Fast Vehicle Detection in UAV Images

Tianyu Tang
College of Electronic Science and Engineering National University of Defense Technology
Changsha, China
ttywhu@163.com

Zhipeng Deng
College of Electronic Science and Engineering National University of Defense Technology
Changsha, China

Shilin Zhou
College of Electronic Science and Engineering National University of Defense Technology
Changsha, China

Lin Lei
College of Electronic Science and Engineering National University of Defense Technology
Changsha, China

Huanxin Zou
College of Electronic Science and Engineering National University of Defense Technology
Changsha, China

***Abstract*—Fast and accurate vehicle detection in unmanned aerial vehicle (UAV) images remains a challenge, due to its very high spatial resolution and very few annotations. Although numerous vehicle detection methods exist, most of them cannot achieve real-time detection for different scenes. Recently, deep learning algorithms has achieved fantastic detection performance in computer vision, especially regression based convolutional neural networks YOLOv2. It's good both at accuracy and speed, outperforming other state-of-the-art detection methods. This paper for the first time aims to investigate the use of YOLOv2 for vehicle detection in UAV images, as well as to explore the new method for data annotation. Our method starts with image annotation and data augmentation. CSK tracking method is used to help annotate vehicles in images captured from simple scenes. Subsequently, a regression based single convolutional neural network YOLOv2 is used to detect vehicles in UAV images. To evaluate our method, UAV video images were taken over several urban areas, and experiments were conducted on this dataset and Stanford Drone dataset. The experimental results have proven that our data preparation strategy is useful, and YOLOv2 is effective for real-time vehicle detection of UAV video images.**

Keywords—CNN; UAV; vehicle detection; CSK; YOLOv2

I. INTRODUCTION

Extracting and identifying vehicles in unmanned aerial vehicle (UAV) images plays an important role for a wide range of applications and is receiving significant attention in recent years [1-10]. However, it is still a challenging problem due to the high resolution with extremely high level of detail, various shooting platform, limited annotated data, and limited processing time for real-time applications.

Classical vehicle detectors are almost based on sliding window search and handcrafted features [3-10]. This is a really time-consuming process. A large number of windows are located over the entire image, and each of them needs to be classified. Moreover, these handcrafted features are not good enough to identify the target. To improve the performance of these methods, some studies combine sliding window search with deep learning features [11]. Accuracy is improved, but the detection speed is still not fast.

Compared with the sliding-window search based methods, region-proposal based methods reduce the computational costs, and they can be divided into two categories: traditional region-proposal generator and convolutional neural networks (CNN) based region-proposal generator. Traditional region-proposal generators segment the image and merge the segmentations that are likely to be the same object, e.g., super-pixels [12], saliency [13], selective search [14], etc. Moreover, handcrafted or shallow-learning based features are usually used to classify the segmentations. Recent studies have incorporated these traditional region-proposal generator with CNN, such as R-CNN [15], Fast R-CNN [16], and etc. The pipeline of R-CNN and its' improved methods [17] can be divided into three steps. They first generate object-like regions and extract highly discriminative features from each region using CNN, and then classify each region with category-specific classifiers. Advance such as Fast R-CNN has reduced the running time with impressive results. But, all traditional region-proposal generators are extremely time consuming. More robust and more efficient methods are CNN based region-proposal methods, e.g., Deepbox [18], region proposal network (RPN) [19], Attend refine repeat [20], etc. The popular Faster R-CNN combines the RPN and Fast R-CNN into a unified network, which achieves much greater performance and further speed improvement. But, this pipeline still uses the idea of region proposal and then classifying them. It's still not quick enough to handle UAV video.

Another type of detection method is based on regression, including YOLO [21], SSD [22] and YOLOv2 [23]. These methods 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, especially YOLOv2. Moreover, they use contextual information about classes and objects' appearance, making less wrong detection. YOLOv2 outperforms state-of-the-art methods like Faster R-CNN with ResNet [24] and SSD [22], both in speed and accuracy. However, to the best of our knowledge, YOLOv2 has not been used for vehicle detection in UAV images.

In this paper, we investigate the use of YOLOv2 for vehicle detection in UAV images, which is one of the first

attempts to successfully use regression based CNN in city management. The training of deep CNN usually requires very large training datasets. As the available manual annotations of vehicles in UAV images are very few, we first use CSK [25] tracking method to help annotate vehicles in images captured from simple scenes. Then, we adopt data augmentation to increase the number of training examples to avoid over-fitting. Finally, YOLOv2 is trained. In our experiments, three regression based CNN models including YOLO, SSD and YOLOv2 are investigated. Comprehensive evaluations on the UAV detection dataset we collected and the Stanford Drone dataset [26] demonstrate that 1) YOLOv2 is effective for real-time vehicle detection of UAV images. It processes images in 0.048s (faster than YOLO and SSD), while achieving top accuracy with a mAP (mean average precision) of 77.12% (vs.67.99% for YOLO, 72.95% for SSD). 2) Due to the proposed data annotation method, data augmentation and fine-tune technology, we can easily annotate small number of training samples, and use them to train a robust detector. Therefore, this data preparation and detection process can handle many new scenes or other object detection, having great potential for wide field application.

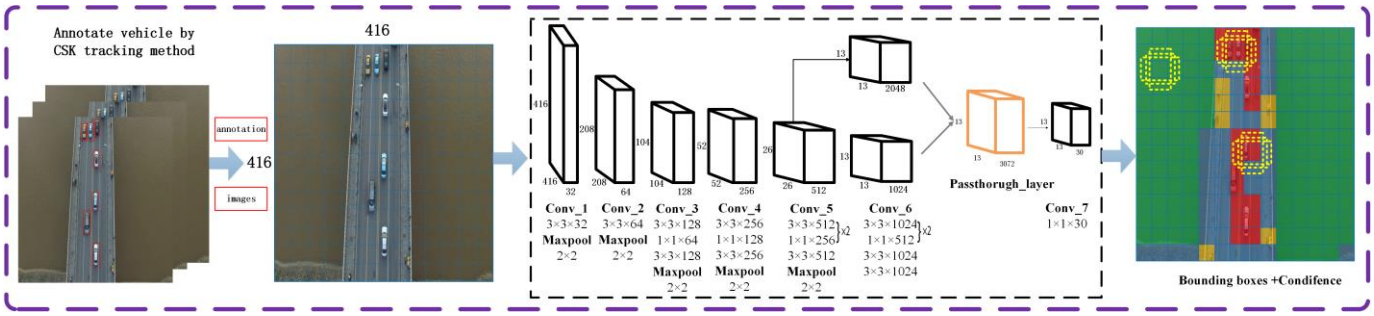

Fig. 1. Pipeline of the training process for our vehicle detection method

II. METHOD

The framework of the training process for our object detection method is illustrated in Fig. 1. It consists of two steps: data preparation and detector training.

A. Data preparation

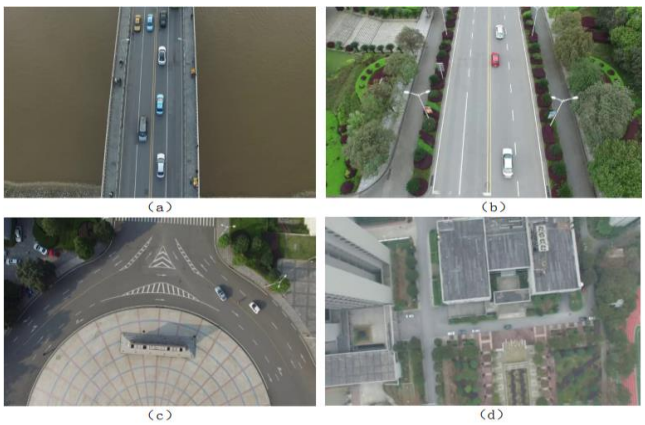

Fig.2. Data collection scenes

First, we use the CSK tracking algorithm to help annotate the training samples. But, this tracking method can only handle the situation that target' size remains unchanged or the target and the platform doesn't move at the same time. Therefore, video images for training are captured from four simple scenes, see Fig.2. Fig.2 (a)-(c) show images captured from fixed platform. In these three simple scenes, UAV is in fixed platform of different heights to capture moving vehicles. Fig.2 (d) shows images captured form moving platform. In this scene, UAV is in rotation to shoot the unmoving vehicle targets. Video images for testing are captured from four more complex scenes, see Fig.4. In these scenes, the UAV platform and vehicle targets are moving at the same time.

To annotate video images for training, the first frame of the video is annotated manually. Then, CSK tracking method is used to annotate the subsequent frames automatically. Thus, we can obtain a number of annotated images from only one manually annotated image. To evaluate the detection method, we annotate testing dataset manually. In actual use, the complex testing data doesn't need to be annotated. Finally, we collect a data set that contains 5151 images for training, and 1374 images for testing.

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. In our implementation, for each training image, we simply rotate it in the step of 90° from 0° to 360°. Thus, based on the 5151 training images, we reconstruct an augmented training data set containing 20604 images with annotation files.

B. YOLOv2 detection

This single neural network takes an image as input, and outputs a set of object bounding boxes with object-like scores.

1) Architecture:

YOLOv2 is a full convolutional network, having 19 convolutional layers and 5 maxpooling layers. The kernel size and number of filters of each layer are illustrated in Fig.1. This model mostly use filters with 3×3 kernels, and use filters with 1×1 kernels to compress the feature map between convolutional layers with 3×3 kernels. To improve the speed of convergence, batch normalization [27] is added on all of the convolutional layers. Moreover, to make the features more suitable for small size object, a passthrough layer is added.

This layer concatenates features of different resolutions by stacking features into different channels. Specifically, the second to last 26 × 26 × 512 feature map is turned into 13 × 13 × 2048 feature map, which is concatenated with the 13 × 13 × 1024 feature map of last convolutional layer. Thus, the features used for detection are fine grained and are much suitable for small size objects. Finally, a 1×1 convolutional layer is used to output the coordinates and object-like scores.

2) Training:

This system divides the input image into a 13×13 grid. If the center of a ground-truth box falls into a grid cell, this cell is responsible for detecting object (we define Pr (Object) =1). Otherwise, if no object locates in the grid cell, we define Pr(Object) =0.

After downsampling by 19 convolutional layers, an 416 ×416 pixels input image gets an output feature map of 13 × 13 pixels. For each grid cell on feature map, 5 anchors and corresponding confidence scores are proposed (yellow boxes in Fig.1). Each anchor corresponds to a predict box on the original image, and confidence scores are defined as Eq. (1):

$$score_{confidence} = Pr(Object) \times IoU \quad (1)$$

Here, Intersection-over-Union (IoU) is defined as Eq. (2):

$$IoU = \frac{aera(B_p \cap B_g)}{aera(B_p \cup B_g)} \quad (2)$$

where $aera(B_p \cap B_g)$ represents the intersection of the vehicle proposal box and ground truth box, and $aera(B_p \cup B_g)$ denotes their union. If a predict box has the IoU bigger than 0.7 with the ground truth box, it's considered as the positive samples for training. These scores represent both the probability of an object appearing in the box and how well the predicted box fits the object.

III. EXPERIMENT

In this section, we evaluate three models for vehicle detection in UAV images. In our work, the augmented UAV data set we collected is divided into two independent subsets: 20604 images for training and validation, and 1374 images for test. Moreover, the images from Stanford Drone Dataset nexus scene are used for estimating the transfer ability of these models. All experiments are implemented based on the deep learning framework Caffe [28], and executed on a PC with an Intel single Core i7 CPU, NVIDIA GTX-1060 GPU (6 GB GPU memory), and 16 GB RAM. The PC operating system was Ubuntu 14.04.

Firstly, we evaluate the performance of YOLO, SSD, YOLOv2 on our collected UAV dataset. The recall rates of three methods under different IoU thresholds are plotted in Fig.3. The plot shows that with IoU thresholds increasing, the recall curve of YOLO and SSD drop more sharply than YOLOv2. Moreover, YOLOv2 has the best recall under various IoU.

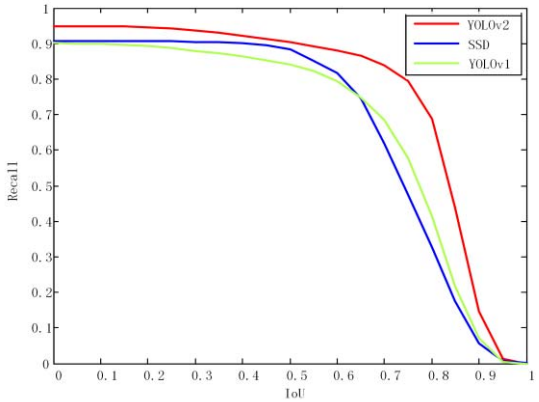

Fig.3. Recall versus IoU of three methods on the UAV dataset we collected.

TABLE Ⅰ and Fig.4 (a)-(i) show the quantitative comparison results measured by AP values, average running time per image, and detection results. As can be seen from them, 1) YOLOv2 achieved the best performance in terms of AP, 2) with the same computation cost, YOLOv2 is considerably faster. It can detect an image (with the size of 640×480 pixels) within 0.048 s with the mAP of 77.12%. Therefore, it can be used for real-time vehicle detection in UAV videos on better GPU. 3) As shown in Fig.4, using those training samples obtained in simple scenes, YOLOv2 can detect most of the vehicles in more complex and unseen scenes. Since there is no training sample of the bus, most of the buses are missing detection. These results demonstrating the effectiveness of our method.

TABLE I. AP AND RUNNING TIME ON UAV DATASET WE COLLECTED

Method	mAP	Average running time per image (second)
SSD	72.95%	0.055
YOLO	67.99%	0.056
YOLOv2	77.12%	0.048

Secondly, to further demonstrate the robustness of these method, we also evaluate them on Stanford Drone Dataset. TABLE Ⅱ and Fig.4 (j)-(l) show the AP values, average running time per image and detection results of three different methods. These results show that YOLOv2 has the best transfer ability. It can detect an image (with the size of 1330× 1947 pixels) within 0.052 s with the mAP of 68.00%, using the model trained on the UAV image dataset we collected. It can be observed that YOLOv2 can detect most of the objects successfully, demonstrating the transfer ability of this method. It also demonstrates that our data preparation method is very useful. Therefore, it has great potential for wide field application.

TABLE II. AP AND RUNNING TIME ON STANFORD DATASET

REFERENCES

[1] Moranduzzo T, and F Melgani, "A SIFT-SVM method for detecting cars in UAV image" Geoscience and Remote Sensing Symposium IEEE, 2012:6868-6871.

[2] Moranduzzo T, Melgani F, "Automatic Car Counting Method for Unmanned Aerial Vehicle Images," IEEE Transactions on Geoscience & Remote Sensing, 2014, 52(3):1635-1647.

[3] Moranduzzo T, Melgani F, "Detecting Cars in UAV Images With a Catalog-Based Approach," Geoscience & Remote Sensing IEEE Transactions on, 2014, 52(10):6356-6367.

Method	mAP	Average running time per image (second)
SSD	56.25%	0.058
YOLO	42.31%	0.059
YOLOv2	68.00%	0.052

[4] Zhao T, Nevatia R, "Car Detection in Low Resolution Aerial Image," Image and Vision Computing, 2001, 21(8):693-703.

[5] H, Chellappa R, Rosenfeld A, "Performance analysis of a simple vehicle detection algorithm," Image & Vision Computing, 2002, 20(1):1-13

Fig.4. Detection results of YOLOv2 on our collected UAV images and Stanford Drone Dataset, red boxes denote correct localization, green boxes denote missing detection, and blue boxes denote wrong detection

[6] Kluckner S, Pacher G, Grabner H, et al, "A 3D Teacher for Car Detection in Aerial Images," International Conference on Computer Vision. IEEE, 2007:1-8.

[7] Holt A C, Seto E Y W, Rivard T, et al, "Object-based Detection and Classification of Vehicles from High-resolution Aerial Photography," Photogrammetric Engineering & Remote Sensing, 2009, 75(7):871-880.

[8] Shao W, Yang W, Liu G, et al, "Car detection from high-resolution aerial imagery using multiple features," International Geoscience and Remote Sensing Symposium. IEEE, 2012:4379-4382.

[9] Chen X, Xiang S, Liu C L, et al, "Vehicle Detection in Satellite Images by Hybrid Deep Convolutional Neural Networks," IEEE Geoscience & Remote Sensing Letters, 2014, 11(10):1797-1801.

[10] Leitloff J, Rosenbaum D, Kurz F, et al, "An Operational System for Estimating Road Traffic Information from Aerial Images," Remote Sensing, 2014, 6(11):11315-11341.

[11] Chen, Xueyun, et al, "Vehicle Detection in Satellite Images by Hybrid Deep Convolutional Neural Networks," IEEE Geoscience & Remote Sensing Letters 11.10(2014):1797-1801.

[12] Z. Chen, C. Wang, C. Wen, and X. Teng, "Vehicle detection in high resolution aerial images via sparse representation and super pixels," IEEE Transactions on Geoscience & Remote Sensing, vol. 54, no. 1, pp. 1–14, 2015.

[13] W. Diao, X. Sun, X. Zheng, and F. Dou, "Efficient saliency-based object detection in remote sensing images using deep belief networks," IEEE Geoscience & Remote Sensing Letters, vol. 13, no. 2, pp. 1–5, 2016.

[14] Uijlings, J. R. R., et al. "Selective Search for Object Recognition," International Journal of Computer Vision, 104.2(2013):154-171.

[15] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-based convolutional networks for accurate object detection and segmentation," IEEE Transactions on Pattern Analysis & Machine Intelligence, vol. 38, no. 1, pp. 1–1, 2015.

[16] R. Girshick, "Fast r-cnn," Computer Science, 2015.

[17] Ammour N, Alhichri H, Bazi Y, Benjdira B, Alajlan N, Zuair M. Deep Learning Approach for Car Detection in UAV Imagery. Remote Sensing. 2017; 9(4):312.

[18] Kuo, Weicheng, B. Hariharan, and J. Malik, "DeepBox: Learning Objectness with Convolutional Networks," ICCV, (2015):2479-2487.

[19] Ren, Shaoqing, et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," IEEE Transactions on Pattern Analysis & Machine Intelligence (2015):1-1.

[20] Gidaris, Spyros, and N. Komodakis, "Attend Refine Repeat: Active Box Proposal Generation via In-Out Localization," (2016).

[21] Redmon, Joseph, et al, "You Only Look Once: Unified, Real-Time Object Detection," (2015):779-788.

[22] Liu, Wei, et al, "SSD: Single Shot MultiBox Detector," Computer Science, 2015.

[23] Redmon, Joseph, and A. Farhadi, "YOLO9000: Better, Faster, Stronger," (2016).

[24] He, Kaiming, et al, "Deep Residual Learning for Image Recognition," IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[25] Henriques, João F., et al, "Exploiting the Circulant Structure of Tracking-by-Detection with Kernels," Lecture Notes in Computer Science7575.1(2012):702-715.

[26] Computational vision and Geometry lab, Stanford Drone Dataset. Available online: http://cvgl.stanford.edu/projects/uav_data/.

[27] Ioffe, Sergey, and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," Computer Science (2015).

[28] V. Turchenko and A. Luczak, “Caffe: Convolutional architecture for fast feature embedding,” Eprint Arxiv, pp. 675–678, 2014.