Using Deep Networks for Drone Detection

Cemal Aker, Sinan Kalkan
KOVAN Research Lab.
Computer Engineering, Middle East Technical University
Ankara, Turkey
{cemal, skalkan}@ceng.metu.edu.tr

Abstract

Drone detection is the problem of finding the smallest rectangle that encloses the drone(s) in a video sequence. In this study, we propose a solution using an end-to-end object detection model based on convolutional neural networks. To solve the scarce data problem for training the network, we propose an algorithm for creating an extensive artificial dataset by combining background-subtracted real images. With this approach, we can achieve precision and recall values both of which are high at the same time.

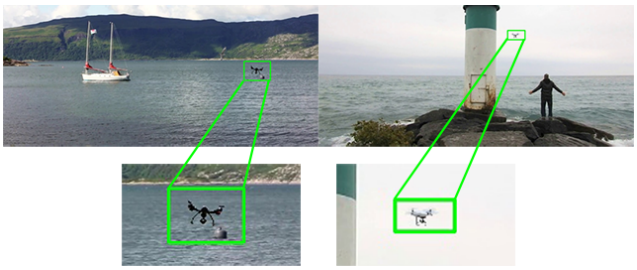

Figure 1: Detection samples from the created dataset where the green rectangles show the bounding boxes of the drones.

1. Introduction

Drone, as a general definition, is the name coined for the unmanned vehicles. However, in this paper the term will refer to a specific type, namely unmanned aerial vehicles (UAV). With the rapid development in the field of unmanned vehicles and technology used to construct them, the number of drones manufactured for military, commercial or recreational purposes increases sharply with each passing day. This situation poses crucial privacy and security threats when cameras or weapons are attached to the drones. Hence, detecting the position and attributes, like speed and direction, of drones before an undesirable event, has become very crucial.

Unpredictable computer controlled movements, speed and maneuver abilities of drones, their resemblance to birds in appearance when observed from a distance make it challenging to detect, identify and correctly localize them. In order to solve this problem, one can think of various types of sensors to perceive the presence of a drone in the environment. These may include global positioning systems, radio waves, infrared, and audible sound or ultrasound signals. However, it has been reported that they have many limitations for this problem, and suggested that computer vision techniques be used [6]. Although deep learning methods have been shown to be very powerful in computer vision tasks, the studies to detect UAVs have not taken advantage of it by placing deep learning methods at the core of the approach. To this end, this study is the first to evaluate the success of convolutional neural networks (CNN) as a standalone approach on drone detection.

In this study we have used an end-to-end object detection method based on CNNs to predict the location of the drone in the video frames. In order to be able to train the network, we created an artificial dataset by combining real drone and bird images with different background videos. The results show that the variance and the scale of the dataset make it possible to perform well on drone detection problem. With this method, we have participated in the Drone-vs-Bird Detection Challenge[1] organized within the International Workshop on Small-drone Surveillance, Detection and Counteraction Techniques, and our trained network ranked third in terms of lowest prediction penalty described in Section 4.

2. Related Work

In this section, we review the related studies in two parts.

[1] https://wosdetc.wordpress.com/challenge

978-1-5386-2939-0/17/$31.00 ©2017 IEEE

2.1. Object Detection Methods with Computer Vision

The task of object detection is to decide whether there are any predefined objects in a given image or not, and report the locations and dimensions of the smallest rectangles that bind them if they exist. Early attempts for this task involves the representations of objects using handcrafted features whereas the state of the art techniques utilizes deep learning.

Detection with Handcrafted Features: The most successful approaches using handcrafted features require bag of visual words (BoVW) [16] representations of the objects with the help of local feature descriptors such as scale invariant feature transform (SIFT) [9], speeded-up robust features (SURF) [1], and histogram of oriented gradients (HOG) [3]. After training a discriminative machine learning model, *e.g*., support vector machines (SVM) [2], with such representations, the images are scanned for occurrence of learned objects with the sliding window technique generally. These methods have two crucial drawbacks. The first one is that the features have to be crafted well for the problem domain to highlight and describe the important information in the image. The second one is the computational burden of the exhaustive search done by the sliding window technique.

Detection with Deep Networks: With the remarkable achievements of the deep learning methods in the image classification tasks, similar approaches have started to be used for attacking the object detection problem. These techniques can be divided into two simple categories; region proposal based and single shot methods. The approaches in the first category differs from the traditional methods by using features learned from data with CNNs and selective search or region proposal networks to decrease the number of possible regions [4, 5, 13]. In the single shot approach, the aim is to compute bounding boxes of the objects in the image directly instead of dealing with regions in the image. A method for this is extracting multi-scale features using CNNs and combining them to predict bounding boxes [7, 8]. Another one, named YOLO, divides the final feature map into a 2D grid and predicts a bounding box using each grid cell [11].

2.2. UAV Detection Methods with Computer Vision

Although the problem of detecting UAVs is not a well studied subject, there are some attempts to mention. Mejias *et al*. utilized morphological pre-processing and Hidden Markov Model filters to detect and track micro unmanned planes [10]. Gökçe *et al*. used cascaded boosted classifiers along with some local feature descriptors [6]. In addition to this pure spatial information based methods, spatio-temporal approaches exist. Rozantsev *et al*. propose a method that first creates spatio-temporal cubes using sliding window method at different scales, applies motion compensation to stabilize spatio-temporal cubes, and finally utilizes boosted tree and CNN based regressors for bounding box detection [14].

3. Method

Our solution is based on a single shot object detection model, YOLOv2 [12], which is the follow-up study of YOLO. We adapt and fine-tune this model to detect objects of two classes (*i.e*., drone and bird). Although the problem is detecting drones in the scene, we have included the bird class so that the network can learn robust features to distinguish them too. In order to achieve high accuracy with such deep models, one needs a large scale dataset that includes many scenarios of the problem, to get better generalization. To this end, we created an artificial dataset including real drones, real birds and real backgrounds. The following paragraphs first describe the approach in YOLOv2, the dataset creation approach, training and testing details.

3.1. The deep network

YOLOv2 tries to devise an end-to-end regression solution to the object detection problem. Former layers of the fully convolutional architecture that can be seen in Figure 2 are trained to extract high level features. Then the two highest level features are combined to get the final feature map of the image. Then it is divided into an $S \times S$ grid where the duty of each grid cell is predicting bounding boxes of the form (x, y, w, h, c). In this output, x and y are the coordinates of the centers of the boxes with respect to the grid cell, w and h are the width and height in proportion to the whole image, and c is the confidence that an object is in the bounding box. The final task of a grid cell is computing conditional class probabilities given the probability that the corresponding bounding boxes have objects in them. While predicting those bounding boxes, the model utilizes some prior information computed by K-means clustering on width and heights of ground truth bounding boxes. The final output size for a grid cell is:

$$Output\ Size = (N_{cls} + N_{coord} + 1) \times N_{anc},$$

where N_{cls} is the number of classes, N_{coord} is the number of coordinates, N_{anc} is the number of anchor bounding boxes used as prior knowledge and the 1 in the parenthesis is for the confidence value. In our approach, grid size is set to 15, number of classes is two, number of coordinates is four and number of anchor boxes is five. Hence, the final output is of the shape $15 \times 15 \times 35$.

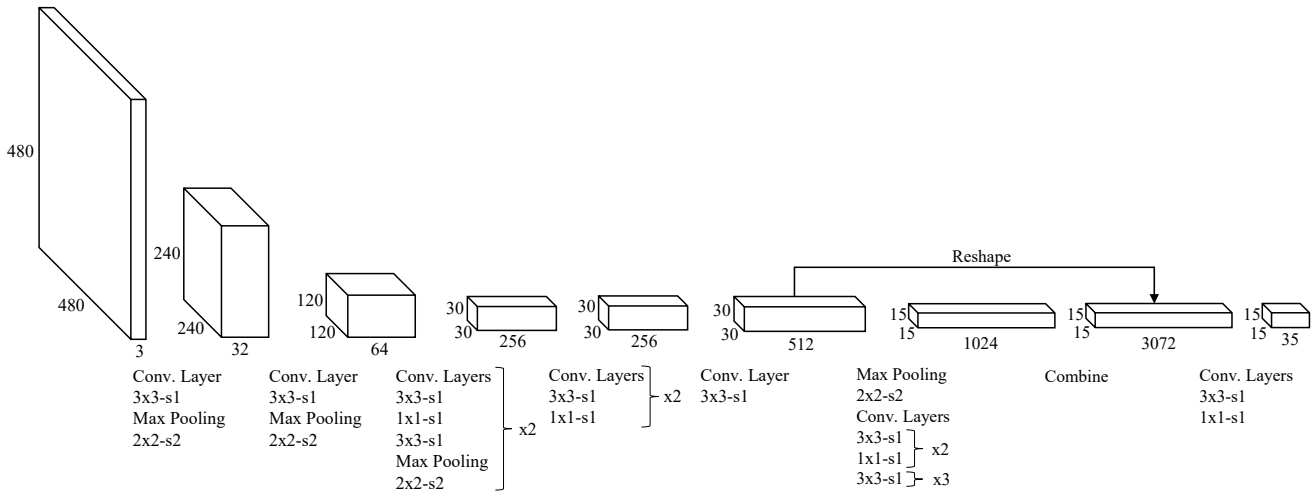

Figure 2: Our adaptation of the YOLOv2 network. All layers are fine-tuned with the dataset collected in the paper.

3.2. Dataset Preparation

Having mentioned the model details, we can now come to the most important part of the study which is dataset preparation. Since drone flights have limitations due to inadequate battery technology, weather conditions and legislative regulations, there is no publicly available large scale dataset for training deep networks. However, our approach requires immense amount of data to learn useful features. One possible solution to this is creating an artificial dataset. For this end, we have collected public domain pictures of drones and birds, and videos of coastal areas. After subtracting the background of drones and birds, they are randomly placed on the frames of the videos. The overall process is summarized in Algorithm 1. The details of the dataset can be found in the Table 1. As can easily be seen, the dataset needs a huge storage size when all of the configurations are used. Hence, we eliminated some portion of the configurations with probability

$$p = 1 - \frac{Max.\ allowed\ size}{Total\ size\ for\ all\ configurations},$$

to reduce the size of the dataset to reasonable amounts. Samples drawn from the resulting dataset can be seen in the Figure 3. These samples show that although they are created artificially, they look like real images of flying drones and birds.

The last things to mention about our approach are the training and prediction procedures. After creating the artificial dataset, we have applied a commonly used technique called fine-tuning. In this technique the network is first trained with a different and more general dataset for a similar problem. This provides us with better initial points than random for the parameters of the network. Then, training is continued with the actual dataset for the actual problem.

Figure 3: Samples from the artificial dataset which represent various scenarios with different backgrounds and bird inclusion. Although the dataset includes very small objects, the bigger ones have been chosen for better visibility. (best viewed in color).

This technique is useful especially when the training data is scarce. After training, there comes the prediction for unseen data. Since the network is trained with two classes, the bird detections are eliminated after getting all predicted bounding boxes from the last layer. Than a threshold, which can be determined according to accuracy on a validation set (our approach is explained in Section 4), is applied on the confidence values for objectness. If this operations eliminate all predicted bounding boxes, it means that the frame does not include a drone or it is not clear enough to detect. Otherwise, the one that has the highest confidence is selected as the prediction. We choose the best prediction to report since the aforementioned challenge requires to detect the only drone in the scene. However, the algorithm can easily be extended to multi-drone situations with more intelligent thresholding strategies. One possible problem with this ap-

Algorithm 1: The algorithm for preparing the dataset.

$S \leftarrow$ predefined size intervals
$D \leftarrow$ foregrounds of drone images
$B \leftarrow$ foregrounds of bird images
$V \leftarrow$ background videos
$R \leftarrow$ # of rows that the image will be divided into
$C \leftarrow$ # of columns that the image will be divided into
$G \leftarrow R \times C$ grid
foreach $(d, g, s, v) \in D \times G \times S \times V$ **do**
    ignore this configuration with probability $p = 1 - \frac{Max.\ allowed\ size}{Total\ size\ for\ all\ configurations}$, and continue
    draw a random position p_0 in g
    draw a random size s_0 for smaller edge of the drone from s
    draw a random frame f_0 from v
    resize d with respect to s_0
    overlay f_0 with d in position p_0
    draw (p_1, s_1, f_1) in the same way
    draw a random bird b_0 from B
    draw $(p_{b,0}, s_{b,0})$ for bird where $s_{b,0}$ is drawn from smaller half of S
    resize d with respect to s_1
    overlay f_1 with d in position p_1
    resize b_0 with respect to $s_{b,0}$
    overlay f_1 with b_0 in position $p_{b,0}$
    draw (p_2, s_2, f_2) in the same way
    draw a random bird b_1 from B
    draw $(p_{b,1}, s_{b,1})$ for bird where $s_{b,1}$ is drawn from greater half of S
    resize d with respect to s_2
    overlay f_2 with d in position p_2
    resize b_1 with respect to $s_{b,1}$
    overlay f_1 with b_1 in position $p_{b,1}$

    save f_0, f_1, f_2 into the dataset
end

proach is encountered when the network mixes a bird up with the drone. If the objectness confidence of it is higher than that of the drone, it is selected as the prediction. In order to decrease the number of such misinterpretations, we propose a *limited ignorance approach*. After determining the bounding box that the network is most confident, we control its intersection with the rectangle having same center, three times the width and height as the predicted bounding box in the previous frame, assuming that the drone cannot move more than its height or width in a single frame. If the rectangles intersect, we can accept the newly predicted one. Otherwise, we ignore the current prediction and report the previous one if the limit has not been exceeded yet.

Table 1: Details of the dataset.

Aspect	Information
# drones	89
# birds	126
# background videos	11
# rows in grid	12
# columns in grid	10
# size intervals	19
size intervals	in [5,160] (bias towards smaller values)
image resolution	850×480
# resulting images	676,534

After exceeding the limit, we reset it and cancel the technique for the same number of frames. During this period, we report the current predictions directly. Likewise, when there is no predicted bounding box in the previous frame, we directly report the prediction in current frame.

4. Experiments

This section describes training details and the conducted experiments on the artificial dataset and the real dataset provided by the organizers of the challenge. The former ones are evaluated quantitatively whereas the others are evaluated qualitatively due to the lack of ground truth information.

Training details: In order to apply fine tuning mentioned in Section 3, we have started with the pre-trained weights using the ImageNet dataset [15] for image classification problem. Then the dataset provided by the challenge organizers and the created one are divided into training (85%) and validation (15%) parts. The training part of the former one is duplicated four times before combining them to training sets since it is too scarce compared to the artificially created, large scale one. Then, the network is fine-tuned for 10,000 iterations with 128 as batch size and batch normalization after all convolutional layers.

After the training phase is completed, we combined the two validation sets to evaluate the resulting network. Although we use $480 \times 480 \times 3$ as input size in training (see Figure 2), we increase the resolution to $800 \times 800 \times 3$ in testing configuration. This is applicable since the network is fully convolutional. This increase is helpful in detecting small sized targets.

Evaluation metrics: We use precision-recall curves to evaluate the network. The curves are constructed by changing the detection threshold. The precision metric is defined as $\frac{tp}{tp+fp}$, where tp is the number of true positives and fp is the number of false positives. Recall is then defined as $\frac{tp}{tp+fn}$, where fn is the number of false negatives. We count

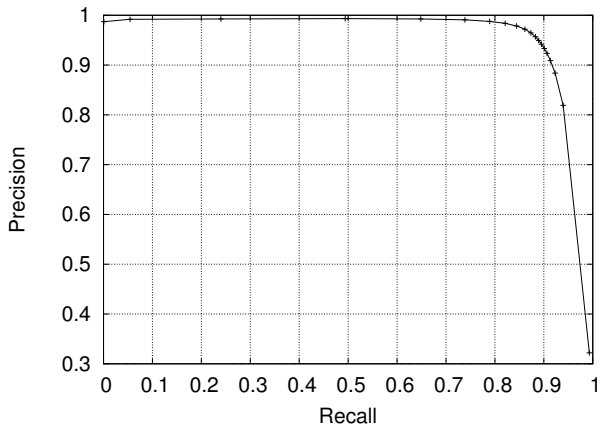

Figure 4: Precision-Recall (PR) curve showing the performance of the approach on the outdoor test videos.

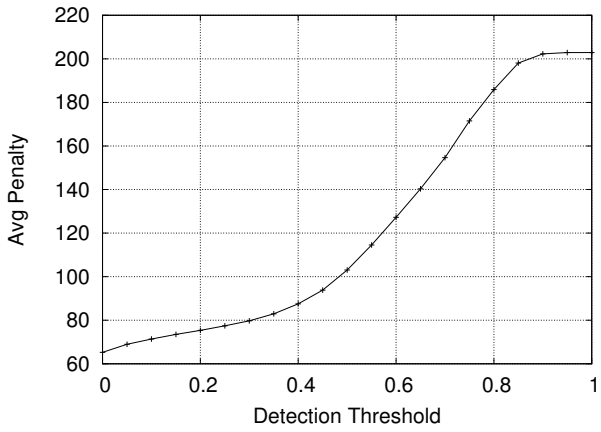

Figure 5: Change of prediction penalty with respect to detection threshold.

a predicted bounding box as true positive if the area of the overlap of the predicted bounding box with the ground truth is greater than half of the area of their union.

Another metric that we used is the prediction penalty, which is basically the area of smallest rectangle that includes both the ground truth and predicted bounding boxes divided by the area of ground truth bounding box.

Results: Figure 4 presents the performance of the method with different detection thresholds in the range [0,1]. The closer the Precision-Recall (PR) curve to the top right corner the better the performance of the method. We can understand from the curve that precision and recall can be achieved to be approximately 0.9 at the same time. This shows that the approach performs well in detecting the correct bounding boxes.

Figure 5 shows the change of the average penalty with respect to detection threshold. The reason for higher penalties is that when the threshold increases detection rate decreases. When a drone cannot be found, the top-left pixel is reported as prediction, which results in a huge penalty. Hence, we have chosen the smallest possible threshold (which is zero) for quantitative evaluation on the test video of the challenge. Although this threshold hurts precision in the artificial dataset, it works well in the provided test video except its detecting the bird as drone when the bird is closer to the camera and in specific poses that cannot be easily distinguished from a drone by human eye. Another observation is that when the drone and the bird are too close to each other, the network supposes that the bird is a part of the drone, and outputs a bounding box enclosing both of them. The predictions are provided online[2] as a video rendered in 15 fps.

[2] `http://user.ceng.metu.edu.tr/~cemal/predictions.mp4`

5. Conclusion

In this study, we showed that drones can be detected and distinguished from birds using an object detection model based on a CNN. The trained network generalizes well as it can achieve high precision and recall values at the same time.

For future work we plan to consider time domain to improve the performance even further. Since collecting such data is not easy, we plan to devise an algorithm that generates random flight videos instead of randomly generated images.

References

[1] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *CVIU*, 110(3):346–359, 2008.

[2] C. Cortes and V. Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[3] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[4] R. Girshick. Fast r-cnn. In *CVPR*, 2015.

[5] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[6] F. Gökçe, G. Üçoluk, E. Şahin, and S. Kalkan. Vision-based detection and distance estimation of micro unmanned aerial vehicles. *Sensors*, 15(9):23805–23846, 2015.

[7] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*, 2014.

[8] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016.

[9] D. G. Lowe. Object recognition from local scale-invariant features. In *ICCV*, 1999.

[10] L. Mejias, S. McNamara, J. Lai, and J. Ford. Vision-based detection and tracking of aerial targets for uav collision avoidance. In *IROS*, 2010.

[11] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016.

[12] J. Redmon and A. Farhadi. YOLO9000: better, faster, stronger. *CoRR*, abs/1612.08242, 2016.

[13] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, 2015.

[14] A. Rozantsev, V. Lepetit, and P. Fua. Detecting Flying Objects using a Single Moving Camera. *PAMI*, 39:879 – 892, 2017.

[15] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015.

[16] J. Sivic, A. Zisserman, et al. Video google: A text retrieval approach to object matching in videos. In *ICCV*, 2003.