remote sensing

Article

Arbitrary-Oriented Vehicle Detection in Aerial Imagery with Single Convolutional Neural Networks

Tianyu Tang *, Shilin Zhou *, Zhipeng Deng, Lin Lei and Huanxin Zou

College of Electronic Science, National University of Defense Technology, Changsha 410073, China; zpdeng@whu.edu.cn (Z.D.); alaleilin@163.com (L.L.); hxzou2008@163.com (H.Z.)

* Correspondence: ttywhu@163.com (T.T.); slzhou@nudt.edu.cn (S.Z.); Tel.: +86-137-8741-0448 (S.Z.)

Academic Editors: Anzhi Yue, Gonzalo Pajares Martinsanz and Prasad S. Thenkabail
Received: 27 September 2017; Accepted: 13 November 2017; Published: 14 November 2017

Abstract: Vehicle detection with orientation estimation in aerial images has received widespread interest as it is important for intelligent traffic management. This is a challenging task, not only because of the complex background and relatively small size of the target, but also the various orientations of vehicles in aerial images captured from the top view. The existing methods for oriented vehicle detection need several post-processing steps to generate final detection results with orientation, which are not efficient enough. Moreover, they can only get discrete orientation information for each target. In this paper, we present an end-to-end single convolutional neural network to generate arbitrarily-oriented detection results directly. Our approach, named Oriented_SSD (Single Shot MultiBox Detector, SSD), uses a set of default boxes with various scales on each feature map location to produce detection bounding boxes. Meanwhile, offsets are predicted for each default box to better match the object shape, which contain the angle parameter for oriented bounding boxes' generation. Evaluation results on the public DLR Vehicle Aerial dataset and Vehicle Detection in Aerial Imagery (VEDAI) dataset demonstrate that our method can detect both the location and orientation of the vehicle with high accuracy and fast speed. For test images in the DLR Vehicle Aerial dataset with a size of 5616 × 3744, our method achieves 76.1% average precision (AP) and 78.7% correct direction classification at 5.17 s on an NVIDIA GTX-1060.

Keywords: arbitrary-oriented; vehicle detection; single convolutional neural networks (CNN); aerial images; near-real-time

1. Introduction

As a fundamental problem faced by intelligent traffic management, vehicle detection in aerial images plays a very important role for many applications [1–5]. Both position and orientation are important information for practical use. However, this is a challenging task not only because of the complex background in man-made areas and relatively small size of the target, but more important is the variable orientations of vehicles. In aerial images captured from the top view, objects are rotated around the vertical axis, making orientation estimation and accurate localization more challenging. Moreover, the image that needs to be processed is of a large size, but near-real-time detection is required.

Over the last few decades, numerous detectors have been developed for vehicle detection in aerial images [6–12]. These methods have shown promising performance, but most of them return the detection results of axis-aligned bounding boxes, which cannot describe the oriented vehicles precisely. They can be simply divided into two categories: traditional methods and deep convolutional neural network (CNN) based methods. In traditional methods, the work of [2] is worth mentioning here, as the authors present a valid method to detect vehicles with orientation attributes on large-scale aerial images without any geo-reference information. Liu [2] employed a fast binary detector using aggregated channel features and an AdaBoost classifier in a soft-cascade structure to detect the location of the vehicles. Then, a histogram of oriented gradient (HOG) feature with the AdaBoost classifier was used to further classify the orientations of vehicles. This method is based on hand-crafted features, which are not effective enough for vehicle discrimination. Moreover, the sliding window technique leads to heavy computational costs.

Being capable of feature representation, deep convolutional neural networks have achieved dramatic progress in object detection, having faster speed and higher accuracy than traditional methods. They can also be divided into two categories: region proposal-based networks and regression-based single networks. The region proposal-based networks use the idea of region proposal and then classify them; while, the regression-based networks use a single convolutional network to predict bounding boxes and class probabilities simultaneously from an input image. Due to the simple pipeline, the regression-based single networks (e.g., YOLO (You Only Look Once, YOLO) [13], SSD (Single Shot MultiBox Detector, SSD) [14]) have extremely fast detection speed. For the task of vehicle detection in aerial images, it is known that only region proposal-based networks have been investigated. The work of [12] is attractive, which makes up for the shortcomings of method [2]. Deng [12] proposed a region proposal-based network, AVPN (Accurate-Vehicle-Proposal-Network, AVPN),to extract the vehicle's location, and then, a Zeiler and Fergus (ZF) model-based [15] classification network was used to estimate the orientation of the vehicle. However, it is quite slow due to the relatively complex pipeline.

All the above methods have three common drawbacks for oriented vehicle detection in aerial images: (1) They all split the detection process into two stages: proposal generation and object classification. This detection framework is complex, so the detection speed should be improved. (2) Axis-aligned bounding boxes (see Figure 1a) are used during the detection process. The axis-aligned boxes contain not only objects, but also backgrounds, increasing the difficulty of accurate detection. (3) They can only estimate the orientation of the vehicle using a specified discrete direction.

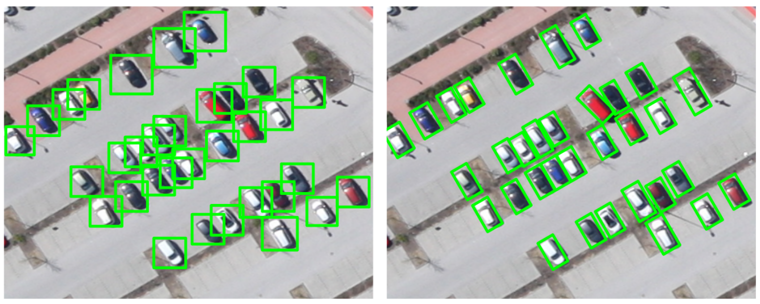
(a) Axis-aligned bounding boxes (b) Oriented bounding boxes

Figure 1. Two styles of bounding boxes.

To address these problems, in this paper, we come up with an end-to-end method, Oriented_SSD, that can automatically provide the vehicle's localization and orientation, only using the input image. Specifically, our work is based on regression-based single networks SSD [14]. A set of default boxes with various scales and ratios is used on each feature map location to produce detection bounding boxes. Meanwhile, offsets are predicted for each default box to better match the object shape, which contain the angle parameter for oriented bounding boxes' generation. Therefore, our method can detect vehicles using oriented bounding boxes (see Figure 1b) with a simple pipeline, which is extremely fast. Additionally, a continuous angle can be estimated for each target. This network is based on the original VGG-16 [16] and adds a few extra feature layers onto it. Predictors from multiple feature maps are combined to make vehicle detection more accurate.

To show the performance of our method, we test and evaluate our method on the DLR Vehicle Aerial dataset [2], which contains numerous vehicles in urban and residential areas, and the Vehicle Detection in Aerial Imagery (VEDAI) dataset [17], which comprises urban areas, as well as agrarian and rural areas with few objects. The results show that our method is faster and more accurate than existing algorithms and is effective for large-scale images captured from both urban and rural areas.

Our main contributions are presented as follows: (1) Our method uses a regression-based CNN model to detect vehicles, which makes the detection very fast, i.e., it takes only few seconds on a 5616×3744 pixel image. To our knowledge, this is the first time that this type of method has been employed in vehicle detection on aerial images. (2) The location and the orientation of the target is estimated at the same time, and oriented bounding boxes are used during the detection process. This end-to-end network makes vehicle detection more efficient and more accurate than previous methods. (3) Our method can predict continuous angles rather than discrete angles, which can describe the targets more precisely.

This paper is organized as follows: Section 2 discusses related works. The proposed method is detailed in Section 3. Section 4 reports the experimental results. Finally, Section 5 concludes this paper.

2. Related Work

Here, we briefly introduce deep CNN-based object detection methods, vehicle detection methods in aerial images and orientation estimation methods.

2.1. Deep CNN-Based Object Detection Methods

Being capable of feature representation, deep convolutional neural networks have achieved dramatic progress in object detection. In the field of computer vision, deep convolutional neural network (CNN)-based object detection methods can be divided into two categories: region proposal-based networks and regression-based single networks. Region proposal-based networks are comprised by R-CNN [18], Fast R-CNN [19], Faster R-CNN [20] and a large number of follow-up improvements (including PVANET [21], R-FCN (region-based fully-convolutional network) [22], MS-CNN (multi-scale CNN) [23]). These famous detection methods have great performance in object detection. However, they have a relatively complex pipeline, which uses the idea of region proposal and then classifying them. Therefore, they are quite slow. The regression-based single networks include YOLO [13], YOLO2 [24] and SSD [14]. These types of detection method use a single convolutional network to predict bounding boxes and class probabilities simultaneously from an input image, reframing detection as a regression problem. Due to the simple pipeline, they are extremely fast. Additionally, SSD uses multiple layers' features for detection and achieves comparable precision with the region proposal-based methods. However, all these methods are designed for object detection in nature scene images, in which the object appears in the horizontal or vertical direction. Thus, axis-aligned bounding boxes are used during the detection process. However, vehicles in aerial images have variable orientations, which cannot be described precisely by axis-aligned bounding boxes.

2.2. Vehicle Detection Methods for Aerial Images

Over the last few decades, numerous detectors have been developed for vehicle detection in aerial images [6–11], including traditional methods and deep CNN-based methods. For traditional methods, some researchers use the road database as an a priori knowledge guide to detect vehicles [3,4,25], which is limited to special scenes covered by the road map database. In recent years, due to the advance of the machine learning technique, many approaches consider object detection as a region-of-interest (RoI) classification problem. In these approaches, vehicle detection processing is split into two distinctive stages: proposal generation and object classification.

The proposal generation stage aims to generate all the bounding boxes of vehicle-like targets. The most common paradigm is based on sliding-window search in which each image is scanned in all positions with different scales [2,3,7,8]. Among these kinds of methods, the aggregated channel feature (ACF) based vehicle detector in aerial images is the most efficient method [2,26]. Nevertheless, searching for targets in high-resolution broad-area aerial images led to heavy computational costs. Another popular paradigm samples hundreds of object-like regions using region-proposal-based methods, e.g., selective search [27], super-pixels, and so on. All these traditional region-proposal generators segment the image and merge the segmentations that are likely to be the same object. Chen [10] generated vehicle-like regions using super-pixels, while Diao [28] used saliency. The RoI generation capability of those traditional region proposal-based methods may become limited or even impoverished under a complex background. What is more, these methods are also computationally expensive.

The object classification stage infers each region's category by learning a classifier. Hand-crafted features with support vector machine (SVM) or the AdaBoost classifier are widely used in vehicle detection. Shao [7] used Haar-like features and local binary patterns (LBP) with a support vector machine (SVM) for vehicle identification. With the AdaBoost classifier, scale-invariant feature transform descriptors (SIFTs) [3], histogram of oriented gradient (HOG) features [8,29] and integral channel features (ICFs) [2] are used to identify the candidate regions. However, these hand-crafted features are not good enough at separating cars from the background in complex environments.

For deep CNN-based methods, some approaches still use the same detection framework as traditional methods. Compared with hand-crafted features or shallow learning features, deep CNN features are more powerful in object representation. Therefore, some methods [11,30,31] replaced hand-crafted features in traditional methods with deep CNN features. This strategy can significantly improve the performance of object detector, but is still time consuming.

Furthermore, some researchers [12,32,33] also investigated whether fast R-CNN or faster R-CNN showed great performance in vehicle detection on aerial images. Deng [12] improved the traditional faster R-CNN for vehicle detection and orientation estimation for aerial images. These methods are all region proposal-based methods, which are not fast enough. As far as we know, there is no research that makes regression-based detection network available for vehicle detection for aerial images, e.g., SSD, YOLO.

2.3. Orientation Estimation

The orientation estimation methods for vehicles can be divided into two categories. One of them is using HOG feature with SVM or the AdaBoost classifier. Liu [2] used the HOG feature with the AdaBoost classifier to divide the detection results of ACF into eight classes. Another way is to use the CNN-based classifier. Chen [34] and Deng [12] used R-CNN and faster R-CNN separately to detect the location of the object in remote sensing images by axis-aligned bounding boxes. Then, another CNN-based classification model was used to classify the orientation of each bounding box. Finally, the oriented rectangle was proposed based on the axis-aligned bounding boxes and rotated angles.

These existing methods divide detection and orientation estimation into two steps, which make the process more complicated and time consuming. Meanwhile, axis-aligned bounding boxes are used during the detection process. They contain not only objects, but also backgrounds, increasing the difficulty of accurate detection. Moreover, these methods can only estimate the orientation of a vehicle in the specified discrete direction.

3. Proposed Method

Our vehicle detection method is based on a feed-forward convolutional network SSD, namely Oriented_SSD. Figure 2 shows the architecture of our method. For an input image, several oriented bounding boxes are proposed from multi-scale feature maps. Then, a non-maximum suppression step is followed to produce the final detections.

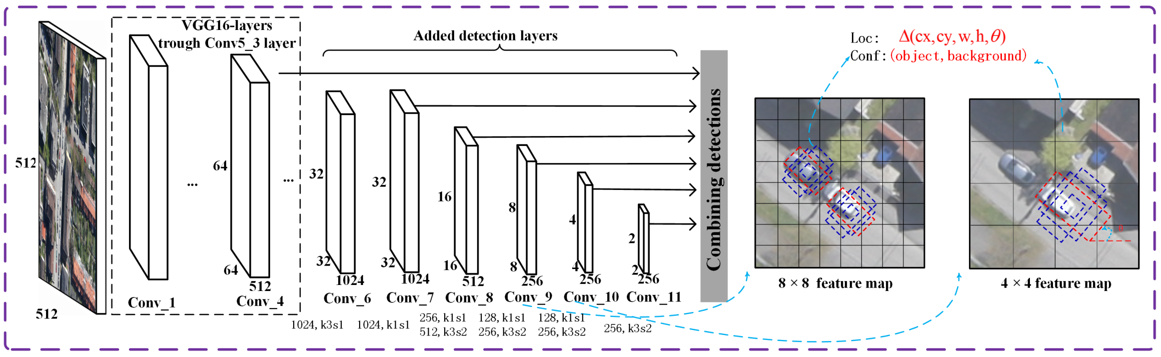

Figure 2. Architecture of our method. This network is based on VGG-16, and some extra convolutional layers are added onto it. Convolutional filters are specified in the format of "filter number, k (kernel size) s (stride)". Convolutional predictors with a kernel size of 3 × 3 are used on six different feature layers.

3.1. SSD

As mentioned in the previous section, SSD [14] is a single CNN for object detection, which reframes object detection as a regression problem. It takes an image as input and outputs both the object location and categories. It uses axis-aligned rectangles to locate the object. It divides the entire image into a $s \times s$ grid and for each grid cell predicts B bounding boxes with confidence scores. Specifically, the axis-aligned detection bounding boxes are produced based on a set of default boxes, which have various scales and aspect ratios and are located on each feature map cell. The default boxes are similar to the anchor in faster R-CNN, which can be denoted by (cx, cy, w, h). cx and cy mean the coordinates of the default box's center point. w and h mean the width and height of the box. To make the detection results more accurate, multi-scale feature maps are used for detection. For each feature map, a set of default boxes is used. Suppose there are n feature maps used. The scale of the default boxes for each feature map is:

$$scale_i = scale_{\min} + \frac{scale_{\max} - scale_{\min}}{n-1}(i-1), \quad i \in [1, n] \tag{1}$$

Here, $scale_{\min} = 0.2$ and $scale_{\max} = 0.9$. The ratios for the default boxes are denoted as $ratio_j \in \{1, 2, 3, \frac{1}{2}, \frac{1}{3}\}$. The width $w_i^j = scale_i\sqrt{ratio_j}$ and height $h_i^j = scale_i/\sqrt{ratio_j}$ for the default box on the i-th feature map are computed. Additionally, for a ratio of one, a default box with scale $scale_i' = \sqrt{scale_i scale_{i+1}}$ is added. Therefore, for each feature map, six default boxes are located at each cell.

For training, the offsets $(\Delta(cx, cy, w, h))$ of a default box to the relative each ground-truth are used to learn the weights of offsets feature maps. At prediction time, offsets are predicted for each default box, and axis-aligned detection bounding boxes are generated according to the default boxes and their corresponding offsets.

3.2. Oriented_SSD

Our network is based on the original SSD. It takes an image as input and outputs the object location using oriented rectangles. The default boxes of our method is also denoted by (cx, cy, w, h). To make the detection bounding boxes oriented, we add an angle parameter to the offsets. Therefore, according to the default boxes and the offsets $(\Delta(cx, cy, w, h, \theta))$, our method can obtain detection results with oriented rectangles.

Following [14], VGG-16 [16] (truncated before any classification layers) is used as the base network. The structure of the VGG-16 model we used is shown in Figure 3. The first convolutional layer (Conv1_1) takes the training images as input and has 64 kernels of size 3 × 3 with a stride of one. Then, a ReLU layer is followed. The second convolutional layer (Conv1_2) takes the output of the previous layer as input and has the same configuration as Conv1_1. After layer Conv1_2, a ReLU layer and a max pooling layer (kernel size: 2×2, stride: two) are used. The configuration of the rest of the layers is similar to layer Conv_1 and can be seen in Figure 3.

Additionally, some extra convolutional layers are added on top of the base network (see Figure 2, Conv_6–Conv_11). The output size of these layers is decreased progressively. Thus, multi-scale feature maps are produced for detection.

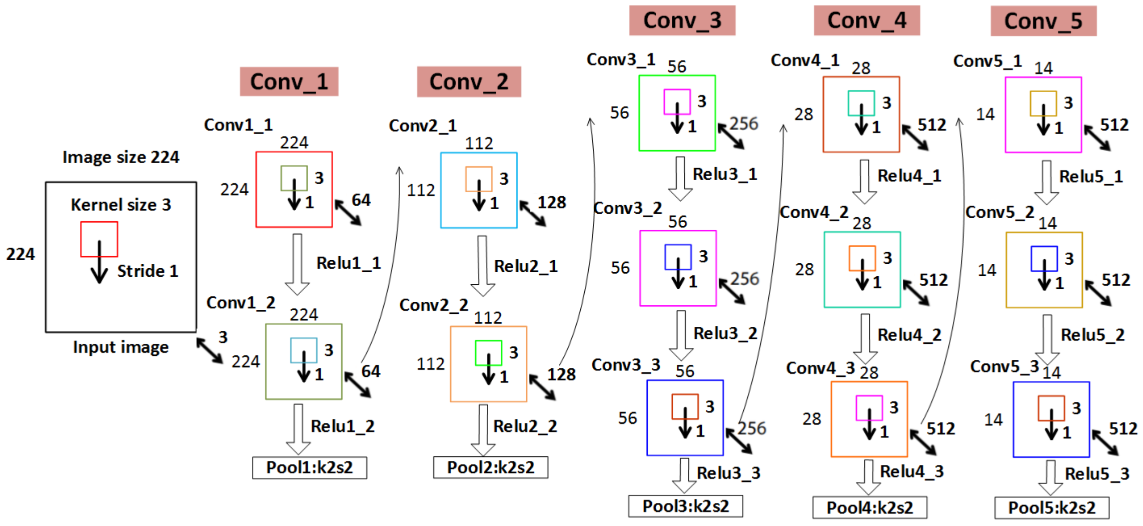

Figure 3. Architecture of truncated VGG-16.

The detection bounding boxes in our method are oriented boxes, denoted by $d = (x_d, y_d, w_d, h_d, \theta_d)$. In order to generate detection bounding boxes, a set of convolutional filters is used to predict the scores and offsets for a set of specified default boxes [14] on all the feature maps. For each cell in a feature map, six default boxes with different ratios and scales are proposed (we use the same ratios and scales as SSD). For an $m \times n$ feature map, the convolutional predictor with the size of 3×3 is applied at each cell. Therefore, for each default box in a cell, the offsets and its score are predicted using the feature at that location. The output of each filter values a score for a category or offsets ($\Delta(cx, cy, w, h, \theta)$) relative to the default box's coordinates. The offsets of a detection bounding box are measured relative to a default box's position on each feature map. The output channels for offsets of detection bounding boxes are $m \times n \times 5$ (five are geometric offsets) and $m \times n \times 2$ for scores (object and background). Thus, $(2 + 5) \times 6$ (six are default boxes) filters are applied on each location of the feature map, and $(2 + 5) \times 6 \times m \times n$ outputs for an $m \times n$ feature map.

Given an input image of size $w_I \times h_I$, the coordinate (x_I, y_I) of the detection bounding box on the image can be computed by the relative default box's coordinate (x_f, y_f) on the $w_f \times h_f$ feature map f:

$$x_I = \frac{w_I}{w_f}(x_f + 0.5) \quad y_I = \frac{h_I}{h_f}(y_f + 0.5) \tag{2}$$

For a location of (x_f, y_f) on the feature map f, the vector at this location along the depth is $(\Delta x, \Delta y, \Delta w, \Delta h, \Delta\theta)$. Therefore, the predicted bounding box at this corresponding location in the image is calculated as Equations (3)–(7):

$$x = a_{fw}\Delta x + x_I \tag{3}$$

$$y = a_{fh}\Delta y + y_I \tag{4}$$

$$w = a_{fw}\exp(\Delta w) \tag{5}$$

$$h = a_{fh}\exp(\Delta h) \tag{6}$$

$$\theta = \Delta\theta \tag{7}$$

Here, a_{fw} and a_{fh} are the width and height of the default box. $a_{fw} = 1.5\frac{w_I}{w_f}$, and $a_{fh} = 1.5\frac{h_I}{h_f}$.

3.3. Training

The images with ground truth annotations $(x_t, y_t, w_t, h_t, \theta_t)$ are used for training. For each ground truth box, if the center of a default box is inside the ground truth and the ratios between the default box and ground truth satisfy Equation (8), then this default box is considered as positive. Otherwise, the default box is labeled as negative.

$$\max\left(\frac{a_{fw}}{w_t}, \frac{w_t}{a_{fw}}, \frac{a_{fh}}{h_t}, \frac{h_t}{a_{fh}}\right) \le 1.5 \tag{8}$$

Then, for each positive default box, offsets are calculated as Equations (3)–(7). Therefore, the offsets are used to learn the weights of the network.

The goal of training the network is to minimize the loss of localization L_{loc} and confidence L_{conf}. Following [14], the loss function is the weighted sum of them:

$$L(x,c,l,g) = \frac{1}{N}(L_{conf}(x,c) + \lambda L_{loc}(x,l,g)) \tag{9}$$

Here, N is the number of positive default boxes. If $N = 0$, the loss is set to zero. x is the label of all default boxes. If the i-th default box is the positive j-th ground truth box, $x_{ij} = 1$. Otherwise, $x_{ij} = 0$. The localization loss is a smooth L1 loss [20]:

$$\begin{aligned} L_{loc}(x,l,g) &= \sum^{N} \sum_{m\in\{cx,cy,w,h,\theta\}} x_{ij} smooth_{L1}(l_i^m - \hat{g}_j^m) \\ \hat{g}_j^{cx} &= (g_j^{cx} - d_i^{cx})/d_i^w \quad \hat{g}_j^{cy} = (g_j^{cy} - d_i^{cy})/d_i^h \\ \hat{g}_j^w &= \log(\frac{g_j^w}{d_i^w}) \quad \hat{g}_j^h = \log(\frac{g_j^h}{d_i^h}) \quad \hat{g}_j^\theta = g_j^\theta - d_i^\theta \end{aligned} \tag{10}$$

Here, l means predicted box, g means ground truth box and d means default box. Additionally, (cx, cy) is the center point of a box, and w, h, θ are the width, height and rotated angle of the box. As the default box has no angle parameters, $d_i^\theta = 0$. The angles are in radians here. Moreover, we use "prior scaling" to adjust the weight of loss on cx, cy, w, h, θ. The prior scaling is set based on experience. The confidence loss is softmax loss over classes' confidences. The parameter c means classes' confidences for object or non-object.

$$L_{conf}(x,c) = -\sum \log c \tag{11}$$

The weight constants λ is set to one.

Moreover, hard negative mining is added to balancing the positive and negative samples. Following [14], the ratio between negatives and positives is 3:1 at most.

4. Experimental Results

In this section, we evaluate our method for oriented vehicle detection on two public datasets, namely the DLR Vehicle Aerial dataset and the Vehicle Detection in Aerial Imagery (VEDAI) dataset.

4.1. Dataset Description and Experimental Configuration

4.1.1. Dataset Description

Two publicly-available datasets are used in the experiments. Following the works of [2], the first ten images of the DLR Vehicle Aerial dataset are used for training and the other ten images for testing. To provide further verification, we also evaluate our vehicle detector on the VEDAI dataset. The specific characteristics of both datasets are summarized in Table 1. The vehicle size, background and number of objects per image are different in the two datasets. The DLR Vehicle Aerial dataset is acquired over Munich, Germany, and contains mainly urban and residential areas. The VEDAI dataset is acquired over Utah, U.S., and contains varying backgrounds such as agrarian, rural and urban areas. Moreover, the VEDAI dataset has two different resolutions and is divided into two parts: VEDAI512 and VEDAI1024. VEDAI512 comprises the downscaled images of VEDAI1024. In our experiments, only VEDAI512 was used.

Table 1. Main characteristics of two datasets. VEDAI, Vehicle Detection in Aerial Imagery (VEDAI).

Dataset	DLR Vehicle Aerial	VEDAI512
#Images	20	1246
Image size	5616 × 3744	512 × 512
GSD (cm/pixel)	13	25
#Objects (car)	Train: 3191/Test: 5799	699
#Objects/image	449.5	1.11
Mean width	27.18 ± 9.09	16.7 ± 5.66
Mean height	25.56 ± 9.40	16.7 ± 5.84

In our experiment, we just train our model and the compared methods on the first 10 images of the DLR Vehicle Aerial dataset. Each training image comes along with its annotation file that contains the oriented bounding box's coordinates (cx, cy, w, h, θ) of each vehicle. In detail, the DLR Vehicle Aerial dataset provides seven vehicle types, such as car, truck, bus, etc. As the number of other types of vehicle is very small, we only select cars with more training samples to train our model. Owing to the limited size of the training set, performing data augmentation to artificially increase the number of training samples is necessary to avoid over-fitting. For data augmentation, each original aerial image is cropped into 165 (15 × 11) image blocks with the same resolution of 702 × 624 pixels, and the adjacent image block overlap ratio is set as 0.5. Then, the image blocks without vehicles are discarded, and the remaining image blocks are rotated with four angles (i.e., 0°, 90°, 180° and 270°). Meanwhile, we generate an annotation file that contains the bounding box of each vehicle for each image block. For effective training, the vehicle annotations that cross image block boundaries need to be discarded. Finally, we reconstruct a training dataset containing 4840 images with annotation files.

For large-scale test images of the DLR Vehicle Aerial dataset, we crop each image into 48 (8 × 6) blocks. To avoid missing detection of vehicles on the cross image block boundaries, we set the overlap of the adjacent image blocks as 50 pixels. The 48 image blocks are detected separately and then stitched together to recombine the original image.

4.1.2. Evaluation Metrics

We adopted three widely-used measures to quantitatively evaluate the performance of our vehicle detection method, namely the precision-recall curve (PRC), average precision (AP) and F1-score. The recall rate measures the fraction of correctly-identified positive detections and true positive detections, while the precision measures the fraction of correctly-identified positive detections and predicted positive detections. The AP metric is measured by the area under the PRC. The higher the value of AP, the better the performance. Moreover, the F1-score is defined as:

$$\text{F1} = \frac{2 \times \text{recall} \times \text{precision}}{\text{recall} + \text{precision}} \tag{12}$$

The F1-score combines the precision and recall metrics into a single measure to comprehensively evaluate the quality of an object detection method.

Generally, a detection result is considered to be a true positive if the Intersection-over-Union (IoU) overlap ratio between a detected bounding box and ground truth bounding box is greater than 0.5. Otherwise, the detection is considered as a false positive. Furthermore, if several detections overlap with the same ground truth, only one detection with the highest overlap ratio is considered a true positive, and others are considered false positives. Additionally, an orientation estimation error histogram is used to evaluate the performance of orientation estimation.

4.1.3. Compared Approaches

In this paper, our compared methods are composed of two parts: object detection and orientation estimation. For object detection, five competitive detection methods are used.

- ACF detector [35]: The aggregated channel feature (ACF) based detector is a traditional state-of-the-art method used in [2]. As a baseline, we use Piotr's Computer Vision MATLAB Toolbox [36] implementation of the ACF detector. This binary detector was trained with a sliding window size of 48 × 48 pixels and 2048 weak classifiers.
- Faster R-CNN [20]: This is a particularly influential detector. In our experiments, both the Zeiler and Fergus (ZF) model [15] and the VGG-16 model [16] are adopted as the feature extractor for detection, namely FRCN_ZF (ZF based Faster R-CNN, FRCN_ZF) and FRCN_VGG (VGG-16 based Faster R-CNN, FRCN_VGG). The ZF model has five convolutional layers, and the VGG-16 model has 16 convolutional layers.
- YOLO [13]: This uses a single feed-forward convolutional network to directly predict classes and bounding boxes. In our experiments, we adopt the detection network from [13], which has 24 convolutional layers followed by two fully-connected layers.
- YOLO2 [24]: This is an improvement of YOLO, which removes the fully-connected layers and uses anchor boxes to predict bounding boxes. In our experiments, we adopt the detection network from [24], which has 19 convolutional layers.
- SSD [14]: This is also an improvement of YOLO, which uses anchor boxes to predict bounding boxes from multiple feature maps with different resolutions. Following [14], we adopt the VGG-16 model as the feature extractor. Moreover, there are two configurations of SSD. SSD300 is trained with the image resized to 300 × 300, and SSD512 is trained with the image resized to 512 × 512. SSD512 has better performance than SSD300 in many detection tasks.

After object detection, three popular methods are tried to estimate the orientation of the compared methods' detection results: the HOG + SVM, Lenet and ZF model-based classifier [15]. From our results, both CNN-based classifiers have higher accuracy than the HOG + SVM classifier. Additionally, the ZF model-based classifier has the best classification performance both in accuracy and speed. Thus, we choose the ZF model-based classifier to estimate the orientation of the compared methods.

These object detection methods followed by the ZF model-based classifier are used as the compared methods. We name them just using the name of the detector (ACF detector, FRCN_ZF, FRCN_VGG, YOLO, YOLO2, SSD300 and SSD512).

4.2. Results on DLR Vehicle Aerial Images

To evaluate our method's performance on the DLR Vehicle Aerial dataset, the experiments are divided into two parts: oriented vehicle detection and orientation estimation.

4.2.1. Evaluation of Vehicle Detection

Figure 4 shows the detection results of different methods. For faster R-CNN, YOLO and SSD, only the best result in each type of method is shown (FRCN_VGG, YOLO2 and SSD512). From this figure, we can see that our methods have better performance than the other methods. Additionally, the deep CNN-based methods have less false detections and less missing detections than the ACF detector, demonstrating the powerful feature representation of deep CNN. Moreover, SSD512 has less missing detections than FRCN_VGG and YOLO2. Therefore, our method is improved based on SSD. The orientation estimation results of our Oriented_SSD512 are more accurate than other compared methods. What is more, Oriented_SSD512 has less missing detections than the original SSD512, which demonstrates that using oriented bounding boxes during detection can help improve the detection accuracy.

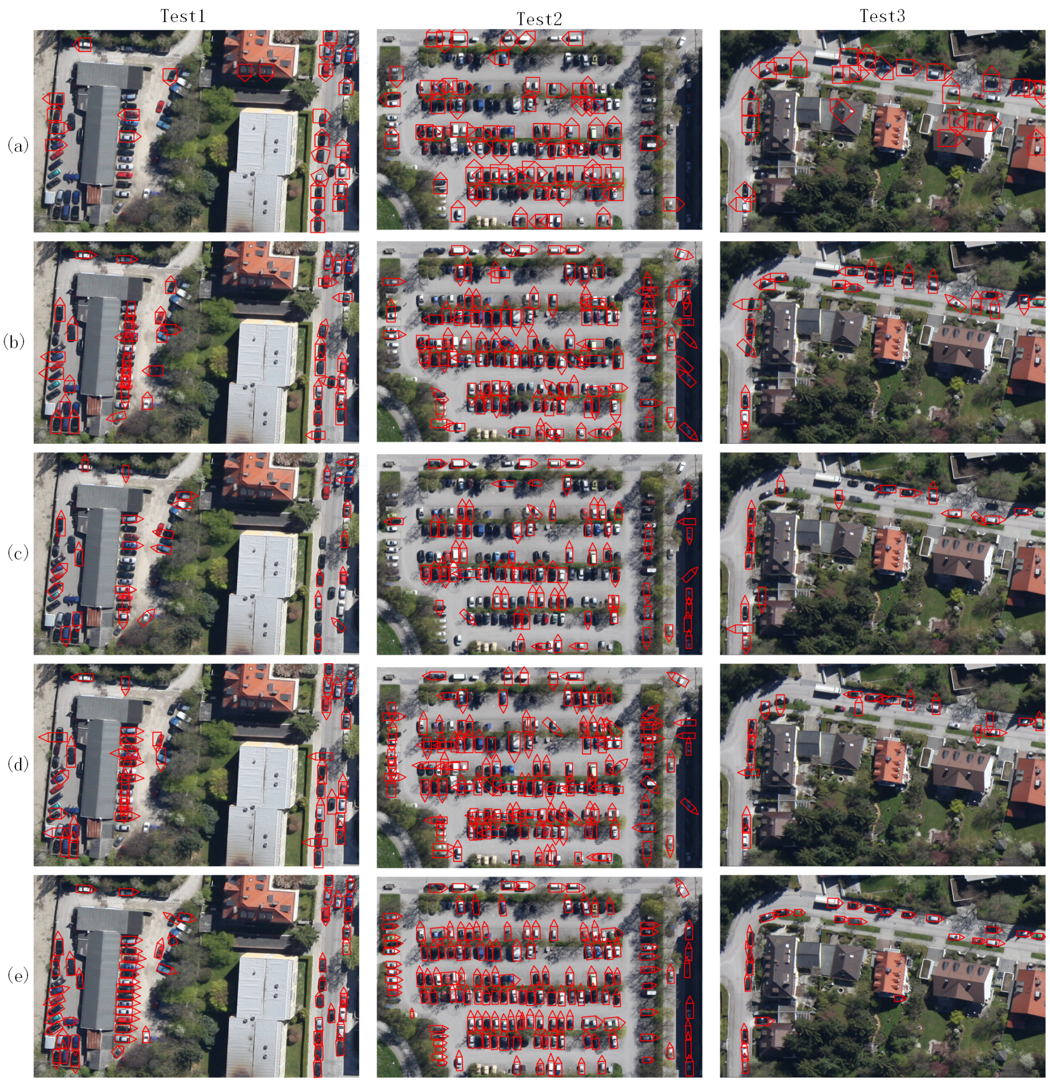

Figure 4. Detection results of different methods for three image blocks on DLR Vehicle Aerial dataset. The red box denotes correct localization, and the triangle represents the direction of the vehicle. (**a**) The results on the first row are detected by the aggregated channel feature (ACF) detector. (**b**) The second row is FRCN_VGG (VGG-16 based Faster R-CNN, FRCN_VGG). (**c**) The third row is YOLO2 (You Only Look Once, YOLO). (**d**) The fourth row is SSD512 (Single Shot MultiBox Detector, SSD). (**e**) The bottom row is Oriented_SSD512.

Figures 5 and 6 display the PRCs and the AP of nine different methods (ACF detector, FRCN_ZF, FRCN_VGG, YOLO,YOLO2, SSD300, SSD512, Oriented_SSD300 and Oriented_SSD512) for the ten test images, respectively. For the compared methods, original axis-aligned bounding box detection results are rotated according to the orientation estimation results from the ZF model-based classifier. Then, these oriented results are used to evaluate the detection performance. Following [37], these results are calculated when IoU = 0.3. As can be seen from these: (1) The CNN-based methods have much better detection performance than the ACF detector. The superior performance of the CNN-based methods demonstrates the high superiority of deep learning feature-based algorithms compared with the algorithms based on human-designed features. (2) Our methods Oriented_SSD300 and Oriented_SSD512 outperform the original SSD300 and SSD512, respectively. This demonstrates that using oriented rectangles during detection can improve the accuracy. Compared to the axis-aligned bounding boxes that contain vehicles and backgrounds, the feature of oriented boxes only represents the objects, which are more precise. (3) Oriented_SSD512 and SSD512 outperform Oriented_SSD300 and SSD300, respectively. This means that feature maps with higher resolution are more valuable in detection tasks. (4) FRCN_VGG has better performance than FRCN_ZF. This result shows that deeper networks have better feature representation abilities, thus improving detection performance. (5) SSD and YOLO2 have higher precision than YOLO. This demonstrates that using features from different layers and anchor boxes can provide better detection results.

Table 2 shows the numerical comparison results of the nine methods on the DLR Vehicle Aerial dataset. The best performances are highlighted in bold. It can be observed that our proposed Oriented_SSD512 achieves the best performance in terms of recall rate, precision rate and F1-score. Oriented_SSD512 and Oriented_SSD300 have better results than the original SSD512 and SSD300 respectively. Furthermore, Oriented_SSD512 achieves an F1-score of 0.82, higher than Oriented_SSD300, YOLO and YOLO2, which demonstrates that Oriented_SSD512 is more accurate for small and dense object detection. Specifically, SSD512_Oriented can improve the recall rate and precision rate effectively. Compared with the ACF detector, CNN-based methods have a higher precision rate owing to the deep feature that has superior vehicle and background classification performance.

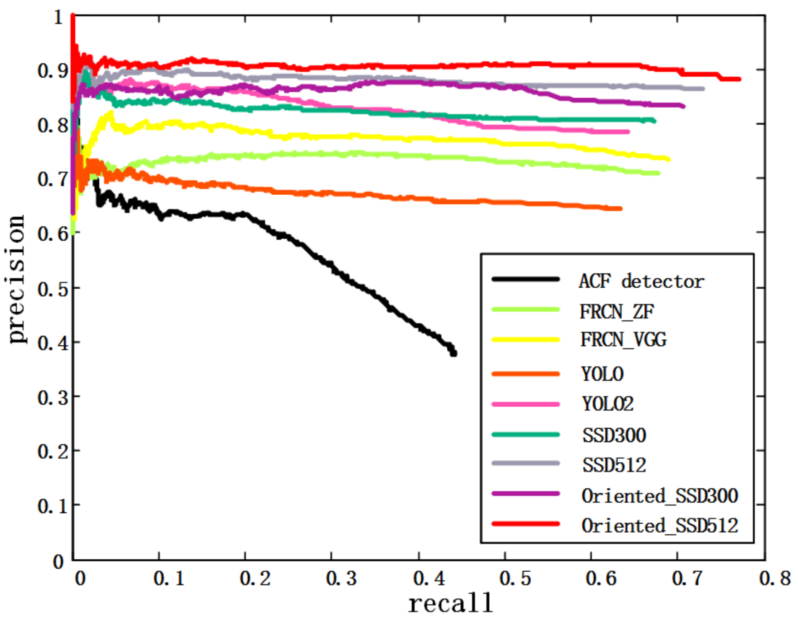

Figure 5. Precision-recall curves (PRCs) of nine different methods on the DLR Vehicle Aerial dataset.

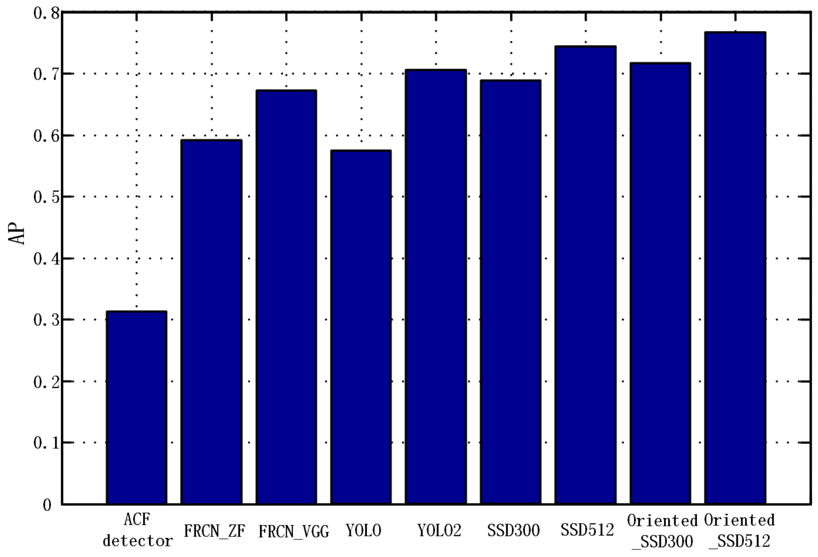

Figure 6. Average precision (AP) of nine different methods on the DLR Vehicle Aerial dataset.

Table 2. Performance comparison between different methods on DLR Vehicle Aerial dataset.

Method	Ground Truth	True Positive	False Positive	Recall Rate	Precision Rate	F1-Score	Time /per Image
ACF detector	5799	3078	4062	53.08%	43.31%	0.47	6.29s
FRCN_ZF	5799	3988	1082	68.77%	78.66%	0.73	5.76s
FRCN_VGG	5799	4076	1017	70.30%	80.03%	0.75	11.32s
YOLO	5799	3557	965	61.34%	78.65%	0.69	4.61s
YOLO2	5799	3877	914	66.86%	80.92%	0.74	**4.22s**
SSD300	5799	4005	985	69.06%	80.26%	0.74	4.61s
SSD512	5799	4400	844	75.88%	83.91%	0.79	5.22s
Oriented_SSD300	5799	4175	963	72.00%	81.25%	0.76	4.50s
Oriented_SSD512	5799	**4572**	**773**	**78.84%**	**85.53%**	**0.82**	5.17s

The processing time is very important for real-time vehicle detection applications. For each large-scale aerial image, we first cropped it into 48 image blocks and then detected vehicles and estimated orientations. Our method can detect vehicles and estimate the orientations at the same time; thus, the processing time for our method is equal to the detection time. For the compared methods, the computational cost consists of two parts: object detection and orientation estimation. The orientation estimation time for each block is 0.04 s, thus 1.92 s for the whole image. The processing time of each method is illustrated in Table 2. YOLO2 has the shortest processing time, and the speed of our method is comparable. Furthermore, using an advanced GPU with higher video memory, we can crop large-scale aerial images into a lower number of image blocks, thus further increasing the speed time of the entire vehicle detection system.

4.2.2. Evaluation of Orientation Estimation

For the compared methods, after the axis-aligned bounding-box detection, we classified the orientation of the vehicles using the ZF model-based classifier. For orientation estimation, eight main directions are clustered (45° rotation difference between adjacent sample groups, respectively), and each cluster is considered as a class, namely 0°, 45°, 90°, 135°, 180°, −135°, −90° and −45°. As the orientations of the ground truth and our methods are not discrete, we divide them into eight classes according to their angle values. For example, if the orientation of the ground truth or our method is between −22.5° and 22.5°, 0° is used to represent the original orientation.

The orientation estimation error histogram of different methods is depicted in Figure 7. Moreover, we calculate the root-mean-square error (RMSE) and weighted mean value (W-Mean) of the orientation prediction error. The results are shown in Table 3. They are calculated by Equations (12) and (14), respectively.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(X_i - \bar{X})^2}{n}} \tag{13}$$

$$W\!-\!Mean = \sum w\,|X_i| \tag{14}$$

Here, X_i means the orientation prediction error (-180, -90, -45, 0, 45, 90, 135, 180). w means the percent of each prediction error for each method.

Table 3. RMSE and weighted mean value (W-Mean) of prediction error for each method.

	ACF Detector	FRCN _ZF	FRCN _VGG	YOLO	YOLO2	SSD300	SSD512	Oriented _SSD300	Oriented _SSD512
RMSE	96.08	97.80	95.17	101.18	82.24	82.34	82.09	78.00	74.38
W-Mean	84.29	68.72	63.59	68.00	43.11	41.49	41.04	38.97	32.85

For our method, the orientation of vehicles is estimated while detecting. For other compared methods, the orientation is estimated by using the ZF model-based classifier on the axis-aligned bounding-box detection results. From this figure, we can see that our method has the best performance in orientation estimation. Moreover, the most common error is when the vehicles are classified in the opposite direction. This is primarily because the vehicle front might be similar to the vehicle rear.

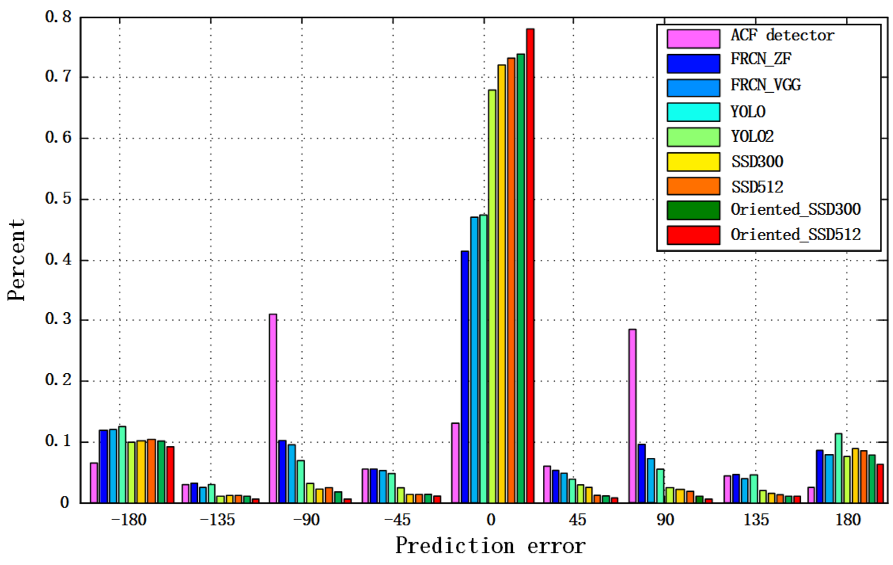

Figure 7. Orientation estimation error histogram of different methods on the DLR Vehicle Aerial dataset.

4.3. Results on VEDAI Images

To demonstrate the robustness of our method, we evaluate it on the VEDAI512 dataset, as well. The numerical comparison results are shown in Table 4.

We use the model trained on the DLR Vehicle Aerial dataset and test on VEDAI512 directly. The best performances are highlighted in bold. It can be observed that our proposed Oriented_SSD512 achieves the best performance in terms of recall rate, precision rate and F1-score, which demonstrates that Oriented_SSD512 is more robust and accurate than the original SSD512 and the other methods.

Figure 8a–c shows our method Oriented_SSD512's results on image blocks of the DLR Vehicle Aerial test aerial images, and Figure 8h–i are the results of the original large aerial images of the DLR Vehicle Aerial dataset. They show that our method can detect most of the vehicles successfully. The missing detected vehicles are mostly located in dense areas or are darker in color. This is because the vehicles located in dense areas have their information easily lost in the small-sized feature maps. Darker vehicles, which have lower image contrast, result in a weak detection response. In future work, we will consider improving the contrast of the image before detection. Compared with [2], CNN-based methods can detect bounding boxes with an adaptive size instead of a fixed window size of 48 × 48 pixels. Additionally, our approach can detect vehicles with orientated bounding boxes directly, which is more precise to describe the vehicles. Figure 8d–g displays several vehicle detection results with the proposed approach on the VEDAI dataset. Our method can detect most of the vehicles successfully, demonstrating the transferability of our method. Therefore, it has great potential for wide field application. However, there are still some missing detections as the vehicle sizes and features might be different between different aerial images.

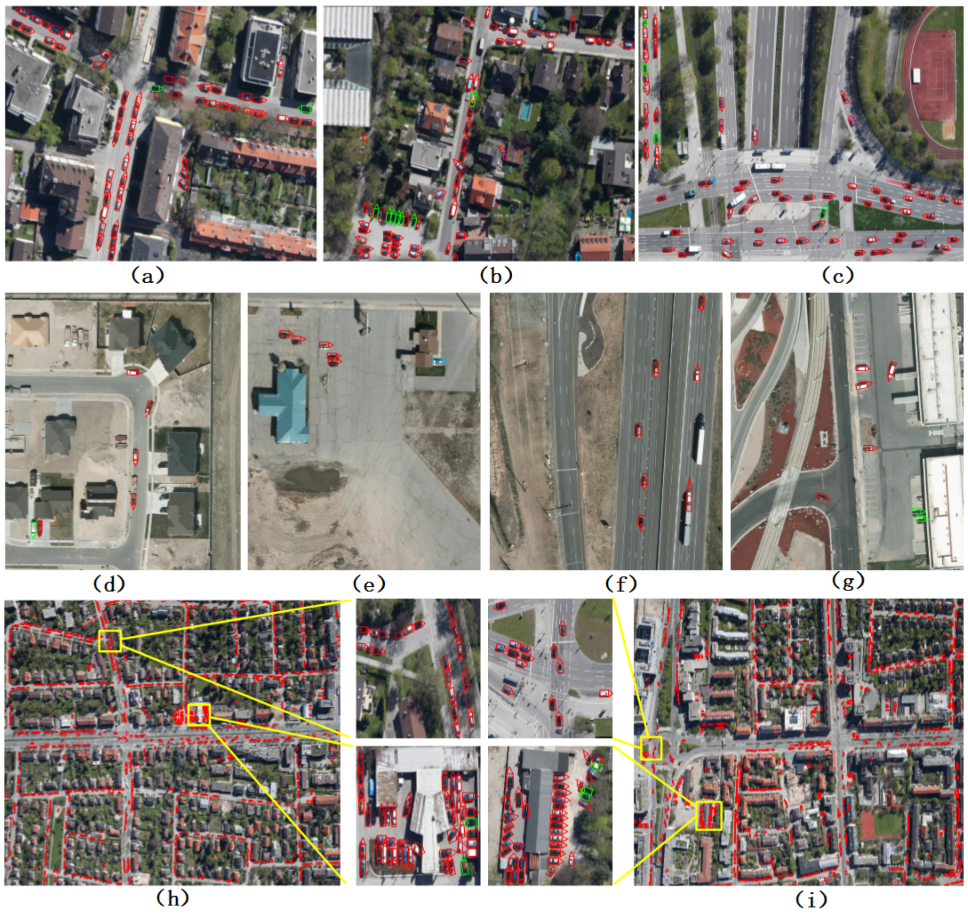

Figure 8. Detection results of our method Oriented_SSD512. A red box denotes correct localization; a blue box denotes false alarm; and a green box denotes missing detection. (**a**–**c**) are image blocks of DLR Vehicle Aerial test aerial images; (**d**–**g**) are VEDAI512 images; (**h**,**i**) are the original large aerial images of the DLR Vehicle Aerial dataset.

Table 4. Results of different methods on the VEDAI512 dataset.

Method	Ground Truth	True Positive	False Positive	Recall Rate	Precision Rate	F1-Score	Time /per Image
ACF detector	1384	501	424	36.20%	45.83%	0.41	0.13s
FRCN_ZF	1384	586	185	42.34%	76.01%	0.54	0.12s
FRCN_VGG	1384	590	190	42.63%	75.64%	0.55	0.24s
YOLO	1384	559	192	40.39%	74.43%	0.52	0.07s
YOLO2	1384	588	183	42.48%	76.26%	0.55	**0.06s**
SSD300	1384	589	**180**	42.56%	76.59%	0.55	0.07s
SSD512	1384	645	196	46.60%	76.70%	0.58	0.11s
Oriented_SSD300	1384	728	201	52.60%	78.36%	0.63	**0.06s**
Oriented_SSD512	1384	**832**	202	**60.12%**	**80.46%**	**0.69**	0.10s

5. Conclusions

In this paper, we presented a single convolutional neural network (CNN) to detect arbitrarily-oriented rectangles for vehicles, in a fully-convolutional manner. Our method detects the location and estimates the orientation of vehicles using an end-to-end network. The detection results of our method are arbitrarily-oriented rectangles, which can describe the vehicles in aerial images more precisely. The quantitative comparison results on the challenging DLR Vehicle Aerial dataset and VEDAI512 dataset show that our method is faster and more accurate than existing algorithms and is effective for images captured from both urban and rural areas. However, as we known, our method still produces some false detection, missing detection, as well as error orientation estimation. Hence, in our future study, we will focus on more precise orientation estimation. Additionally, we will incorporate a multi-GPU configuration to further reduce the computation time.

Acknowledgments: The work is supported by the National Natural Science Foundation of China under Grants 61331015. The authors would like to thank Kang Liu and Gellert Mattyus, who generously provided their image dataset with the ground truth.

Author Contributions: Tianyu Tang contributed to the idea and the data collection of this study. Tianyu Tang developed the algorithm, performed the experiments, analyzed the experimental results and wrote this paper. Zhipeng Deng, Shilin Zhou, Lin Lei and Huanxin Zou supervised the study and reviewed this paper.

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

References

1. Leitloff, J.; Rosenbaum, D.; Kurz, F.; Meynberg, O.; Reinartz, P. An Operational System for Estimating Road Traffic Information from Aerial Images. *Remote Sens.* **2014**, *6*, 11315–11341.
2. Liu, K.; Mattyus, G. Fast multiclass vehicle detection on aerial images. *IEEE Geosci. Remote Sens. Lett.* **2015**, *12*, 1938–1942.
3. Moranduzzo, T.; Melgani, F. Automatic car counting method for unmanned aerial vehicle images. *IEEE Trans. Geosci. Remote Sens.* **2014**, *52*, 1635–1647.
4. Moranduzzo, T.; Melgani, F. Detecting cars in UAV images with a catalog-based approach. *IEEE Trans. Geosci. Remote Sens.* **2014**, *52*, 6356–6367.
5. Chen, Z.; Wang, C.; Luo, H.; Wang, H.; Chen, Y.; Wen, C.; Yu, Y.; Cao, L.; Li, J. Vehicle Detection in High-Resolution Aerial Images Based on Fast Sparse Representation Classification and Multiorder Feature. *IEEE Trans. Intell. Transp. Syst.* **2016**, *17*, 2296–2309.
6. Cheng, H.Y.; Weng, C.C.; Chen, Y.Y. Vehicle detection in aerial surveillance using dynamic Bayesian networks. *IEEE Trans. Image Process.* **2012**, *21*, 2152–2159.
7. Shao, W.; Yang, W.; Liu, G.; Liu, J. Car detection from high-resolution aerial imagery using multiple features. In Proceedings of the IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Munich, Germany, 22–27 July 2012; pp. 4379–4382.

8. Kluckner, S.; Pacher, G.; Grabner, H.; Bischof, H.; Bauer, J. A 3D teacher for car detection in aerial images. In Proceedings of the IEEE International Conference on Computer Vision, Rio de Janeiro, Brazil, 14–21 October 2007; pp. 1–8.
9. Kembhavi, A.; Harwood, D.; Davis, L.S. Vehicle detection using partial least squares. *IEEE Trans. Pattern Anal. Mach. Intell.* **2010**, *33*, 1250–1265.
10. Chen, Z.; Wang, C.; Wen, C.; Teng, X. Vehicle Detection in High-Resolution Aerial Images via Sparse Representation and Superpixels. *IEEE Trans. Geosci. Remote Sens.* **2015**, *54*, 1–14.
11. Chen, X.; Xiang, S.; Liu, C.L.; Pan, C.H. Vehicle Detection in Satellite Images by Hybrid Deep Convolutional Neural Networks. *IEEE Geosci. Remote Sens. Lett.* **2014**, *11*, 1797–1801.
12. Deng, Z.; Sun, H.; Zhou, S.; Zhao, J.; Zou, H. 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.* **2017**, *PP*, 1–13.
13. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 28–30 June 2016; pp. 779–788.
14. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Fu, C.; Berg, A.C. Ssd: Single shot multibox detector. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; pp. 21–37.
15. Zeiler, M.D.; Fergus, R. Visualizing and understanding convolutional networks. In Proceedings of the IEEE International European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014; pp. 818–833.
16. Simonyan, K.; Zisserman, A. Very deep convolutional networks for Large-Scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
17. Razakarivony, S; Jurie, F. Vehicle detection in aerial imagery: A small target detection benchmark. *J. Vis. Commun. Image Represent.* **2016**, *34*, 187–203.
18. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Region-based convolutional networks for accurate object detection and segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *38*, 142–158.
19. Girshick, R. Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision, Boston, MA, USA, 8–10 June 2015; pp. 1440–1448.
20. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149.
21. Kim, K.H.; Hong, S.; Roh, B.; Cheon, Y.; Park, M. PVANET: Deep but Lightweight Neural Networks for Real-time Object Detection. *arXiv* **2016**, arXiv:1608.08021.
22. Dai, J.; Li, Y.; He, K.; Sun, J. R-FCN: Object Detection via Region-based Fully Convolutional Networks. *arXiv* **2016**, arXiv:1605.06409v2.
23. Cai, Z.; Fan, Q.; Feris, R.S.; Vasconcelos, N. A Unified Multi-scale Deep Convolutional Neural Network for Fast Object Detection. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016; pp. 354–370.
24. Redmon, J.; Farhadi, A. YOLO9000: Better, Faster, Stronger. *arXiv* **2016**, arXiv:1612.08242.
25. Zheng, Z.; Zhou, G.; Wang, Y.; Liu, Y. A Novel Vehicle Detection Method With High Resolution Highway Aerial Image. *IEEE J. Sel Top. Appl. Earth Obs. Remote Sens.* **2013**, *6*, 2338–2343.
26. Li, F.; Lan, X.; Li, S.; Zhu, C.; Chang, H. Efficient vehicle detection and orientation estimation by confusing subsets categorization. In Proceedings of the IEEE International Conference on Computer and Communications, Chengdu, China, 14–17 October 2017; pp. 336–340.
27. Uijlings, J.R.R.; Sande, K.E.A.V.D.; Gevers, T.; Smeulders, A.W.M. Selective Search for Object Recognition. *Int. J. Comput. Vis.* **2013**, *104*, 154–171.
28. Diao, W.; Sun, X.; Zheng, X.; Dou, F.; Wang, H.; Fu, K. Efficient saliency-based object detection in remote sensing images using deep belief networks. *IEEE Geosci. Remote Sens. Lett.* **2016**, *13*, 137–141.
29. Tuermer, S.; Kurz, F.; Reinartz, P.; Stilla, U. Airborne vehicle detection in dense urban areas using HoG features and disparity maps. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2013**, *6*, 2327–2337.
30. Ševo, I.; Avramović, A. Convolutional neural network based automatic object detection on aerial images. *IEEE Geosci. Remote Sens. Lett.* **2016**, *13*, 740–744.

31. Qu, T.; Zhang, Q.; Sun, S. Vehicle detection from high-resolution aerial images using spatial pyramid pooling-based deep convolutional neural networks. *Multimed. Tools Appl.* **2016**, pp. 1–13.
32. Sommer, L.W.; Schuchert, T.; Beyerer, J. Fast Deep Vehicle Detection in Aerial Images. In Proceedings of the IEEE Winter Conference on Applications of Computer Vision, Santa Rosa, CA, USA, 24–31 March 2017; pp. 311–319.
33. Deng, Z.; Lei, L.; Sun, H.; Zou, H.; Zhou, S.; Zhao, J. An enhanced deep convolutional neural network for densely packed objects detection in remote sensing images. In Proceedings of the IEEE International Workshop on Remote Sensing with Intelligent Processing, Shanghai, China, 18–21 May 2017; pp. 1–4.
34. Chen, C.; Gong, W.; Hu, Y.; Chen, Y.; Ding, Y. Learning Oriented Region-based Convolutional Neural Networks for Building Detection in Satellite Remote Sensing Images . *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2017**, *XLII-1/W1*, 461–464.
35. Dollar, P.; Appel, R.; Belongie, S.; Perona, P. Fast Feature Pyramids for Object Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *36*, 1532–1545.
36. Dollár, P. Piotr's Computer Vision Matlab Toolbox (PMT). Available online: https://github.com/pdollar/toolbox (accessed on 27 September 2017).
37. Tang, T.; Zhou, S.; Deng, Z.; Zou, H.; Lei, L. Vehicle Detection in Aerial Images Based on Region Convolutional Neural Networks and Hard Negative Example Mining. *Sensors* **2017**, *17*, 336.