SSD For Object Detection In Aerial Videos

Conference Paper · June 2019

CITATIONS
0

READS
90

4 authors, including:

Nguyen D. Vo
67 PUBLICATIONS **199** CITATIONS
SEE PROFILE

Khang Nguyen
Ho Chi Minh City University of Information Technology
25 PUBLICATIONS **38** CITATIONS
SEE PROFILE

SSD For Object Detection In Aerial Videos

Quynh M. Chung
University of Information Technology, VNU-HCM
Ho Chi Minh City, Vietnam
e-mail: 15520712@gm.uit.edu.vn

Nguyen D. Vo
University of Information Technology, VNU-HCM
Ho Chi Minh City, Vietnam
e-mail: nguyenvd@uit.edu.vn

Tien D. Le
University of Information Technology, VNU-HCM
Ho Chi Minh City, Vietnam
e-mail: 15520881@gm.uit.edu.vn

Khang Nguyen
University of Information Technology, VNU-HCM
Ho Chi Minh City, Vietnam
email: khangnttm@uit.edu.vn

***Abstract*—Recent development of drones, along with the help of advanced object detection algorithms have made ways for a myriad of potential applications. However, due to the lack of thorough datasets and the constant visual change, the task remains challenging. In this paper, we conduct a comprehensive evaluation using Single Shot Multi-Box Detector (SSD), one of the state-of-the-art object detectors, on VisDrone2018, a recently published large-scale benchmark collected by drones. Initial results are very low for SSD because most of the objects are small-scaled and there is a significant class imbalance. Thus, we evaluate on four different cases to see the effects between training entirely on the dataset and on four classes (e.g. car, van, truck, and bus), in addition to training with and without confusing information. We also propose hyperparameters that results in an improved model.**

Keywords: drone; object detection in video; single deep neural network

I. INTRODUCTION

Object detection is a task that involves detecting one or multiple instances of a known class on an image. The detection results are usually in the form of information including the bounding box, the class of the object, and, sometimes, other information must also be retrieved. In a video, object detection is more difficult. The algorithms must be able to make quick and precise predictions in real time. Usually, the ratio of object size to the video resolution is very small. Because of this, there is a trade-off between accuracy and inference time in these algorithms.

Among the cutting-edge technology, drones are one of the newest inventions that can be used as surveillance cameras. Time-series data or aerial images received from camera-mounted drones can vary greatly in terms of scales, perspectives, weather conditions, etc.

Current approaches to the task include one-stage and two-stage detectors. One-stage detectors such as SSD [1] or YOLO [2] rely on a set of predefined locations to detect on an image. Meanwhile, two-stage detectors such as Faster R-CNN [3] use a separate part to search for interesting locations, then classifies the objects in another part. Experimentally, the first technique proves to be more reliable in terms of speed, whereas the second gives higher accuracy. Nonetheless, SSD has a fair balance between accuracy and speed on PASCAL VOC 2007 [4] and 2012 [5].

Thus, in this paper, we experiment SSD on a new dataset, VisDrone2018 [6], collected by drones. Because of the difficulty of the dataset, the evaluation is divided into four different cases. The two first are evaluated using all classes. The two last are evaluated using four classes with bigger objects, e.g. car, van, truck, and bus. For each of the two, one uses all information, and one uses filtered information. Finally, we propose hyperparameters that can raise the performance of the model.

The paper is organized as follows: in Section 2, we briefly describe current solutions to object detection using deep learning. In Section 3, we discuss in more detail about SSD. Section 4 describes experiments and results. Finally, the paper is concluded in Section 5.

II. RELATED WORK

In this section, we present a brief summary of a few excellent approaches to a quick and accurate object detector, along with their pros and cons.

A. Faster R-CNN

Instead of relying on a computationally expensive method such as Selective Search like R-CNN [7] and Fast R-CNN [8], Faster R-CNN [3] uses an internal network to generate all regions of interest (ROIs). The network, called Region Proposal Network (RPN), takes the output from the feature map created by using a convolutional neural network (CNN), making it a

component inside the whole architecture. This has drastically increased the speed of the model and reduced the inference time per image. With *k* guesses per box, the network returns 4*k* coordinates of bounding boxes and 2*k* scores which determines whether they contain an object or not. These, along with the feature map, are then mapped together to pick out the potential regions which are then gone through a ROI Pooling layer to reshape the output to a fixed size. After that, two fully connected layers are added and act as a classifier where a softmax layer is used to determine the object class and a linear regressor helps refining the bounding box. Faster R-CNN also applies non-maximal suppression (NMS) to filter redundant overlapping ones based on their class score. It has achieved 73.2% [3] on PASCAL VOC 2007, but only 7 FPS [3] due to too many candidate regions. An overview of the model is shown in Figure 1.

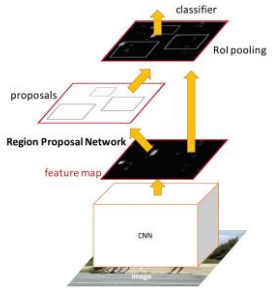

Figure 1. Faster R-CNN architecture [3].

B. *YOLO*

Recognizing the drawbacks of two-stage detectors, You Only Look Once (YOLO [2]) adopts another approach by building a one-stage detector. Unlike R-CNN family, the proposed method skips the first stage, wherein a model decides which region to make predictions on, and instead selects predefined locations to look for objects. This significantly boosts the overall speed of the process. YOLO first splits an image into *S*×*S* cells. Then, for each cell that has an object's center inside, it predicts *B* bounding boxes, a confidence score determining if there is an object, and probability of *C* classes. The bounding boxes have a spatial limitation, also known as one-object rule, to prevent them from repeating each other. Because of this, YOLO cannot perform well on objects in clusters. At the end, the final convolutional layer has a tensor shape of $S \times S \times (5B+C)$, which is the output shape for a pre-train image classification CNN. Finally, the whole architecture is then trained end-to-end with the dataset, leading to further accuracy improvements. The model also uses non-maximum suppression (NMS) to remove any duplications that have a lower confidence. In Pascal VOC2007, YOLO has a remarkable speed with 45 FPS but has a moderately good accuracy with 63.4% mAP [2]. Figure 2 illustrates the idea of YOLO model.

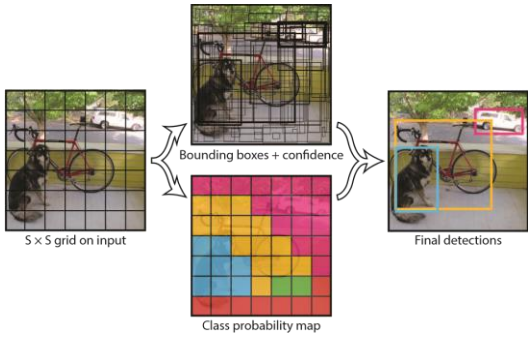

Figure 2. The YOLO model [2].

Besides YOLO, SSD [1] is also a good object detection model in terms of performance. In PASCAL VOC 2007, SSD300 has impressively achieved 74.3% mAP at 46 FPS, outranking all others. Additionally, it adopts a scaling mechanism to deal with objects at different sizes which can be easily configured (see Figure 4). Meanwhile, data taken from camera-mounted drones is always in constant change regarding perspectives, lighting, directions, and object scaling, etc. Thus, we are motivated to evaluate SSD on the new challenging dataset as well as finding hyperparameters that improve the model.

III. SSD

SSD [1] is one of the first object detection models to utilize one-stage strategy, resulting in an accurate, high speed detector.

A. *Base network VGG-16*

SSD is built on top of VGG-16 base network [1]. VGG-16 [9] is a deep convolutional neural network (DCNN) proposed by K. Simonyan and A. Zisserman [9] that focuses on simplicity and depth. The model uses 16 convolutional layers with only 3×3 filters to extract features. As the model goes deeper, the number of filter doubles after each max-pooling layer. Noticeably, the convolutional layers of the same type go in combination. This creates an effect of that of a bigger filter size while preserving the benefits of smaller filter size. At the end, three fully connected layers follow with 4096 channels each. The last one contains 1000 channels for each class and is concatenated with a softmax layer to return the detection results. The model works well on classification and localization tasks and has achieved above 89% mAP in PASCAL VOC 2007 and 2012 datasets [9]. Although VGG-16 is not quite as fast as the newer ones, it has been reused in many models because of its valuable extracted features.

B. Model architecture

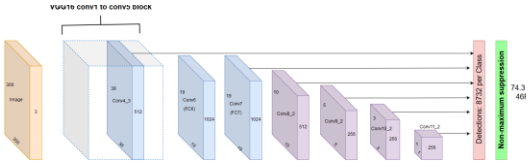

Figure 3. The model architecture of SSD [1].

SSD extends the pretrained VGG-16 [9] model (on ImageNet [10]) by adding new convolutional layers conv8_2, conv9_2, conv10_2, conv11_2 in addition to using the modified conv4_3 and fc_7 layers to extract useful features. Each layer is designed to detect objects at a certain scale in every cell using *k anchor boxes* (also known as *default boxes* or *prior boxes*), where 4*k* offsets *c* class probabilities are computed by using 3×3 filters. Thus, given a feature map with a size of *m*×*n*, the total number of filters to be used is *kmn*(*c*+4). The anchor boxes are chosen manually. Here, we use the original formula (1) to calculate anchor box scales at different levels, where s_{min} = 0.2 and s_{max} = 0.9.

$$s_k = s_{min} + \frac{s_{max} - s_{min}}{m-1}(k-1), k \in [1, m] \quad (1)$$

By combining a range of aspect ratios (1, 2, 3, 1/2, and 1/3) with the scales, and an additional larger scale, there are 6 anchor boxes per location. These layers represent a hierarchical feature pyramid which is used to efficiently predict objects at different scales.

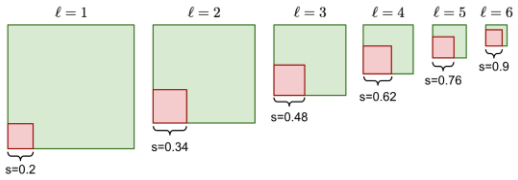

Figure 4. Illustration of an anchor box with aspect ratio 1 is scaled up along with the layer index *l* using s_{min} = 0.2 and s_{max} = 0.9. (source [11])

Figure 4 simplifies and shows how an anchor box is created given a feature map. In practice, there are up to *kmn* boxes per layer. The smallest scaling starts at the shallowest layer (to the left) which is supposed to detect smaller objects while deeper layers (to the right) are expected to search for bigger objects.

When training, SSD uses a fixed input size, usually 300×300 (8732 predictions) and 512×512 (24564 predictions). The bigger the input the more accurate the model but slower inference time. It also relies on a few data augmentation strategies, e.g. different scaling, random sub-region, etc. before feeding images to the network. This is important for SSD because small objects are a big problem to the model, and it needs to be substantially trained.

SSD adopts hard negative mining strategy. Because it makes a lot of predictions, there is a relatively large imbalance between foreground and background samples. As a result, the false detections can easily overwhelm the positive ones and affect the training. However, the model still needs to know what constitutes a bad detection. Therefore, SSD keeps the ratio of bad negatives to positives at about 3:1, where only the hard samples with low confidence are considered.

Lastly, SSD uses non-maximal suppression (NMS) at post-processing to leave out any detections that are too close and overlap each other. The method discards any boxes that have lower confidence score but have an IoU coefficient higher than a threshold (i.e. 0.5) with a better one. This reduces duplications and increases precision.

IV. EXPERIMENTS AND RESULTS

A. Dataset

VisDrone2018 is a large-scale benchmark dataset published in a paper [6]. The dataset provides detailed annotations of images and videos captured by drones. It consists of 10,209 still images and 179,264 frames split from 263 video clips. The data varies in environment, density, and object categories. Each annotation includes bounding box, unique identity, truncation, and occlusion (the proportion of which an object is covered). Additionally, ignored regions are also given, in which any objects that appear are omitted during the evaluation process. Most of the data is captured from the sky, hence the small scaling of the objects.

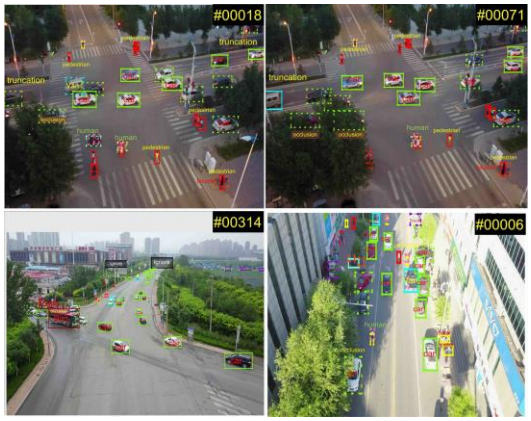

Figure 5. Some examples of annotated video frames in object detection in videos task. The colors indicate different object categories and the dashed bounding box depicts an occluded object [6].

The dataset was created for four different tasks, but we mainly focus on the second task: object detection in videos. The part dedicated to this task consists of 96 video clips, with 56 clips for training, 7 clips for validation and 33 clips for testing. Some examples are provided in Figure 5. There are 10 object categories: pedestrian (a person who either stands or walks), person, car, van, bus, truck, motor, bicycle, awning-tricycle, and tricycle. The last category (others) defines any objects that do not fall into the 10 classes. The highest resolution is 3840×2160.

B. *Evaluation cases*

After analyzing the benchmark, we find the dataset quite challenging as most of the objects are relatively small comparing to the image size and there is a significant class imbalance. Therefore, we believe training on a few selected categories can give a better result. We create two cases, one where the model uses the whole dataset, and one where only four classes with big objects: car, van, truck, and bus are used.

Furthermore, we believe some attributes are more suitable for tracking task than video object detection. Specifically, we find occlusion information misleading for the training process because any objects that are completely covered (e.g. a motor behind a truck, a car under an overpass) can make the training model confuse the covering object with the covered object. Therefore, in the evaluation, we define two new difficulty levels, as follows:

- Easy: entire dataset without heavily occluded (50-100%) examples.
- Hard: entire dataset.

We create sub-datasets for each of the evaluation cases, respectively.

The last case is our proposed hyperparameters to improve the model. Particularly, we extend the scaling range by decrease min ratio to 15% in order to cover more objects.

C. *Evaluation metrics*

Given two bounding boxes, a ground truth (actual) and a detection (predicted), we use the Intersection over Union (IoU), also known as Jaccard index, to calculate the similarity between the two boxes and score of the predicted box. The coefficient is then compared with a threshold to determine if the predicted box is correct or not.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cup B|} \quad (2)$$

Precision is the ratio of the number of correctly predicted instances to the number of retrieved actual instances, while recall is the fraction of positive instances over the total ground truth instances. In our evaluation, if multiple correct predictions overlap on same object, then only the first counts as true positive and the rest are false positives.

Average Precision (AP) is the mean value of all precisions taken at a certain recall value. We use the 11-point interpolated AP evaluation metric from PASCAL VOC 2007 [4], which calculates the AP at an equally spaced set of recall levels (0, 0.1, ..., 1). Each interpolated value is calculated by taking the maximum precision whose recall value is greater than the current one.

$$AP = \frac{1}{11} \sum_{r \in \{0, 0.1, \ldots, 1\}} \rho_{interp}(r)$$
$$\text{where } \rho_{interp}(r) = \max_{\tilde{r}:\tilde{r} \geq r} (\tilde{r}) \quad (3)$$

Mean Average Precision (mAP) is the average AP of all classes and a common metric to evaluate an object detection model. Additionally, in VisDrone2018 dataset, the bounding boxes that fall into ignore regions such that the IoU between the two is equal or greater than 0.5 are excluded. Moreover, the "others" class is also skipped.

Additionally, the dataset also introduces several evaluation criteria. Specifically, the metrics include $AP^{IoU=0.50:0.05:0.95}$, $AP^{IoU=0.50}$, $AP^{IoU=0.75}$, $AR^{max=1}$, $AR^{max=10}$, $AR^{max=100}$. Notably, the $AP^{IoU=0.50:0.05:0.95}$ is the main score to rank a method.

D. *Experiment details*

We evaluate all models on the same environment: Ubuntu OS 16.04.6 LTS, 6x Intel(R) Xeon(R) CPU @ 2.20GHz, 22.5 GB RAM, Nvidia Tesla K80 GPU.

Despite the statistics provided by the dataset [6], we notice many inconsistencies in the data we received. Unfortunately, the test set was unavailable, so we use the validation set as the test set. The analysis is shown in Figure 6.

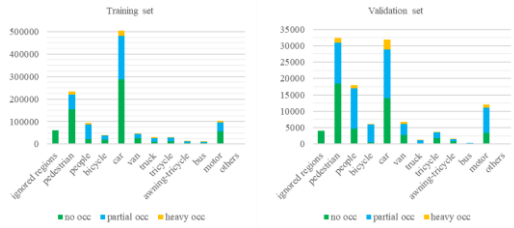

Figure 6. The number of objects per category with different levels of occlusion in the actual training and validation set.

For training phase, we use the training split given from the dataset. The split consists of 56 clips, with a total of 24,198 frames and 1,169,359 objects. We

remove ignored regions during the train process as we do not want the models to predict them. We train on 5 models as describe in Section 4B. Additionally, the training set for each case is modified such that only relevant classes are kept. All models are trained with 10 epochs, batch size 32, learning rate 10^{-4}, using SSD300. We also use the pretrained VGG16 on ILSVRC model as the starting point.

For testing phase, we use the validation split as test split. The split contains 7 clips including a total of 2,846 frames and 118,127 objects. The clips also include all classes defined in the training split.

For evaluation, we use the code provided by the SSD's author [1] to output the bounding boxes for each frame. Then, the boxes are taken to the VisDrone's toolkit to calculate all metrics and results. To measure inference time, we add time calculating function to the original code, excluding preprocessing time. The inference time is measured with a batch size of 32.

E. Results

Table 1 shows the results of SSD performance on the new dataset. We use the same hyperparameters as in the original code provided by the paper [1] and train the model with 10 epochs and batch size 32. As can be seen, all APs and mAP at IoU $\geq$ 0.50 have outranked the ones at IoU $\geq$ 0.75. $AP^{IoU=0.75}$ can be used to measure the performance of a model on objects that are proportionally small comparing to image size. Based on formula (2), if ground truth box is small, detection box must also be small to get a high result. Even a slight change in location can make the box negative. Meanwhile, most objects in the training data fall into those cases. This concludes why SSD underperforms on objects that are very small scaled.

Regarding the figures for car category, they all are the highest in comparison with the others. The reason why could be the total number of training samples for cars with about 500,000 examples (Figure 6). The large quantity of data has substantially provided the model with more variance including changes in size, shapes, and colors. Nonetheless, there is an uneven distribution of training samples, which makes the model give biased detections towards the well-trained classes and badly affects the accuracy of the others.

Furthermore, due to the characteristics of the dataset, some categories e.g. car and van, pedestrian and people, may have an undeniable visual similarity, as shown in Figure 7. However, car and pedestrian have more data to train than the others. Consequently, the model tends to mark the confusing objects as these classes when they are not. This hinders the APs of both classes.

(a) car (b) van (c) pedestrian (d) people

Figure 7. Some examples of visually similar objects with different classes, selected randomly from the training data.

Table 2 shows the results of SSD performance on four previously described cases. Configurations remain the same as in Table 1's. Looking at the $mAP^{IoU=0.5:0.05:0.95}$, training on four classes has improved mAP of four classes remarkably (about 0.32% with heavy occlusion and 0.8% without it). This is because when using the same method, training exclusively on fewer classes is much easier than on eleven classes. Additionally, we use the same epoch size, which means the four class cases have more time to fit on the training data, resulting a higher accuracy. Another key point is, comparing to the other classes, the four selected ones are bigger in size. This makes

Table 1. Initial results of all classes of SSD on VisDrone2018.

IoU \ Class	pedestrian	people	bicycle	car	van	truck	Tricycle	awn.	bus	motor	mAP
0.50	5.60%	0.45%	0.00%	24.52%	2.64%	0.00%	0.00%	0.29%	0.04%	0.52%	4.10%
0.75	0.08%	0.00%	0.00%	7.47%	1.82%	0.00%	0.00%	0.22%	0.01%	0.01%	1.12%
0.50:0.05: 0.95	1.22%	0.08%	0.00%	10.63%	1.59%	0.00%	0.00%	0.20%	0.02%	0.11%	1.64%

Table 2. mAP of four classes with different cases using original hyperparameters.

Classes	Heavy occlusion	IoU ≥ 0.5	IoU ≥ 0.75	IoU ≥ 0.5:0.05:0.95					
		mAP	*mAP*	*mAP*	$AR^{max=1}$	$AR^{max=10}$	$AR^{max=100}$	$AR^{max=500}$	$AR^{max=1}$
11	✓	9.06%	3.10%	4.08%	0.80%	2.51%	3.95%	4.14%	0.80%
		9.81%	2.83%	4.12%	0.82%	2.35%	3.84%	4.01%	0.82%
4	✓	9.47%	3.76%	4.40%	2.24%	6.07%	8.51%	8.91%	2.24%
		10.41%	**4.09%**	**4.88%**	**2.72%**	**6.96%**	**9.63%**	**9.96%**	**2.72%**

Table 3. Evaluation results on four cases with new hyperparameters. Improved results are in bold.

Classes	Heavy occlusion	IoU ≥ 0.5	IoU ≥ 0.75	IoU ≥ 0.5:0.05:0.95					
		mAP	*mAP*	*mAP*	$AR^{max=1}$	$AR^{max=10}$	$AR^{max=100}$	$AR^{max=500}$	$AR^{max=1}$
4	✓	**10.83%**	**4.27%**	**4.99%**	**2.98%**	**7.19%**	**9.81%**	**10.14%**	**4.99%**
		10.95%	**4.37%**	**5.11%**	**2.84%**	**7.04%**	**9.85%**	**10.21%**	**5.11%**

the features they represent on a high-resolution image much clearer and in greater detail, easing out the process of detecting the class. Besides, according to our observations, in the dataset, most of the regarding objects have a lower proportion of being heavily occluded. Consequently, the chances of having confusing training samples are reduced substantially, which is why their scores are higher even when training on the full dataset. In fact, in our experiments, filtering out confusing annotations has further increased the accuracy of four classes by 0.26%.

Table 3 depicts the results using our proposed hyperparameters, where min ratio is reduced from 0.2 to 0.15. According to the numbers, lowering the min ratio outputs a better model. Specifically, the mAP all increases 0.41% on average. This is likely because now that we decrease the min ratio, lower level feature layers have a higher chance of correctly detecting objects. The original min ratio parameter is 0.2, which means the shallowest layers is expected to learn to detect objects with smaller scales starting at 20%. However, because most of training objects' scale factors are even lower, cutting down min ratio to about 0.15 has adequately covered more objects, resulting in an improved model.

All models mentioned in the evaluation have an average FPS of 18 when use Nvidia Tesla K80 GPU. The models perform better on Nvidia GeForce GTX 2080 Ti, where they achieve 30 FPS on average.

V. CONCLUSION

In this paper, we have conducted an evaluation of SSD, one of the greatest object detectors on new challenging large-scale dataset collected via drones, VisDrone2018. Experiment results show that training on four classes and filtered information gives an improved performance. In addition, our proposed hyperparameters have further increased the accuracy. Nonetheless, small-scaled objects are still a major setback for the model and the outcomes are still very low comparing to what it has achieved in other datasets. In the future, we will have a deeper look into the model and search for more ways to enhance the current performance.

ACKNOWLEDGMENT

This research is funded by Vietnam National University HoChiMinh City (VNU-HCM) under grant number C2018-26-03.

REFERENCES

[1] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu and A. C. Berg, "SSD: Single Shot MultiBox Detector," in Computer Vision – ECCV 2016, 2016, pp. 21-37.

[2] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[3] S. Ren, K. He, R. Girshick and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," CoRR, vol. abs/1506.01497, 2015.

[4] M. Everingham, L. V. Gool, C. K. I. Williams, J. Winn and A. Zisserman, The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results, http://www.pascal-network.org/challenges/VOC/voc2007/workshop/, 2007.

[5] M. Everingham, L. V. Gool, C. K. I. Williams, J. Winn and A. Zisserman, The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results, http://www.pascal-network.org/challenges/VOC/voc2012/workshop/, 2012.

[6] P. Zhu, L. Wen, X. Bian, H. Ling and Q. Hu, "Vision Meets Drones: A Challenge," CoRR, vol. abs/1804.07437, 2018.

[7] R. Girshick, J. Donahue, T. Darrell and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014.

[8] R. Girshick, "Fast R-CNN," in The IEEE International Conference on Computer Vision (ICCV), 2015.

[9] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition.," arXiv preprint arXiv:1409.1556, 2014.

[10] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in CVPR09, 2009.

[11] L. Weng, "Object Detection Part 4: Fast Detection Models," 27 December 2018. [Online]. Available: https://lilianweng.github.io/lil-log/2018/12/27/object-detection-part-4.html.