Real-Time Drone Detection Using Deep Learning Approach

Manjia Wu[1(✉)], Weige Xie[1,2], Xiufang Shi[1,2], Panyu Shao[1,2], and Zhiguo Shi[2]

[1] State Key Laboratory of Industrial Control Technology, Zhejiang University, Hangzhou, China
manjiawu@gmail.com
[2] College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China

Abstract. The arbitrary use of drones poses great threat to public safety and personal privacy. It is necessary to detect the intruding drones in sensitive areas in real time. In this paper, we design a real-time drone detector using deep learning approach. Specifically, we improve a well-performed deep learning model, i.e., You Only Look Once, by modifying its structure and tuning its parameters to better accommodate drone detection. Considering that a robust detector needs to be trained using a large amount of training images, we also propose a semi-automatically dataset labelling method based on Kernelized Correlation Filters tracker to speed up the pre-processing of the training images. At last, the performance of our detector is verified via extensive experiments.

Keywords: Drone detection · Deep learning · Visual detection

1 Introduction

In recent years, with the continuous development of related technologies, drone companies such as DJI, Parrot, and 3DRobotics are rapidly developing. And because of the low price and user-friendly operation, drones have been widely used in both military and civilian areas and meet an explosive growth of consumption. According to the Federal Aviation Administration (FAA), the purchases of drones were 1.9 million in 2016 and may increase to 4.3 million by 2020 [1].

However, the rapid development and widespread application of drones also bring various hidden dangers, such as public safety, personal safety, personal privacy, and so on. The occurrences of accidents caused by the illegal fly of drones become more frequent. Therefore it is necessary to regulate the fly of drones. A few companies like DJI, set up no-fly zones to ensure the safety in some sensitive areas, such as airports, prisons. But the effect of no-fly zones is very limited. It is therefore significant to implement real-time drone detection to give warning accurately in time.

L. Meng and Y. Zhang (Eds.): MLICOM 2018, LNICST 251, pp. 22–32, 2018.
https://doi.org/10.1007/978-3-030-00557-3_3

Many existing techniques, e.g., radar, radio frequency, acoustic [2–4] and optical sensing techniques can be used for drone detection [5,6]. Because of high detection accuracy and long effective range, video-based detection has attracted many research interests [7,8] and has great potential for drone detection. In our work, we employ video-based detection based on deep learning approach which is a powerful tool in the computer vision area.

Drone detection is essentially an object detection problem. In early years, video-based object detection is by extracting discriminant features such as Local Binary Pattern (LBP) [9], Scale Invariant Feature Transform (SIFT) [10], Histogram of Oriented Gradient (HOG) [11] and Speeded Up Robust Features (SURF) [12] then using these features to train the detector. In 2012, Krizhevsky et al. [13] showed the amazing power of the convolutional neural network (CNN) in the ImageNet grand challenge. Since then, the developments and applications of deep learning methods increase rapidly. There are several variants in CNNs such as the R-CNN [14], SPPNet [15] and Faster-RCNN [16]. Since these networks can generate highly discriminant features, their performances are far beyond the traditional object detection techniques. In 2015, Redmon et al. [17] raised up a new approach of object detection called You Only Look Once (YOLO), which takes object detection as a regression problem with a single neural network. YOLO can achieve very fast detection, i.e., 67 frames per second (FPS) and can realize real-time detection. Moreover, compared with many detectors based on deep learning, YOLO has much lower requirement on computer configuration, which needs normally 4 GB GPU-RAM or even 1GB GPU-RAM for the tiny model. Therefore, we choose YOLO as the model in the detection of drones.

In our work, we develop a drone dataset called Anti-Drone Dataset, including 49 videos. And we use Kernelized Correlation Filters (KCF) tracker [18] to label the videos in the dataset without manually labelling which saves a lot of time and avoids manual errors. Then we improve the traditional YOLO model by adjusting key parameters such as the resolution of input image and the dimension of anchor box. Afterwords, we train the detector using both Anti-Drone Dataset and manually labelled dataset from the Internet and evaluate the detector's performance on these two drone datasets. Apart from that, we deploy our drone detector in Yuquan campus of Zhejiang University and conduct experiments to verify the feasibility of the detector in real world.

The rest of this paper is organized as follows. The drone datasets and labelling method are introduced in Sect. 2. The detection model and its improvement are described in Sect. 3. Experiments are shown in Sect. 4. Conclusion is given in Sect. 5.

2 Drone Dataset and Labelling

The establish of drone detector using YOLO requires labelled drone dataset for training. In the following we will introduce the drone datasets and the labelling method.

2.1 Drone Dataset

In this paper, we utilize two datasets for training: one is a public-domain drone dataset, the other one is a dataset built by ourselves.

Public-Domain Drone Dataset. For our training and testing work, we use a public-domain drone dataset, USC Drone Dataset [19], which consists of 30 YouTube video sequences and captures different drone models with various appearance. And the dataset is rich in diversity with different environments including both indoor and outdoor environments, such as grassland, courtyard, warehouse and so on. Some samples in the dataset is shown in Fig. 1. These video clips have a frame resolution of 1280×720 and their duration time is about one minute. Some video clips contain more than one drone, while not all of them are marked out. Furthermore, some shoots are discontinuous.

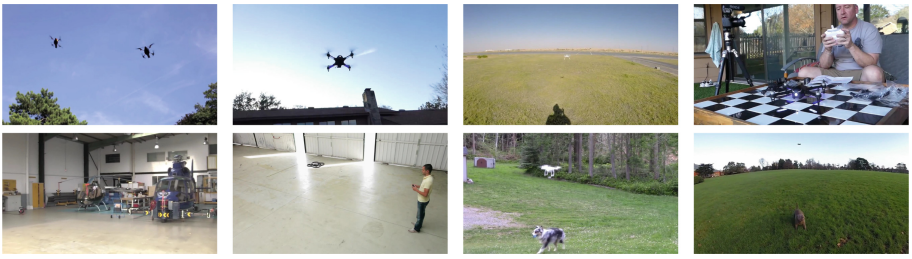


Fig. 1. Some samples from USC drone dataset

Anti-drone Dataset. A wide variety dataset is one of the essential conditions to training a robust detector based on neural-network. And as the number of existed drone dataset is very limited, we build a dataset and label it. We shoot 49 experimental videos by HIKVISION DS-2DF7330IW Network PTZ camera, as shown in Fig. 2(a). In these videos, there are three drone models, MAVIC PRO, PHANTOM 2 and PHANTOM 4, as shown in Fig. 2(b), (c), (d).

(a) HIK camera (b) MAVIC PRO (c) PHANTOM 2 (d) PHANTOM 4

Fig. 2. Equipment used in the experiment

The frame resolutions are 2048×1536 and 1024×768 respectively in main stream and sub stream. The frame rate is 24 FPS. To shoot these video clips,

we consider comprehensively about shooting backgrounds, camera angles and magnifications, weather conditions, day or night. The videos are designed to capture real-world drone attributes such as fast motion, extreme illumination, small size and occultation. Several examples are shown in Fig. 3.

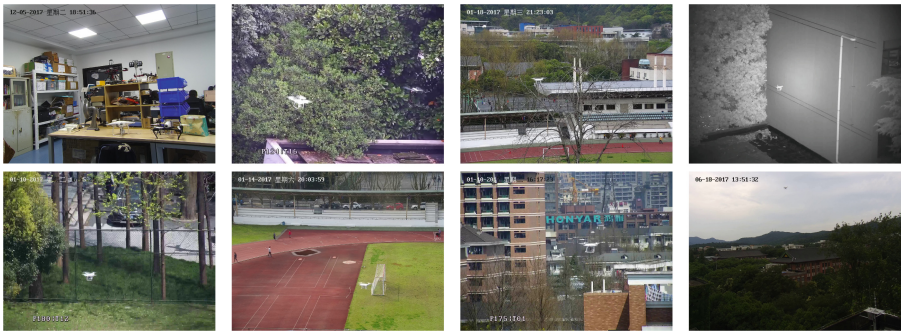


Fig. 3. Anti-drone dataset

2.2 Dataset Labelling

For drone detection, labelling the drones' locations especially in the videos is a labor intensive and tedious task. And the accuracy is also affected by manual errors. Motivated by the above observations, we use the KCF tracking algorithm to semi-automatically label these videos. The KCF tracker is widely used tracker and achieves very appealing tracking performance both in accuracy and speed because of the introduction of ridge regression and circulant matrix theory, which can efficiently enhance the discriminative ability of correlation filter-based algorithm.

Our basic idea is labelling the drone locations in the first frame and then obtaining next frames automatically by using KCF tracking algorithm. Our goal is to label a large number of videos in a short time with high accuracy. Due to the characteristics of the tracking algorithm, we have cut videos to ensure the drones appear all the time. With this method, we have labelled 60 videos without marking every frame manually and saved a lot of time. And in order to balance the number of videos in different background scenes and shooting conditions, we select the same number of frames for each scenario.

3 Real-Time Detection Model

In this section, we present a deep convolution neural network model, YOLOv2 [20], which is the state-of-the-art on standard detection tasks. Then we propose a new model for real-time drone detection by modifying the structure and tuning parameters of YOLOv2, which makes the model better adapt to the drone detection.

3.1 YOLOv2

YOLOv2 frames object detection as a regression problem to spatially separate bounding boxes and associate class probabilities, which is different from prior object detection repurposing classifiers to perform detection. YOLOv2 only uses a single neural network to predict bounding boxes and class probabilities directly from the whole image in one round evaluation. Since the whole detection pipeline is a single network, it can be optimized end-to-end directly on detection performance, which makes the detection extremely fast. The flowchart of YOLOv2 is shown in Fig. 4.

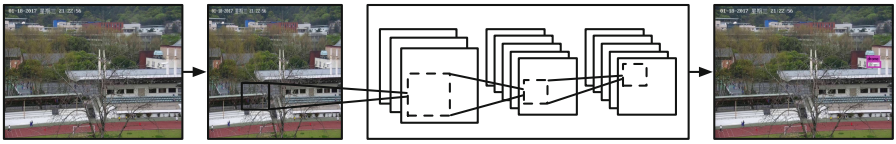


Fig. 4. YOLO detection system. (1) resize the input image to 416×416, (2) run a single convolutional network on the image, (3) threshold the resulting detections by non-max suppression.

Network Design. The network in our model is based on Darknet19 [20], which uses mostly 3×3 filters following with 1×1 filters for compressing features. The whole network for detection is obtained by removing the last convolutional layer of Darknet19, adding three 3×3 convolution layers with 1024 filters and 1×1 convolution layer with 30 filters at the end of network. There is also a passthrough layer from the last $512 \times 3 \times 3$ layer to the last but one convolution layer, which enables the model to have fine grain features, as shown in Fig. 5.

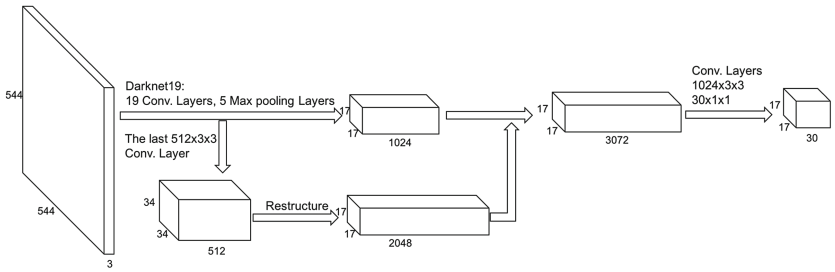


Fig. 5. The passthrough layer of the network. We combined the last $512 \times 3 \times 3$ convolution layers with the last but one convolution layer to make the model have fine grain feature.

Final Prediction of the Network. YOLOv2 models detection as a regression problem. Firstly, YOLOv2's convolutional layers downsample the image. With a 416×416 input image, we get an output feature map of 13×13. Secondly, sliding window sampling is performed on the feature map, and each center predicts k anchor boxes with different sizes and aspect ratios. Thirdly, the anchor box simultaneously predicts the class probability and coordinates. So the prediction should be a $13 \times 13 \times k \times (classes + coordinates + confidence)$ tensor. As drone detector only detects one class object and predicts 5 anchor boxes for every center, the final prediction of model is $13 \times 13 \times 30$ as shown in Fig. 6.

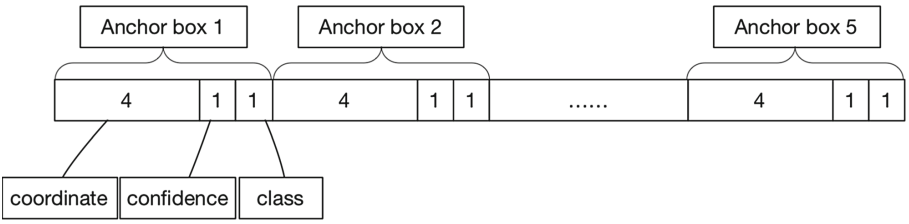


Fig. 6. The final prediction of model for drone detection

As shown in Fig. 6, the network predicts 5 bounding boxes at each cell. For each bounding box, it predicts 4 coordinates, t_x, t_y, t_w, t_h, and confidence reflecting how confidently the box contains an object and how accurately the box locates.

$$confidence = Pr(Object) \times IOU(box, object) \tag{1}$$

If no object exists in certain cell, $Pr(Object) = 0$ and $confidence = 0$. Otherwise if an object exists, $Pr(Object) = 1$ and confidence score equal to the IOU between the predicted box and the corresponding cell.

3.2 Improvement

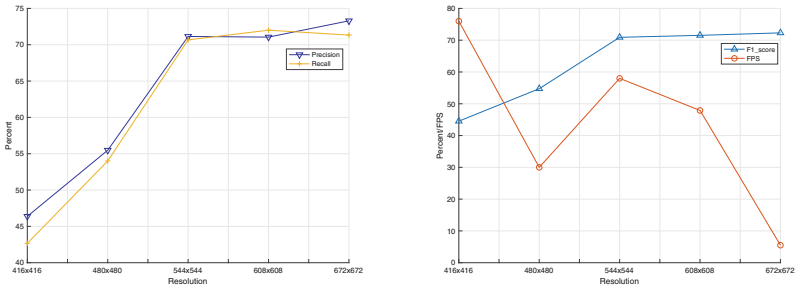
Resize Image. The purpose of our real-time drone detection is to detect drones for early warning. For the sake of real-time and accurate detection of drones, we resize images to higher resolution to adapt to small objects.

As objects tend to occupy the center of the image, it would be better to have a single location right at the center to predict these objects instead of four locations nearby. So we need an odd number of locations. In the original network of YOLO, the convolutional layers downsample the image by a factor of 32. With a 416×416 input image, we get an output feature map of 13×13. Therefore, we modify the network to make the input image resized as 416×416, 480×480, 544×544, 608×608 or 672×672, while ensuring an odd number of feature map locations. However, the higher resolution the feature map is, the more time may be spent to detect per image. We evaluate different resolutions with precision and recall, shown in Fig. 7(a). The valid dataset is composed of images with small

object which is smaller than 56×32 in full image of 2048×1536. And in order to consider both precision and recall, we use $F1\_score$ to value the detection performance.

$$F1\_score = \frac{1}{2}(Precision + Recall) \tag{2}$$

Processing speed is shown in Fig. 7(b). We choose 544×544 as a good tradeoff between time complexity and high $F1\_score$.



(a) Precision and recall (b) F1_score and FPS

Fig. 7. Resize resolution. We resize input image in different resolution to get better detection performance for small target and guarantee the real-time.

Anchor Box Prior Dimension. The second improvement of our model is to adjust the pre-defined anchor box's dimensions to make the network more adaptive to drone detection. We use k-means++ proposed by Arthur et al. [21] to cluster the box dimensions in our dataset. As our goal is to get better IOU scores which is effected by box dimensions and locations, the distance function is

$$d(box, centroid) = 1 - IOU(box, centroid) \tag{3}$$

We try different numbers of clusters with k-means++ algorithm. The Avg_IOU corresponding different numbers is shown in Fig. 8. From Fig. 8, we can see that as the amount of clusters increases, the ascension of the curve becomes less. And it is obvious that more anchor boxes means more complex the model is. Therefore, in order to keep balance between model complexity and high recall, we consider that the inflection point of the curve is the optimal number of clusters. We choose $k = 5$ in our detection model and the anchor box dimensions are (3.2896, 3.5241), (0.7357, 0.7050), (7.4209, 5.8418), (1.6510, 1.6501), (11.2494, 11.4231) for 544×544 feature map.

4 Experiments

4.1 Experimental Environment

We implement network training and evaluation on a computer server. The configuration of this server is shown in the first three rows of Table 1. And the

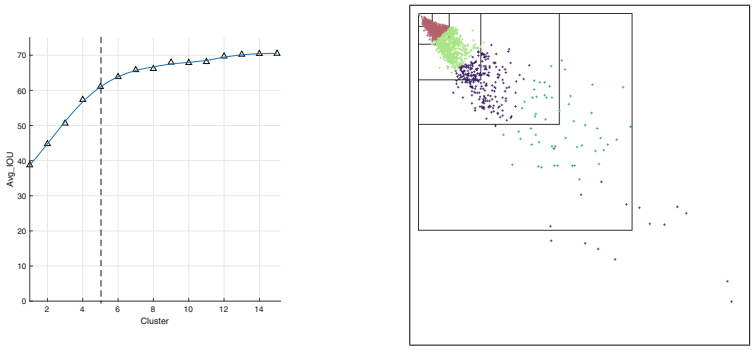


Fig. 8. Cluster anchor boxes. We cluster bounding boxes in dataset to get better anchor box dimension priors. The left image shows the curve of average IOU with various numbers for clusters. And the right image shows anchor boxes when $k = 5$.

configuration of our drone detection system is shown in the last three rows of Table 1.

Table 1. Experiment platform configuration

Computer server	GPU	2 × TITAN X Pascal
	CPU	2 × Intel(R) Xeon(R) CPU E5-2640 v4 @ 2.40 GHz
	System	64-bit, Ubuntu 14.04.5
Drone detection system	GPU	NVIDIA GeForce GTX 1050
	CPU	Inter(R) Core(TM) i7-7700K CPU @ 4.20 GHz
	System	64-bit, Windows 10

4.2 Pre Training

The pre-training of the network is an important part of object detection. But pre-training will cost a lot of time and have high requirements for computer configuration. Therefore, we use the weights of Darknet19 which has been pre-trained with the standard ImageNet 1000 class classification dataset for 160 epochs.

4.3 Training for Detection

We train the network using USC Drone Dataset from the Internet and Anti-Drone Dataset labelled by KCF tracker. We divide datasets by 3:1 for training and testing. And we train the network with a starting learning rate of 10^{-3} and change it when the iteration step is 100, 25000 or 35000. Then we set momentum weight decay as 0.0005 which prevents overfitting. As attitudes and conditions of drones are varying from moment to moment, we generate new images for training

in every iteration with $angle = 5$, $saturation = 1.5$, $exposure = 1.5$, $hue = .1$, which is in order to gain more robust model. These weights are evaluated by precision and recall. The results with different iterations shown in Fig. 9. As the curve shows, we choose the weights when the iteration is 3600 with recall of 85.44% and precision of 88.35%.

4.4 Evaluation

We evaluate the detector by the precision-recall curve. Precision is the fraction of detected region proposals which are true positive. Recall is the fraction of true positive which are detected. The effectiveness of KCF labelled dataset is illustrated in Fig. 9. In the figure, we compare the performances of the detector trained by only Public-Domain drone dataset and the detector trained by the dataset which combines the USC Drone Dataset with Anti-Drone Dataset. We can see that when we set the threshold as 0.2, the detector trained by the combined dataset is better than USC only detector. The IOU of the detector trained by the combined dataset is 60.59%, which is however a bit lower than that of USC only (62.44%).

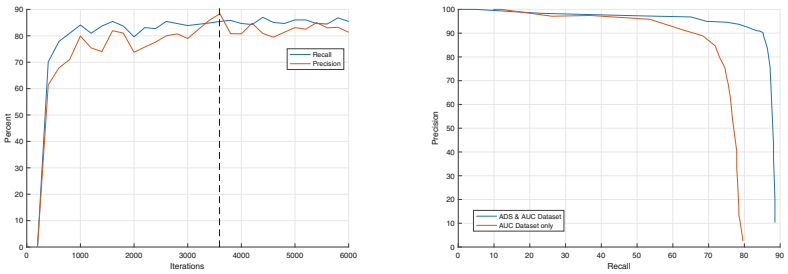


Fig. 9. Experiment results. The left figure shows the change of precision and recall as iterations increase. The right figure shows comparison of the drone detector trained by only USC dataset and combined with KCF labelled dataset.

Verification with Practicality of Detector. In order to verify that our detector can be used in an actual system while ensuring real-time performance and with low system cost, we implement detector on different GPU-RAM configurations. With GPU-RAM 4 GB, the processing speed is 33 FPS. With GPU-RAM 2 GB, the processing speed can achieve 19 FPS. Although it is a bit slower than the video frame rate (24 FPS), it's still useful as we can detect drones by frame skip. Therefore, our detector can be applied to an actual detection system, which can realize real-time detection with low system cost.

5 Conclusion

The video-based real-time drone detection using deep learning approach was implemented in this work. We firstly developed a dataset, which is semi-automatically labelled by KCF tracker instead of manually labelling. Then we

improved the YOLOv2 model by modifying the structure with the resolutions of input images and adjusting parameters of the dimension of anchor box. The designed drone detector is trained using both our own Anti-Drone Dataset and the public domain dataset from the Internet. Extensive experiments showed that the detector can achieve real-time detection with high accuracy.

Acknowledgments. This work was supported by NSFC under Grant 61772467, Zhejiang Provincial Natural Science Foundation of China under Grant LR16F010002, 973 Project under Grant 2015CB352503, the Fundamental Research Funds for the Central Universities (2017XZZX009-01), and China Postdoctoral Science Foundation funded project.

References

1. Wargo, C., Snipes, C., Roy, A., Kerczewski, R.: UAS industry growth: forecasting impact on regional infrastructure, environment, and economy. In: 2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC), pp. 1–5. IEEE (2016)
2. Chang, X., Yang, C., Wu, J., Shi, X., Shi, Z.: A surveillance system for drone localization and tracking using acoustic arrays. In: 2018 IEEE 87th Vehicular Technology Conference (2018)
3. Chang, X., Yang, C., Shi, X., Li, P., Shi, Z., Chen, J.: Feature extracted DOA estimation algorithm using acoustic array for drone surveillance. In: 2018 10th IEEE Sensor Array and Multichannel Signal Processing Workshop (2018)
4. Yang, C., Wu, Z., Chang, X., Shi, X., Wo, J., Shi, Z.: DOA estimation using amateur drones harmonic acoustic signals. In: 2018 10th IEEE Sensor Array and Multichannel Signal Processing Workshop (2018)
5. Shi, X., Yang, C., Xie, W., Liang, C., Shi, Z., Chen, J.: Anti-drone system with multiple surveillance technologies: architecture, implementation, and challenges. IEEE Commun. Mag. **56**(4), 68–74 (2017)
6. Chen, J., Kang, H., Wang, Q., Sun, Y., Shi, Z., He, S.: Narrowband internet of things: implementations and applications. IEEE Internet Things J. **4**(6), 2309–2314 (2017)
7. Sevil, H.E., Dogan, A., Subbarao, K., Huff, B.: Evaluation of extant computer vision techniques for detecting intruder sUAS. In: 2017 International Conference on Unmanned Aircraft Systems (ICUAS), pp. 929–938. IEEE (2017)
8. Hwang, S., Lee, J., Shin, H., Cho, S., Shim, D.H.: Aircraft detection using deep convolutional neural network in small unmanned aircraft systems. In: 2018 AIAA Information Systems-AIAA Infotech@ Aerospace, p. 2137 (2018)
9. Ojala, T., Pietikainen, M., Harwood, D.: Performance evaluation of texture measures with classification based on kullback discrimination of distributions. In: Proceedings of the 12th IAPR International Conference on Pattern Recognition, 1994. Vol. 1-Conference A: Computer Vision & Image Processing, vol. 1, pp. 582–585. IEEE (1994)
10. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. **60**(2), 91–110 (2004)
11. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR, vol. 1, pp. 886–893. IEEE Computer Society (2005)

12. Bay, H., Ess, A., Tuytelaars, T., Van Gool, L.J.: Speeded-up robust features (SURF). Comput. Vis. Image Underst. **110**(3), 346–359 (2008)
13. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS, pp. 1106–1114 (2012)
14. Girshick, R.B., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR, pp. 580–587. IEEE Computer Society (2014)
15. He, K., Zhang, X., Ren, S., Sun, J.: Spatial pyramid pooling in deep convolutional networks for visual recognition. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8691, pp. 346–361. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10578-9_23
16. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: NIPS, pp. 91–99 (2015)
17. Redmon, J., Divvala, S.K., Girshick, R.B., Farhadi, A.: You only look once: unified, real-time object detection. In: CVPR, pp. 779–788. IEEE Computer Society (2016)
18. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: High-speed tracking with kernelized correlation filters. IEEE Trans. Pattern Anal. Mach. Intell. **37**(3), 583–596 (2015)
19. Usc drone dataset. https://chelicynly.github.io/Drone-Project/
20. Redmon, J., Farhadi, A.: YOLO9000: better, faster, stronger. In: CVPR, pp. 6517–6525. IEEE Computer Society (2017)
21. Arthur, D., Vassilvitskii, S.: k-means++: the advantages of careful seeding. In: SODA, pp. 1027–1035. SIAM (2007)