCTION

Vehicle object detection in aerial photography scenarios plays an important role in both military and civil fields. This technology can be effectively applied to traffic management and military target strikes. In the civil field, vehicle recognition is one of the important directions of object detection research in aerial imagery scenes, and it is an important part of the intelligent transportation system. It provides technical support for real-time road conditions acquisition, accident monitoring, illegal parking monitoring and other application scenarios. In the military field, object detection of aerial vehicles can obtain a wealth of battlefield information and can accurately strike specific targets. Object detection in aerial photography requires real-time detection, which has high requirements for detection speed and accuracy. On the unmanned aerial vehicle (UAV) embedded platform, the detection model needs to be able to be separated from the graphics processing unit (GPU) device for real-time detection. This requires reducing the size and calculation of the model to meet the requirements of real-time detection.

Recently, vehicle detection based on aerial image has received extensive attention and achieved a series of good detection results [1–6]. However, these detection methods still have obvious shortcomings. First, the size of aerial images is generally very large while the vehicle is very small, and it is really difficult to detect several small vehicle objects in a large range. Second, the vehicles have varieties of styles and colours, and they are often accompanied with a complex background information such as occlusion, shadows.

In previous studies [7–8], various algorithms for vehicle detection in aerial images have been proposed, which have improved the robustness and effectiveness in the past period. The most common algorithms are based on the sliding-window method and apply the filter to all possible locations and scales in the image. Although these traditional target detection algorithms have achieved some good detection results, their
disadvantages are also obvious. First, the hand-crafted features and shallow learning–based features restricted the ability to extract and represent features. Second, lots of redundant computations were produced in the sliding-window method, which will significantly increase the computational burden.

Deep convolutional neural networks (CNN) are widely used in semantic segmentation [9–10] and object detection [11–15]. In the field of object detection, the method based on convolutional neural network has achieved exciting results in natural scene images. There are some typical models working very well in object detection such as R-CNN [11], Fast R-CNN [12], Faster R-CNN [13], you only look once (YOLO) [14–15], single shot detector (SSD) [18] and so on. Running high-quality CNN model with strict constraints on memory and computational budget is becoming a rising interest in computer vision. In recent years, many innovative networks have been proposed such as PAVNet [21], MobileNets [22], ShuffelNet [23], NASNet [24]. However, these architectures heavily rely on deep separable convolution operations, which lacks efficient implementation. At the same time, there are few researches on the combination of efficient algorithms with fast object detection model. The SSD [18] is one of the first attempts at using a pyramidal feature hierarchy. Ideally, the SSD-style pyramid would reuse the multiscale feature maps from different layers computed in the forward pass and thus come free of cost. But to avoid using low-level features, SSD foregoes reusing already computed layers and instead builds the pyramid starting from high up in the network and then by adding several new layers. Thus, it misses the opportunity to reuse the higher-resolution maps of the feature hierarchy.

In this study, we propose a fast and accurate vehicle detection framework in large-scale aerial image (see Figure 1). Our method combines the lightweight feature extraction network with the object detection algorithm Faster R-CNN, which is more suitable for the detection of small objects in aerial scenes. Moreover, we set an appropriate scale and ratio of the anchor box according to the size of object vehicle in aerial image, which will further improve the effectiveness of the detection. We evaluate and validate the effectiveness of the proposed method in Munich dataset [2] and our collected dataset [45]. The results show that our method achieves the most advanced level in terms of detection accuracy and time consumption. At the same time, the limited annotation data would easily lead to over-fitting of the model, and the large-scale aerial images would increase the detection time. In order to overcome these problems, we segment the aerial image into blocks, and increase the amount of data through rotation, scale and Gaussian noise.

The main contributions of our work are presented as follows:

1. A fast and effective lightweight vehicle detection model in aerial scenes is established. In this detection framework, a series of methods for small object detection such as transition layers with same channels, feature extraction under small convolution kernel and three-way dense layer for fusing different feature maps have been implemented in our proposed network, which improves the expressive ability of detection network and significantly reduces the amount of calculations in the model.

2. We combine the proposed lightweight feature extraction model with the object detection algorithm Faster R-CNN and set an appropriate size anchor box according to the size of the vehicle object, so that it can better detect small objects in aerial photography scenes.

3. Extensive ablation studies validate the effectiveness and efficiency of the proposed approach. Moreover, we have
published our aerial vehicle image dataset with the ground truth and verified the effectiveness of our method on this dataset, which further validates the effectiveness and robustness of our model.

2 RELATED WORK

In this section, we briefly introduce the recent methodologies related to object detection including the method of traditional handcrafted features and deep CNN. In addition, some recent vehicle detection models are introduced as well.

2.1 Traditional handcrafted features for object detection

In feature extraction, Haar wavelet, local binary pattern (LBP), scale invariant feature transform (SIFT) and histogram of oriented gradient (HOG) are typical handcrafted features for object detection. Viola et al. [26] proposed an integral image method of Haar wavelet feature to increase the speed of object detection. This research is successfully applied to face detection [27]. The LBP is used to extract the texture features of the image [28], which has characteristics of rotation and grey invariances. SIFT is a local feature description operator [29], which is robust to the transformation of light, noise as well as the angle of view in the image.

The HOG gradient histogram algorithm uses a sliding window to filter all possible positions and scales in the image [30]. Additionally, the deformable parts model (DPM) algorithm calculates the gradient histogram of HOG, and then uses support vector machine (SVM) to perform the matching and classification of the object [31]. As an important object detection algorithm, DPM has advantages in posture analysis and object location.

However, these feature extraction methods are manually designed, and based on the understanding of the specific task by the designers, which usually have less actual parameters and unstable performance. Although traditional object detection methods show good performance in the specific detection task, its disadvantages are also obvious, therefore, it still cannot meet the requirements of large-scale data at the current stage.

2.2 Object detection with the deep CNNs model

There are some typical CNN detection models such as regions convolutional neural networks (R-CNN) [11], Fast R-CNN [12], Faster R-CNN [13], YOLO [14], YOLOv2 [15], SSD [18], feature pyramid networks (FPN) [19], and RefineDet [20]. The R-CNN uses a selective search algorithm to generate object-like regions and extracts deep features on classification with an SVM classifier [11]. Although this algorithm has improved the efficiency of detection, a large number of repetitive operations will restrict the performance of the model. Compared with R-CNN, Fast R-CNN has some greater advantages in detection performance [12]. It designs a pooling layer structure of ROI pooling, which solves a lot of repetitive operations in R-CNN and greatly improves the performance of the algorithm. However, Fast R-CNN requires selective search to generate positive and negative samples, which limits the efficiency of the detection. The Faster R-CNN employs a region proposal network (RPN) auxiliary network to determine whether there is an object in the candidate box, and then it performs classification detection through classified positioning [13]. In Faster R-CNN, the CNN of the whole process can share the extracted feature information, which greatly improves the computational efficiency. YOLO [14] algorithm is based on the global information of the image, and it directly predicts the coordinate information and confidence of the category in each grid, which has improved the speed of detection. On the basis of YOLO, YOLOv2 [15] introduces batch normalisation, high resolution classifier and convolution with anchor boxes, which makes the detection model much better and stronger. SSD [18] algorithm helps to predict object regions on feature maps of different convolution layers which output discrete multiscale default coordinates.

Although these detection models based on CNNs have achieved high accuracy, the deep convolutional networks with enormous computational overhead leads to longer detection time. Recently, a series of lightweight networks have been developed, including PVANET [21], MobileNet [22], ShuffleNet [23], PeleeNet [25], GoogleNet [32], Xception [33], ResNeXt [34], and SqueezeNet [35], which have less complex structures and provide comparable or even better results compared with other CNNs. However, in the object detection application, there are few studies combining the backbone network with detection algorithm.

PAVNET [21] is mainly aimed at improving the speed of the detection model. In PAVNET, the authors propose a lightweight network, which is based on the principle of more layers with less channels. In addition, the authors use concatenated rectified linear units (CReLU) and Inception structure to reduce network redundancy. In PeleeNet [25], the authors combine the parallel multi-size kernel convolutions together into a two-way dense layer to the squeeze model, and adopt a residual block to improve the accuracy after feature extraction stage. Compared with MobileNet and ShuffleNet on the Pascal VOC dataset, the PeleeNet has more accurate results and smaller model size. However, since the SSD algorithm uses multiscale hierarchical feature maps for object detection, it tends to be not suitable for small-target detection and the model also generates a large number of redundant calculations. Therefore, the detection structure of the backbone PeleeNet with SSD detection algorithm is not suitable for vehicle detection under large-scale aerial scenes.

2.3 Vehicle detection with the deep CNNs model

The detection model with deep CNN has great advantages in the field of object detection, and it has become an important detection method in this research.
Yuan et al. [36] present a sequential vehicle detection model to deal with serious occlusions in natural scenes. The algorithm can effectively detect and track vehicles in complex interaction scenarios. Wang et al. [37] performs traffic congestion detection by detecting vehicles on the road. In this algorithm, the author uses the low-level texture information and kernel regression to detect the degree of traffic congestion.

Nassim et al. [38] propose to segment the aerial image into similar regions on the basis of the CNN, and then they use a trained SVM classifier to locate and classify the object after determine the candidate region of the vehicle. Cheng et al. [39] propose rotation invariant convolutional neural networks (RICNN) algorithm for target detection in aerial image. Although these measures mentioned above further improves the detection performance, the algorithm still significantly increases the overhead of the network. Tuermer et al. [43] suggest an innovative processing chain for vehicle detection that is able to exclude large homogeneous regions of multi-channel images without exclusion of useful information, whose results show a high-detection rate and good reliability. Audebert et al. [44] present a deep-learning–based segment-before-detect method for segmentation and detection of several varieties of vehicles in remote sensing images, and this research shows that deep-learning is also suitable for object-oriented analysis of aerial images because object detection can be obtained as a byproduct of accurate semantic segmentation.

Recently, some related studies employ Faster R-CNN for vehicle detection in aerial imagery. In [3], the authors present a CNN-based detection model with two convolutional networks, where the first network is applied to generate vehicle-like regions from multi-feature maps. Then, the generated candidate regions are fed into the second network for feature extraction and object detection. In [4], the authors propose a method of vehicle detection in aerial images by adding negative sample marks and establishing an hyper region proposal network (HRPN) network to improve the detection accuracy. However, the aerial image has its own characteristics such as large scale of image, small object for detection, and complicated background information, which usually brings many challenges to the detection work. Therefore, the deep detection model mentioned before cannot be directly used for vehicle detection under aerial scenes, and it is necessary to make targeted improvements according to the characteristics of aerial images.

In this study, we have established a feature extraction network for small targets in aerial scenes, which can better extract the features of small targets. Meanwhile, we set the anchor box of the appropriate size according to the target size in the object detection algorithm Faster R-CNN. Additionally, we combine the feature extraction network with the improved object detection algorithm Faster R-CNN, which has a significant improvement to the accuracy and speed of detection.

3 | MATERIAL AND METHODS

The framework of our proposed vehicle detection method is illustrated in Figure 2. Methods based on the deep neural network have high requirements on GPU memory during image processing, especially for processing large-size images. Meanwhile, the number of aerial image datasets is limited, which easily leads to over-fitting. Considering these difficulties and challenges, we preprocess and augment the dataset according to [4] and [5]. In the training phase, we crop the original large-size images into blocks, and increase the amount of data through rotation, scale and Gaussian noise. In the testing stage, the test results are obtained based on the trained model, and the detection results of image blocks are stitched together to recombine the original image.

3.1 | The blocks in the proposed network

Our proposed detection network is composed of the stem and stage blocks. The highlights of this network are adopting
transition feature maps with equal channel width, small convolution kernel for feature extraction of small objects and three-way layer to fuse different feature maps.

3.1.1 Stem block

Motivated by Inception-v4 [41] and PeleeNet [25], we design a cost-efficient stem block before the first layer. In the stem block, one branch is stacked with a $1 \times 1$ and a $3 \times 3$ convolution, and the other branch is a $2 \times 2$ max pooling layer (see Figure 3(a)). We find that adding this simple stem structure can evidently improve the detection performance in our experiments.

At present, the mainstream lightweight networks usually adopt depthwise separable convolutions, in which pointwise convolutions account for a large part of the computation. However, the memory consumption is still high when the input and output have different numbers of channels. Recently, ShuffleNet V2 [40] indicates that equal channels minimises memory access cost (MAC). In the convolution operation, we assume that the size of the convolution kernel is $n \times n$, and the number of channels for input and output are $c_1$ and $c_2$. FLOPs is the abbreviation of floating point operation per second which is a very important standard to measure the operating efficiency. On the feature map of spatial size $h \times w$, the calculation of the FLOPs is as shown in Equation (1):

$$FLOPs = h \times w(c_1 \times n \times n \times c_2) = n^2 hw c_1 c_2$$  (1)

If the storage space of the cache is large enough to store all the feature maps and parameters, the MAC is calculated as shown in Equation (2):

$$MAC = n^2(hw + c_1 c_2) + n^2 hw c_2$$  (2)

In Equation (2), the size of the feature map is $h \times w$ and the convolution kernel is $n \times n$, and all input feature maps and convolution kernels need to be loaded into the cache, the number of memory access operations is $n^2(hw + c_1 c_2)$. Then the result of calculation is returned to memory, which needs to access the memory $n^2 hw c_2$ times.

The inequality about MAC can be simplified by mean value inequality ($c_1 + c_2 \geq 2\sqrt{c_1 c_2}$). The simplified MAC inequality is shown in Equation (3):

$$MAC \geq n^2(2hw\sqrt{c_1 c_2} + c_1 c_2)$$  (3)

According to this result, it can be found that when $c_1$ and $c_2$ are equal, the MAC can take the minimum value.

The stem block can reduce the loss of information in the original input image, which improves the ability of feature expression without adding computational cost too much. Our experimental results show that the addition of this structure can effectively improve the detection performance. Unlike PeleeNet, we have the same number of feature channels in the stem block (see Figure 3(a)). In our stem block, we only use feature maps with 16 channels for feature extraction and propagation. Our experiments show that this design reduces the computational cost of the stem block by 33.2%.

3.1.2 Stage block

Motivated by GoogLeNet [32] and PeleeNet [25], we apply a three-way dense layer to fuse different scales of feature maps which is named stage block (see Figure 3(b)). In PeleeNet [25], the stage block is used to extract features of various sizes of objects, one way of the block uses a $1 \times 1$ convolution and a $3 \times 3$ convolution to extract features of small objects, the other way uses two stacked $3 \times 3$ convolutions to capture large-size objects. However, in our detection task, it is mainly used to detect small objects in the aerial scene, so we need to perform more refined feature extraction of small objects. In the structure of our stage block (see Figure 3(b)), the design principle fully takes into account the advantages of $1 \times 1$ small convolution kernel for feature extraction of small objects. We concatenate these two layers and the previous layer for feature fusion. Meanwhile, we add a $1 \times 1$ convolution kernel on the last layer of each stage block, which enhances the feature representation ability of small objects. Moreover, we still use the design principle of equal channels in the stage block, which can accelerate the extraction and propagation of the features.
Recent researches (PeleeNet [25], NIN [42], GoogleNet [32], InceptionNet [41]) show that $1 \times 1$ convolution kernel plays a cross-channel aggregation role, and it can reduce the parameters and dimensions of the feature maps. In other words, if the $1 \times 1$ convolution kernel is used, and this operation is a linear combination of multiple feature maps, it can realise the change of feature maps in the number of channels. When the convolutional layer passes through the excitation layer, the $1 \times 1$ convolution adds non-linear activation to the learning representation of the previous layer, which improves the expressive ability of the network. The $1 \times 1$ convolution kernel can perform fine feature extraction in a small range, which makes the feature of the small object to be extracted better. Therefore, the advantages of $1 \times 1$ convolution kernel for small object detection are fully utilised in our model, which improves the detection accuracy and reduces the detection time.

### 3.2 Anchor box

The anchor box has a basic size parameter, which gets different sizes of boxes based on the area scales and aspect ratios. In Faster R-CNN, the anchor box has different shapes ($128^2$, $256^2$, $512^2$) with three different aspect ratios (1:1, 1:2, 2:1). In this way, a total of nine kinds of anchors with different shapes are obtained. On the basis of the obtained feature map, the centre point of the corresponding original image is obtained through a $3 \times 3$ sliding window. In each window position, an area in the original picture can be inversely derived according to nine different anchors, and then the size and coordinates of the area can be obtained by calculation.

However, the anchor box generated in the Faster R-CNN is used to detect the object in the natural scene, which is not suitable for a specific detection task such as small-object detection in aerial scene. As shown in Figure 4, the object detection in the natural and aerial scenes has a big difference. In the natural scene, the size of the object is large, the background is simple, the object features are obvious, and the number of objects is small. While in the aerial image, the size of the object is small, the background is complex, the features are not obvious, and the number of objects is large. Therefore, we need to set the appropriate size anchor box on our detection task.

In the object detection of natural scenes, the parameters of the anchor are set to: Base size is 16, ratios are (2:1, 1:1, 1:2), and

**FIGURE 4**  The anchor box in natural scene and aerial image.
TABLE 1 The anchor box in faster R-CNN and our model

| Ratio (2:1) | Ratio (1:1) | Ratio (1:2) |
|------------|------------|------------|
| Faster R-CNN | 184 × 96  | 128 × 128  | 88 × 176  |
|            | 368 × 192  | 256 × 256  | 176 × 352 |
|            | 736 × 384  | 512 × 512  | 352 × 704 |
| Our model  | 62 × 38    | 38 × 38    | 38 × 62   |
|            | 84 × 52    | 52 × 52    | 52 × 84   |
|            | 110 × 68   | 68 × 68    | 68 × 110  |

scales are (8, 16, 32). After a series of operations, nine anchors with three scales (128, 256, 512) and three ratios (2:1, 1:1, 1:2) are finally generated (see Table 1). As shown in Table 1, the size of the anchor box generated in the Faster R-CNN is so large that it is not suitable for vehicle object detection in aerial scenes. For the question of how to set the appropriate size of anchor box, YOLO9000 [15] proposes a k-means clustering algorithm to find the appropriate anchor box size. The YOLO9000 [15] does not adopt the method of manually selecting the anchor box, but automatically obtains the size of the anchor boxes by adopting k-means clustering on the bounding box of training set, which can significantly enhance the stability of the model and improve the detection accuracy. However, this method needs to specify the number of k first and is very sensitive to the initialisation of seed points. In addition, this method also significantly increases the amount of calculation.

Differential search algorithm [16] is designed to be a stochastic direct search method, which has the advantage of being easily applied to experimental minimisation. Inspired by reference [17], we use a differential search algorithm to optimise the ratios and scales of the anchors. We aim to find the optimal three scales and three ratios on the anchor setting. The goal of this algorithm optimisation is to maximise the overlapping area of the vehicle bounding-box and anchor box. We set the base size of anchor as 100. Through optimisation calculations, we have obtained the best three scales of 0.38, 0.52, and 0.68, and the three ratios of 1.62:1, 1:1, and 1:1.62.

After a series of operations, it produces nine anchors of different sizes (see Table 1), which basically cover the size of all types of vehicles. Compared with the anchor box in the Faster R-CNN, our anchor box increases the mean average precision (mAP) by 9.1% in Munich dataset, and the detection time is reduced by 17%. The reason is that fewer detections (fewer poor aligned boxes with small confidence scores) are generated and thus the non-maximal suppression (NMS) is faster.

### 3.3 Training stage

In the training process of the proposed network, we have 160k iterations on our model. In each iteration, the region proposal network has two types of predictive value outputs, binary classification and bounding box regression adjustment. The feature maps of all positive and negative sample regions are input into the loss function, and two output layers are obtained by iteration. The first layer outputs the probability score of the object vehicle in each predicted region, which can be obtained by a softmax classifier. The second layer outputs the coordinate vector of each prediction region after bounding-box regression and fine tuning. Similar to [10], the loss function is defined as shown in Equation (4):

\[
A = \frac{1}{N_{db}} \sum_I L_{cl}(p_i, p_i^*)
\]

\[
B = \lambda \frac{1}{N_{lbir}} \sum_i p_i L_{lbir}(loc_i, loc_i^*)
\]

\[
L(p_i, loc) = A + B
\]

where \(i\) is the index parameter of mini-batch. \(p_i^*\) is the ground truth label, if the region box is positive \(p_i^* = 1\), if the region box is negative \(p_i^* = 0\). \(L_{cl}\) is a softmax loss function and indicates the classification of the vehicle and background. \(L_{lbir}\) outputs the coordinates vector \(loc = (x, y, w, h)\) in each predicted region, where \(x, y\) represent the top-left coordinates, \(w, h\) denote the width and height. In each iteration, we process a batch for training data containing the same number of the positive and negative region box. \(N_{db}\) is normalised by the mini-batch size, and \(N_{lbir}\) is normalised by the number of anchor locations.

We set \(\lambda = 2\) to make \(L_{lbir}\) and \(L_{lbir}\) have the same weight. Moreover, \(L_{lbir}\) denotes a smooth \(L_1\) loss, which is the same as that in Fast R-CNN [10]. It is defined as Equation (5):

\[
L_{lbir}(loc, loc^*) = f_{l_1}(loc - loc^*)
\]

\[
f_{l_1}(x) = \begin{cases} 0.5x^2, & \text{if } |x| < 1, \\ |x| - 0.5, & \text{otherwise} \end{cases}
\]

In order to suppress the redundant boxes, we need to use NMS algorithm. In this algorithm, the suppression is an iterative-traversal-elimination process, its purpose is to remove redundant detection boxes and retain the best one.

### 3.4 Training details

The proposed network can be trained by stochastic gradient descent. In order to prevent over-fitting of training model, we adopted the pre-trained PelceNet model for VOC 2012 to initialise the stem block. In addition, the parameters of weight are randomly initialised in the new layer from a zero-mean Gaussian distribution with a 0.01 standard deviation. In the training process, we use a weight decay of 0.0001 and a momentum of 0.9. The learning rate is set at 0.001 for the first 100k iterations, and 0.0001 for the next 80k iterations. The RPN batch size is set to 256. Then we generate approximately 300 overlapped candidate region boxes. In order to suppress those redundant boxes, NMS algorithm is adopted on the proposed regions based on the confidence score.
4 | EXPERIMENTAL RESULTS

In this section, we present the results of our model in vehicle detection. Meanwhile, the experimental results are analyzed in detail and compared with other state-of-the-art detection algorithms.

4.1 | Dataset description

In the experiment, we use two aerial datasets to evaluate our model. The first dataset is a public vehicle dataset, which is collected over the city of Munich, Germany. The other one is our collected vehicle dataset, which is captured over the city of Nanjing, China. These two datasets contain a large number of high-resolution aerial vehicle images.

(1) Munich vehicle dataset: The Munich vehicle dataset is a public dataset that has been widely used by many researchers to evaluate the performance of aerial vehicle object detection [3–5]. The images are captured from an airplane by a Canon Eos 1Ds Mark III camera with a resolution of $5616 \times 3744$ pixels, 50 mm focal length and they are stored in JPEG format. The Munich vehicle dataset is collected by Liu and Mattyus [2] and can be downloaded from [45].

In order to overcome the limitation of dataset on training, each original aerial image ($5616 \times 3744$ pixels) is cropped into $11 \times 10$ image blocks ($702 \times 624$ pixels) with overlap. The data augmentation is divided into three steps. In the first step, we add Gaussian noise to each image block to double the original dataset. In the second step, we enlarge the image by 10% and reduce it by 10% to expand the dataset. The original image is rotated vertically, horizontally and mirrorly to expand the dataset and to prevent over-fitting. Most of the images are captured from aerial photographs of city roads. Our collected dataset is captured over the city Nanjing, China. We have uploaded the dataset to the public repository, and readers can download the dataset from the link in [45].

(2) Our collected dataset: The collected vehicle dataset contains 615 high-resolution aerial images, and the pixel of image is $1368 \times 770$. The UAV captures these images at a height of 60 m. The vehicle information is annotated on each image. There are approximately 30 car-type samples annotated in every image, so our collected dataset contains about 18,450 samples with the ground truth. We choose 80% of the data as training samples and the remaining data as testing samples. In the training stage, the original image is rotated vertically, horizontally and mirrorly to expand the dataset and to prevent over-fitting. Most of the images are captured from aerial photographs of city roads. Our collected dataset is captured over the city Nanjing, China. We have uploaded the dataset to the public repository, and readers can download the dataset from the link in [45].

4.2 | Evaluation index

In our experiments, four typical indicators are used to evaluate the detection performance, namely, precision rate, recall rate, mAP and $F_{1-score}$.

The precision rate is for the prediction result, which indicates how many of the samples are predicted to be truly positive. The definition of precision rate is shown in Equation (6):

$$\text{precision} = \frac{TP}{TP + FP}$$

where true positive ($TP$) represents the number of positive samples predicted to be positive, and false positive ($FP$) represents the number of negative samples predicted to be positive.

The recall rate is the ratio of the number of samples detected to the total number of samples, which is an important indicator to measure the detection performance. It is defined as Equation (7):

$$\text{recall} = \frac{TP}{TP + FN}$$

Ideally, the precision and recall achieve the desired results at the same time, but in fact these two indicators are contradictory in some cases. In different situations, we need to judge which one we want it to be higher. At this time, it is necessary to have an evaluation index of comprehensive recall rate and precision rate. The $F_{1-score}$ is an important indicator used to measure the detection performance in object detection, which takes the precision and recall rate of the classification model into account. The definition of $F_{1-score}$ is shown in Equation (8):

$$F_{1-score} = \frac{2 \times \text{recall} \times \text{precision}}{\text{recall} + \text{precision}}$$

The mAP is a detection index proposed to solve the single point value limitation of precision rate, recall rate and $F$-measure, which can comprehensively reflect the global performance. The definition of the mAP is as follows (where $P$ and $R$ represent the precision rate and recall rate, respectively):

$$AP = \int_{0}^{1} \text{Precision(Recall)}d(\text{Recall})$$

$$\text{mAP} = \frac{1}{\sum_{i=1}^{N} \text{AP}_{i}}$$

The intersection-over-union (IoU) represents the ratio of the overlap between the prediction area and the ground-truth box. If the IoU of the predicted region is higher than 0.6, we assign a positive label to it. Whereas, if the IoU of a predicted region is lower than 0.3 for all ground-truth box, we assign a negative label to it. Then, the remaining regions are disregarded. The IoU ratio is defined as follows:

$$\text{IoU} = \frac{\text{area}(P \cap G)}{\text{area}(P \cup G)}$$

4.3 | Results on Munich vehicle dataset

In this study, we compare the results of our proposed method with those of the methods in reference. We do not copy the
results of other algorithms directly, and all algorithms are reproduced on the Munich dataset and our collected dataset. This ensures that the detection results are not affected by hardware differences. All detection algorithms have the same training and test sets, and the evaluation criteria of detection results are exactly the same.

We have evaluated the performance of our proposed model on the test set of the Munich dataset. The baseline model is PeleeNet. Table 2 shows that the effects of various design choices and components of performance. The design choices of the stem block, small convolution kernel and transition layer with the same channels have a contribution to the improvement of the detection accuracy and the reduction of computational consumption of the model. The components of the three-way dense layer and appropriate anchor box have an effect on the improvement of the model detection accuracy.

Moreover, in our experiments, we find that although the appropriate anchor box is not helpful to model computing consumption, this design can significantly reduce detection time. The appropriate anchor box can accelerate the iteration of the model, and make the model locate the position of the detected vehicle quickly and accurately. It can be seen that after combing all these design choices and components, our proposed detection network achieves 85.8% precision and 242 million MAC on Munich vehicle dataset, which achieves state-of-the-art performance in terms of accuracy and computational cost.

As we can see from Table 3, our method achieves a compelling detection result on Munich vehicle dataset. The mAP is 85.8%, which is 6.0% higher than that of PeleeNet. Meanwhile, the computational cost and model size of the proposed network is only about half of PeleeNet. The accuracy is also higher than that of ResNet50 and DenseNet-121. Moreover, the computational cost of our detection network is only 6.3% of the cost of ResNet50 and 8.5% of the one in DenseNet-121.

In Table 4, we compare our method with state-of-the-art aerial vehicle detection models in terms of detection accuracy and time. As shown in Table 4, our proposed method achieves state-of-the-art performance in terms of recall rate, precision rate, \(F_1\)-score and detection time. Compared with Yu’s method [6], our model shows superior performance in \(F_1\)-score and mAP, while the detection time is only 65.2%.

Our method achieves a recall rate of 81.7% on the Munich dataset. At the same time, the precision is 91.3%, the \(F_1\)-score is 0.862, the mAP is 85.8% and the detection time is 1.78 s per image. These indicators currently has reached the leading level. In addition, our model has a training time of 0.17 s per iteration on the Munich dataset. The model has been iterated 160k times, so the total training time is about 7.5 hours.

NMS is an algorithm that removes non-maximum values which is an important parameter in object detection. In object detection, after feature extraction and classification recognition, each sliding window will get a score. But the sliding window caused many windows to be included or intersected with other windows. At this time, NMS is needed to select the candidate boxes with the highest neighbourhood score and suppress the windows with low scores. Figure 5(a) shows our model comparison with other state-of-the-art vehicle detection models in mAP-NMS curve. Compared with other models, the result of our model has a higher mAP when the NMS is different. In particular, the mAP achieves the highest value when NMS is near 0.35, so in our model we choose NMS = 0.35 in test. The precision-recall curve visually shows the recall and precision of the model in the sample population, which is a very important indicator to measure the performance of the detection model. Figure 5(b) shows the precision-recall curves of our proposed method and several other methods. It is not difficult to see from Figure 5(b) that our method has obvious advantages over other vehicle detection methods in precision-recall. This further illus-

### Table 2 Effects various components on performance

| Components                        | From PeleeNet to our method |
|-----------------------------------|-----------------------------|
| Appropriate anchor box            | ✓                           |
| Stem block                        | ✓                           |
| Three-way dense layer             | ✓                           |
| Small convolution kernel          | ✓                           |
| Transition layer with same channels | ✓                     |
| mAP                               | 68.3% 77.4% 79.2% 81.9% 83.7% 85.8% |
| MAC                               | 916 916 612 486 347 242    |

MAC, memory access cost/million; mAP, mean average precision.

### Table 3 The results of various detection networks on Munich vehicle dataset

| Model       | Computational cost (million MACs) | Model size (million parameters) | mAP (%) (Mean Average precision) |
|-------------|-----------------------------------|---------------------------------|----------------------------------|
| VGG16       | 15,346                            | 138                             | 74.5                             |
| ResNet50    | 3832                              | 23.48                           | 78.3                             |
| DenseNet-121| 2833                              | 6.87                            | 77.2                             |
| 1.0 MobileNet| 569                               | 3.32                            | 72.3                             |
| DenseNet-41 | 545                               | 1.13                            | 74.6                             |
| PeleeNet    | 507                               | 2.18                            | 79.8                             |
| Our method  | 242                               | 1.08                            | 85.8                             |
TABLE 4  Performance comparison between different methods

| Method              | Recall rate (%) | Precision rate (%) | $F_1$ score | mAP (%) | Time per image |
|---------------------|-----------------|--------------------|-------------|---------|----------------|
| Fast multiclass [2] | 69.3            | 86.8               | 0.771       | 72.3    | 4.40           |
| VPN_VAD [3]         | 80.3            | 76.2               | 0.782       | 73.7    | 3.63           |
| HRPN [4]            | 78.5            | 87.3               | 0.826       | 79.5    | 3.68           |
| AVPN [5]            | 79.4            | 88.5               | 0.837       | 81.2    | 3.67           |
| Yu’s method [6]     | 80.6            | 89.4               | 0.848       | 83.4    | 2.73           |
| Our method          | 81.7            | 91.3               | 0.862       | 85.8    | 1.78           |

A VPN, accurate vehicle proposal network; HRPN, hyper region proposal network; VPN_VAD, vehicle proposal network, vehicle detection network.

FIGURE 5  Comparisons with other vehicle detection algorithms (a) mean average precision (mAP) versus non-maximal suppression (NMS) curve, (b) precision recall curve.

trates the excellent performance of our proposed method in vehicle detection.

Figure 6 shows the results of our method on the Munich test dataset. The red boxes denote the correct localisation, and the yellow boxes represent missing detection. In Figure 6, we select those image blocks that are more difficult to detect to demonstrate the detection performance of our model. As shown in Figure 6, Figure 6(a) shows the scene in which the object vehicle has a large density and many vehicles need to be detected, Figure 6(b) shows the vehicle accompanied by a complicated background, Figure 6(c) is the scene where the object vehicle has occlusion and shadows. Figure 6(d) is the local detection effect of Figure 6(c).

Figure 6(a) shows some missing samples at the boundary of the image block. This is due to that the part of the information and features of the vehicle appear at the boundary, which makes the detection more difficult. In fact, the image blocks in the test dataset are partially overlapped at the boundary when the original image is segmented, so the samples of the boundary are substantially not missed when the detection of the image block is merged into the original image. In Figure 6(b), a small number of missed samples appear in the shaded area. This is due to the influence of shadows on the extraction of object features, especially when the colour of the vehicle is darker or the occlusion is accompanied. In Figure 6, it shows the detection results of four hard-to-detect image blocks. In this case, the overall detection accuracy reaches 96.1%, which shows that our proposed detection model has superior robustness and accuracy.

4.4  Results on our collected dataset

In order to further demonstrate the robustness of our method, we also have evaluated the algorithm on the collected vehicle dataset. Our model spends 0.193 s per training session on the collected vehicle dataset, the number of iterations is 160k times, so the total training time is about 8.6 hours.

Table 5 shows the numerical results of several methods. Our method reaches a $F_1$-score of 0.923 on the collected vehicle dataset, at the same time the recall rate is 91.4%, the precision rate is 93.2%, and the detection time is 0.048 s per image. These indicators are significantly better than other most advanced detection methods, which demonstrate that our method is more robust and accurate than other methods.

As shown in Figure 7, our model can successfully detect vehicles in a variety of complex backgrounds. More details, because
FIGURE 6 Detection results on Munich vehicle dataset

TABLE 5 Results of different methods for the collected vehicle images

| Method          | Recall rate (%) | Precision rate (%) | $F_1$-score | mAP (%) | Time per image |
|-----------------|-----------------|--------------------|-------------|---------|----------------|
| Fast multiclass [2] | 76.8            | 81.7               | 0.792       | 77.3    | 0.139          |
| VPN_VAD [3]     | 86.3            | 80.3               | 0.832       | 81.6    | 0.155          |
| HRPN [4]        | 86.9            | 85.4               | 0.861       | 83.8    | 0.156          |
| AVPN [5]        | 86.7            | 84.9               | 0.860       | 84.2    | 0.158          |
| Yu’s method [6] | 88.6            | 87.4               | 0.880       | 87.3    | 0.106          |
| Our method      | 91.4            | 93.2               | 0.923       | 91.2%   | 0.048          |

AVPN, accurate vehicle proposal network; HRPN, hyper region proposal network; VPN_VAD, vehicle proposal network_vehicle detection network.
the collected vehicle datasets have higher pixels than the Munich dataset, the vehicles detected on the collected dataset has higher scores, and most of the detected vehicles scores more than 0.95, which indicates that the detection results have higher reliability.

5 | DISCUSSION

In this study, compared with the lightweight network PeleeNet, the proposed network increased by 6.0% on mAP in the Munich dataset, while the computational cost and model size are decreased by approximately 50%. Compared with the most advanced aerial vehicle detection of Yu’s method, the proposed method has increased the recall rate and the precision rate by 1.1% and 0.9%, respectively, but the detection time is only 65.2% of this method. All of these indicate that the proposed method has reached the most advanced level in the field of aerial vehicle detection.

Although our detection model achieves good performance on vehicle detection in aerial image, there are still some flaws in this model. The first is in hard example detection, for example, when some vehicles in aerial image are partially occluded or in the shadow, our detection model is not able to perform effective detection. In addition, we only detect car-type vehicles in this study, the detection of fewer vehicle types is also a drawback that exists in our model. For future study, we will pay attention to extract the features of the hard example vehicles, and improve the detection effect of the hard examples. Moreover, we will increase the categories of vehicles, and make the model accurately detect the different types of vehicles, which is also the focus of our future study.

6 | CONCLUSIONS

In this study, we propose a fast and accurate aerial vehicle detection model. In our model, we have established a lightweight network to extract the characteristics of the object vehicle. At the same time, we make some improvements on the feature extraction network for vehicle detection in aerial scenes. For example, transition layers with same channels and small convolution kernel for object feature extraction are introduced into
the feature extraction network. We have combined the proposed object feature extraction network with the target detection algorithm Faster R-CNN, which makes the algorithm better for the detection of small objects in aerial scenes. In addition, we set the appropriate anchor box according to the characteristics of the objects in the proposed model, which further improves the accuracy and efficiency of the model. Moreover, we evaluate our proposed method in the Munich vehicle dataset and the collected dataset.

REFERENCES

1. Liu, R., et al.: Multiscale road centerlines extraction from high-resolution aerial imagery. Neurocomputing 329(1), 384–396 (2017)
2. Liu, K., Mattyus, G.: Fast multiclass vehicle detection on aerial images. IEEE Geosci. Remote Sens. Lett. 12(9), 1938–1942 (2015). https://doi.org/10.1109/LGRS.2015.2439517
3. Zhong, J., et al.: Robust vehicle detection in aerial images based on cascaded convolutional neural networks. Sensors 17(12), 2720–2737 (2017)
4. Tang, T., et al.: Vehicle detection in aerial images based on region convolutional neural networks and hard negative example mining. Sensors 17(2), 336–352 (2017)
5. Deng, H., et al.: Toward fast and accurate vehicle detection in aerial images using coupled region-based convolutional neural networks. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 10(8), 3652–3664 (2017)
6. Yu, Y., et al.: Vehicle detection from high-resolution remote sensing imagery using convolutional capsule networks. IEEE Geosci. Remote Sens. Lett. 16(12), 1894–1898 (2019)
7. Chen, Z., et al.: Vehicle detection in high-resolution aerial images via sparse representation and superpixels. IEEE Trans. Image Process. 54(1), 103–116 (2016)
8. Kembhavi, A., et al.: Vehicle detection using partial least squares. IEEE Trans. Pattern Anal. Mach. Intell. 33(6), 1250–1265 (2011)
9. Ozturk, S., Akdemir, B.: HIC-net: A deep convolutional neural network model for classification of histopathological breast images. Comput. Electr. Eng. 76, 299–310 (2019)
10. Ozturk, S., Akdemir, B.: A convolutional neural network model for semantic segmentation of mitotic events in microscopy images. Neural Comput. Appl. 31(8), 3719–3728 (2019)
11. Girshick, R., et al.: Region-based convolutional networks for accurate object detection and segmentation. IEEE Trans. Pattern Anal. Mach. Intell. 38(1), 142–158 (2016)
12. Girshick, R.: Fast R-CNN. In: International Conference on Computer Vision (ICCV), Santiago, Chile, pp. 1440–1448 (2014)
13. Ren, S., et al.: Faster R-CNN: towards real-time object detection with region proposal networks. IEEE Trans. Pattern Anal. Mach. Intell. 39(6), 1137–1149 (2017)
14. Redmon, J., et al.: You only look once: Unified, real-time object detection. In: IEEE Conference on CVPR, Las Vegas, USA (2016)
15. Redmon, J., Farhadi, A.: YOLO9000: Better, faster, stronger. In: IEEE Conference on CVPR, Honolulu, USA, pp. 6517–6525 (2017)
16. Storn, R., Price, K.: Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces. J. Global Optim. 11, 341–359 (1997)
17. Zlocha, M., et al.: Improving RetinaNet for CT lesion detection with dense masks from weak RECISt labels. In: Medical Image Computing And Computer Assisted Intervention, pp. 402–410. Springer, Cham (2019)
18. Liu, W., et al.: SSD: Single shot multibox detector. In: ECCV 2016, Amsterdam, The Netherlands, pp. 21–37 (2016)
19. Lin, T., et al.: Feature pyramid networks for object detection. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, USA (2017)
20. Zhang, S., et al.: Single-shot refinement neural network for object detection. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, Utah, USA, pp. 4203–4212 (2018)
21. Hong, S., et al.: PVANet: Lightweight deep neural networks for real-time object detection. arXiv:1611.08588 (2016)
22. Howard, A.G., et al.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv:1704.04861 (2017)
23. Zhang, X., et al.: Shufflenet: An extremely efficient convolutional neural network for mobile devices. arXiv:1707.01083 (2017)
24. Zoph, B., et al.: Learning transferable architectures for scalable image recognition. arXiv:1707.07012 (2018)
25. Wang, R., et al.: Pcle: A real-time object detection system on mobile devices. In: NIPS, pp. 1967–1976. Curran Associates Inc., Red Hook, New York (2018)
26. Zhang, C., et al.: Multiple instance boosting for object detection. In: NIPS, pp. 1417–1422. MIT Press, Cambridge, Massachusetts (2005)
27. Chandrappa, D.N., et al.: Face detection using a boosted cascade of features using OpenCV. In: ICIP 2012, Bangalore, India, pp. 399–404 (2012)
28. Ojala, T., et al.: Gray-scale and rotation invariant texture classification with local binary patterns. IEEE Trans. Pattern Anal. Mach. Intell. 24(9), 971–987 (2002)
29. David, G.L.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vision 60(2), 91–110 (2004)
30. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR 2005, San Diego, California (2005)
31. Felzenszwalb, P.F., et al.: Object detection with discriminatively trained part-based models. IEEE Trans. Pattern Anal. Mach. Intell. 32(9), 1627–1645 (2010)
32. Szegedy, C., et al.: Going deeper with convolutions. In: Conference on Computer Vision and Pattern Recognition (CVPR), Boston, Massachusetts (2015)
33. Chollet, F.: Xception: Deep learning with depthwise separable convolution. In: Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, Hawaii (2017)
34. Xie, S., et al.: Aggregated residual transformations for deep neural networks. In: Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, Hawaii (2017)
35. Cheng, G., et al.: Learning rotation-invariant convolutional neural networks for object detection in vhr optical remote sensing image. IEEE Trans. Geosci. Remote Sens. 54(12), 7405–7415 (2016)
36. Ma, N., et al.: ShuffleNet v2: Practical guidelines for efficient CNN architecture design. arXiv:1807.11164 (2018)
37. Szegedy, C., et al.: Inception-v4, inception-ResNet and the impact of residual connections on learning. arXiv:1602.07261 (2016)
38. Lin, M., et al.: Network in network. arXiv:1312.4400 (2013)
39. Tuermer, S., et al.: Airborne vehicle detection in dense urban areas using HoG features and disparity maps. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 6(6), 2327–2337 (2013)
40. Audebert, N., et al.: Segment-before-detect: Vehicle detection and classification through semantic segmentation of aerial images. Remote Sens. 9(4), 1–12 (2017)
41. Duan, X., et al.: Multiple instance boosting for object detection. In: NIPS, pp. 1417–1422. MIT Press, Cambridge, Massachusetts (2005)
42. Tang, T., et al.: Vehicle detection in aerial images based on lightweight deep convolutional network. [308x159]
43. Sheng, J., Liu, N., Sun H.: Vehicle detection in aerial images based on lightweight deep convolutional network. IET Image Process. 2021;15:479–491. https://doi.org/10.1049/ipr.2013