Real Time UAV Detection using Deep Learning

Omkar Vinayak Nadkarni
190758309
Prof. Ebroul Izquierdo
MSc FT Artificial Intelligence with
Industrial Experience
School of Electronic Engineering and
Computer Science
Queen Mary (University of London)
London, United Kingdom
o.nadkarni@se19.qmul.ac.uk

***Abstract*— Drones have become very popular in recent times with many applications in different fields such as delivery, entertainment and surveillance. This has led to security concerns specifically in sensitive areas such as airports, military bases. In this work I have proposed an object detection model based on convolutional neural networks to detect drones in the sky in real time. The model is trained on different videos of drones available online with annotations consisting of location of drone in all frames. The model is trained considering the camera is stationery at all times and since the data available for training is very limited it is difficult to train an entire network that can generalize well across different scenarios. To address this issue two techniques are used. Firstly, a pretrained model is trained further also known as transfer learning which allows the model to converge faster and secondly, data augmentation techniques are used to make the data more diverse and robust to geometrical and illumination changes. The model is evaluated on validation set during training by calculating a weighted average of precision, recall, F1 score and mean average precision (mAP) and model with best performance is selected.**

I. INTRODUCTION

Unmanned aerial vehicles or more commonly known as drones have become very popular mainly in sectors such as delivery, entertainment and navigation. It is also easy to attach cameras for surveillance or weapons to cause destruction which poses a huge threat to property and human life. Due to lack of government regulation it is also very accessible and in the wrong hands can prove very destructive. Although there has been extensive research in object detection using computer vision based methods and there exist state-of-the-art models such as EfficientDet (Tan, et al., 2019) , DetectoRS (Qiao, et al., 2020), drone detection system cannot really take advantage of this due to its time complexity during inference as the system needs to detect objects in real time and even slight delay could have devastating results. Apart from this, drones are also available in different shapes and sizes and in some cases can also resemble other objects such as planes, birds which makes it difficult for the model to correctly classify any identified object as drone or not. Other factors include geometrical and illumination changes due to weather and unpredictable movements which makes object tracking difficult.

In this study we have implemented a single stage object detection model capable of processing video frames in real-time and detect the presence of drone and predict its accurate location. To train a network that can generalize well and detect accurately in different scenarios such as illumination changes and in variant to distance of the object it is important to have large amount of training data. Since there is no publicly available data, we have employed techniques such as transfer learning and data augmentation to train the network efficiently and achieve good results.

The rest of the paper is organized as follows. In section 2 we will discuss about the different techniques and approaches that have been explored in this domain. Section 3 contains the methodologies and implementation of the framework proposed by us. In section 4 we will discuss about the performance, analysis and experimental results. The paper concludes with future work in section 5.

II. LITERATURE REVIEW

In order to achieve a robust drone detection system many different techniques and methodologies have been explored. In this section I will mainly focus on computer vision based solutions. Although there are different approaches to object detection the most popular is two stage object detection model. In this type of model, the first stage includes a region proposal network which is trained to identify regions of interest i.e regions containing objects in the input image. The second stage then determines the category the object belongs to along with bounding box regression. Such types of models are popular mainly due to its high accuracy but are computationally expensive and therefore slower. In this section we will first talk about research done by adopting this model followed by other novel approaches. In the paper by Saqib, et al., (2017) the authors compared two different classification architectures namely, VGG-16 (Simonyan & Zisserman, 2014) and ZFNet (Zeiler & Fergus, 2014) using Faster R-CNN (Ren, et al., 2015) model architecture. The models were evaluated based on mean average precision (mAP). Due to inadequate training data pre-trained models were used.

A similar model was proposed by Schumann, et al.(2017) where in the authors employed different techniques depending on camera configuration. In case of video feed received from stationery camera median background subtraction was used to determine objects that are in the foreground where as in case of moving camera, a region proposal network was employed. The second stage included a classification module to detect the object. The model was trained on two classes namely, drones and birds to improve performance and avoid misclassification.

Apart from this, researchers have also experimented with one stage detectors such as YOLOv2 (Redmon & Farhadi, 2017) by (Aker & Kalkan, 2017). To solve the scarce data problem an artificial dataset was created which included drones and birds. The paper also presented a prediction penalty to penalize the model for wrong predictions and the resulting model could process videos at 15 frames per second (fps).

Another interesting approach worth mentioning is by (Rozantsev, et al., 2016) where the features were extracted by combining multiple frames and using sliding window approach at different scales along with motion compensation to obtain spatio-temporal cubes. A classifier was then trained on these cubes to determine the presence of a flying object.

III. PROPOSED FRAMEWORK

In this study we have used YOLOv4 (Bochkovskiy, et al., 2020) as our base model and fine-tuned it to fit our dataset. Since there is lack of training data to train the network of this size (100M parameters), a pre-trained model trained on ImageNet (Russakovsky, et al., 2015) is used. This helps the model to converge faster.

A. The YOLO network

YOLO is a one stage detector which treats object detection as a regression problem. The architecture has three important parts which are explained in detail below.

Backbone: The backbone is responsible for extracting important features. This can be done by any classification network such as ResNet (He, et al., 2016), EfficientNet by removing the last layer which is responsible for classification. In the previous version YOLOv3 (Redmon & Farhadi, 2018), DarkNet53 was introduced which has 53 convolutional layers, has over 28M parameters and follows the idea of DenseNet (Huang, et al., 2017) using residual blocks and skip connections. Each convolutional layer is followed by batch-normalization layer and uses Leaky ReLu as the activation function. In yolov4 this network is further improved by implementing cross stage partial network (CSPNet) (Wang, et al., 2020) over it which significantly improved the processing speed of the network.

Neck: The neck consists of a spatial pyramid pooling (SPP) (He, et al., 2015) block which increases the receptive field of the network and allows propagation of important context features without adding to computational time. Along with this, a modified version of path aggregation network (PANet) (Liu, et al., 2018) is used which concatenates the features obtained. This benefits small scale object detection even when detection is for large object.

Head: In two stage detectors the bounding box predictions and the classification of the object is done separately where as in one stage detector such as YOLO it is done at the same time. The head outputs a vector of size (x, y, w, h, c) format where x,y are the centres of the bounding box, w,h corresponds to width and height and c is the score of confidence that an object is present inside the box also known as objectness score. To determine the class of the object, class probability is computed by conditioning it over the objectness score in the given box. Bounding box predictions can differ greatly depending on objects such as detecting a person requires a bounding box of ratio 1:3 where as a car will require 2:1. To find bounding box priors also known as anchors YOLO uses k-means clustering to determine the ratios in which the bounding boxes are likely to appear from the labels in the training dataset and during training the model learns to make slight adjustments to these bounding boxes which are easier to learn than the large ones.

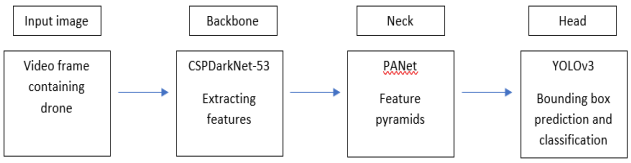

Figure 1: Overview of YOLOv4 framework

Non-Maximal suppression

The model can predict multiple bounding boxes to the same object which will reduce model's accuracy. To solve this issue, non-maximal suppression is used which selects the bounding box with highest confidence and all the the boxes which intersect with it and have higher IoU than a set threshold are removed.

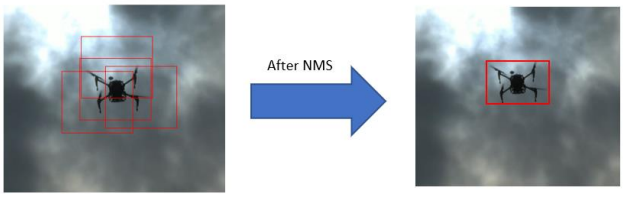

Figure 2: object localization after non-max suppression

IV. EXPERIMENTAL RESULTS

A. Dataset

The dataset is prepared by obtaining various annotated drone videos which include different illumination settings and various drone models. Each frame is converted to an image. The total set consists of 12,500 images which is further divided into train (80%) and validation (20%) set. The resolution of the videos is 1920*1080.

B. Training the network

The training was done using pre-trained weights trained on ImageNet Dataset. The network is then trained for 200 epochs with batch size 64 and subdivisions of 16 and learning rate of 10^{-3} and uses stochastic gradient descent as the optimizer. The training images are further resized to 1024*1024 in order to reduce computational complexity during training. In this work we have used a Pytorch implementation of the YOLO network. The model is trained using two nvidia RTX 6000 for a total time of 43 hours.

Data augmentation: Data augmentation is often used when training data is limited. For this problem we have used mosaic data augmentation technique which combines multiple training images into one with different aspect ratios. This allows model to learn to identify objects at different scales and sizes and also allows it to localize object at different positions in the input frame.

Figure 3: input image after mosaic data augmentation

C. *Evaluation metrics*

There are four important metrics which are discussed below along with graphical representation and are computed for object confidence threshold of 0.5.

Precision: Precision tells us how many predictions made by the model are true out of the total predictions during evaluation. This is calculated as $\frac{tp}{tp+fp}$ where tp stands for true positives and fp stands for false positives. A predicted object instance is considered as true positive only when its IoU is greater than 0.5 and all others are considered as false positives.

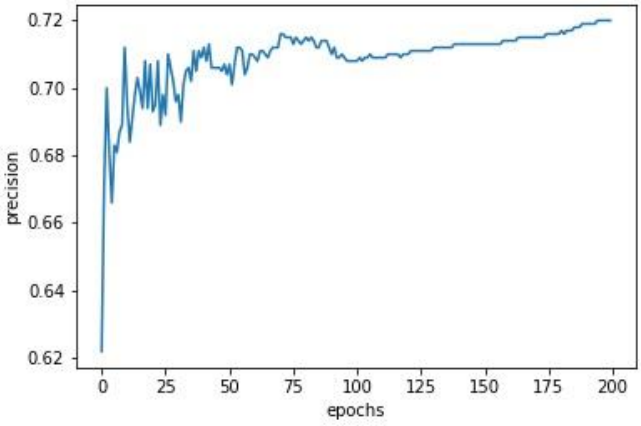

Figure 4: The figure shows precision on validation set during training.

Recall: It is defined as the measure of correctly identified object instances out of total ground-truth instances. It is calculated as $\frac{tp}{tp+fn}$ where fn stands for false negatives which are the instances not detected by classifier.

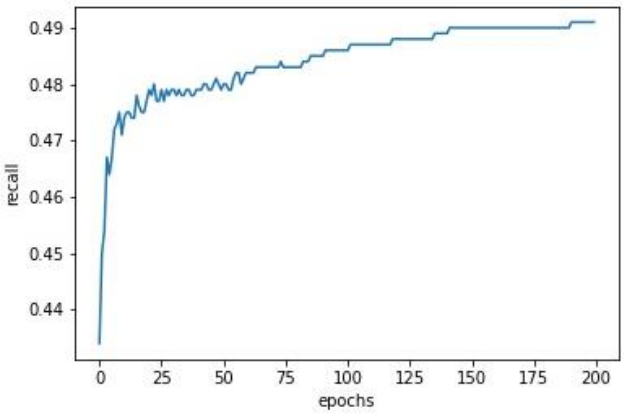

Figure 5: The figure shows recall on validation set during training.

F1 score: It is defined as the harmonic mean of precision and recall and is calculated as $2.(\frac{Precision\ X\ Recall}{Precision+Recall})$ and gives a better understanding of the model's performance.

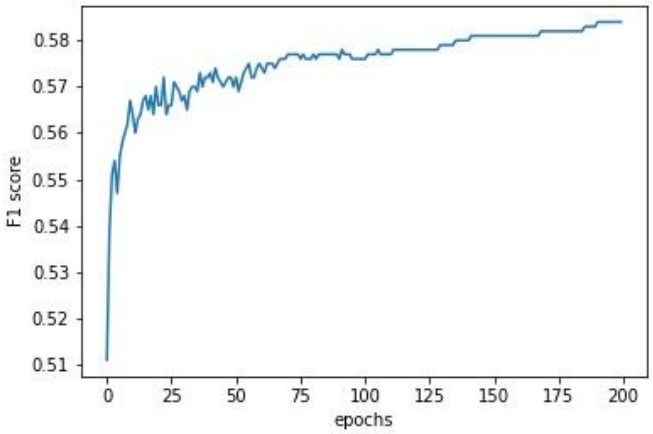

Figure 6: The figure shows F1 score on validation set during training.

Generalized intersection over union (GIoU): GIoU was implemented by Rezatofighi, et al. (2019) which is an improvement over the very popular IoU or commonly known as jaccard index. IoU is calculated as $J(A,B) = \frac{|A \cap B|}{|A \cup B|}$. The disadvantage is that when there is no intersection between prediction and ground-truth the value is zero. As shown in figure 7 below, The model can become better at predictions even though there is no intersection. GIoU solves this issue by calculating the smallest area that encloses both prediction and the label. GIoU is calculated as $GIoU = IoU - \frac{|C \setminus (A \cap B)|}{|C|}$ where C is the smallest area that encloses both the bounding boxes. The range of the GIoU is -1 to 1 and is negative when C is greater than IoU and converges to 1 as predictions get closer to ground truth. Another disadvantage of IoU is that it is zero when there is no intersection between predicted bounding box and ground truth and gradient does not exist in such cases where as GIoU is always differentiable.

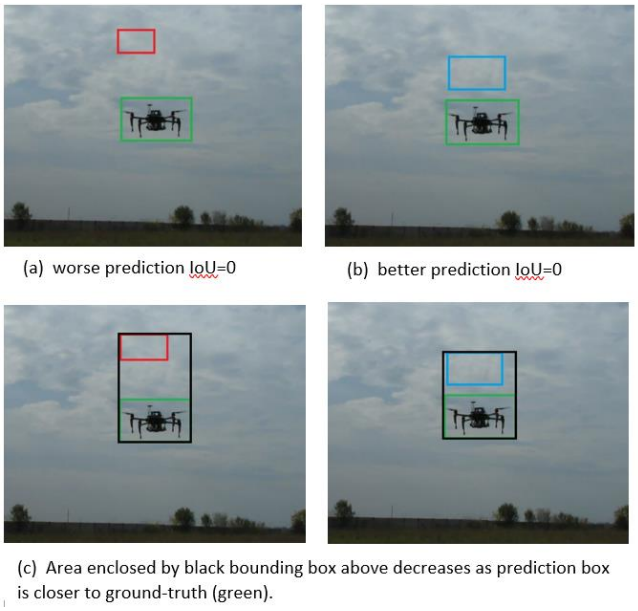

Figure 7: In (a) and (b) IoU loss does not change as intersection is zero where as GIoU has a negative value.

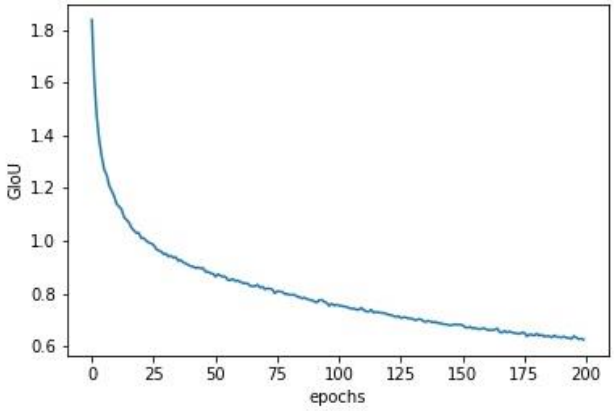

Figure 8: The figure shows decrease in GIoU loss during the training of the model.

D. Performance on test set

Although the model is successful in detecting drones confidently even at larger distances as shown in figure 8, there are several instances where it misclassifies other objects such as birds as drones. This misclassification is highly variable and depends on lot of factors such as illumination, background and distance of the bird from the camera as well as the orientation of the bird. The model is able to process videos of 1920*1080 resolution at 20 frames per second running on single GTX 1080Ti GPU.

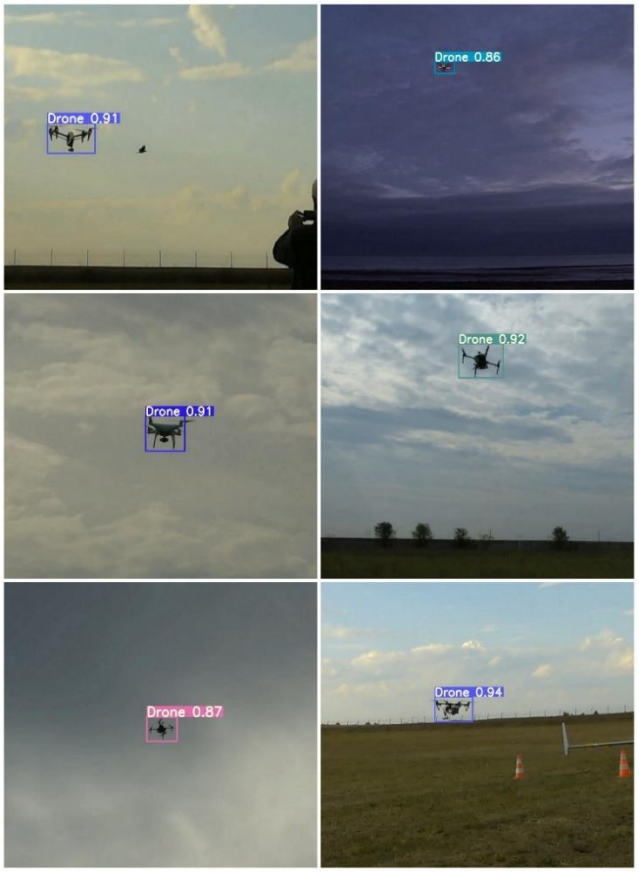

Figure 9: Drone detection by model during inference.

Apart from this, the model is tested at different detection thresholds to check its performance. Given below is the summary of the experiment.

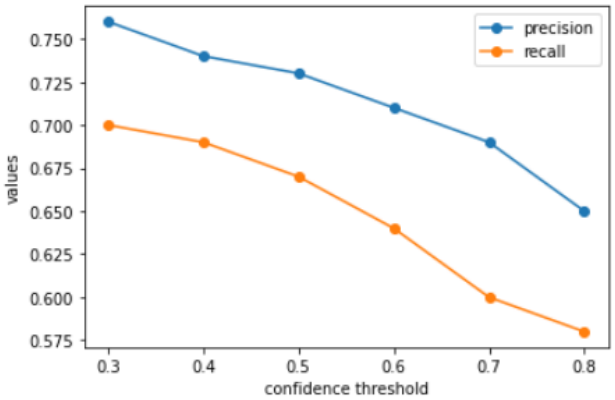

Figure 10: precision and recall values on test dataset at different thresholds.

From figure 10 we can infer that model is accurately able to detect drones but has lower confidence on its detections. The model has lower confidence when detecting drones that are at a larger distance and appear smaller in the input frame.

Given below are some example where model misclassifies other objects as drones and in some cases does not detect drone.

Figure 11: Street light being detected as drone during inference.

In figure 11 above, the street light is detected as a drone. Scenarios like this can be rectified by using background subtraction technique since in a static camera configuration the position of the light will remain same and it will be considered as a background and hence not detected as a drone.

Figure 12: bird being detected as drone during inference

In figure 12 the bird is detected as a drone and there are two techniques to rectify this. Increasing the detection threshold as the model is only 36% certain that the object is a drone will work in most cases but not all as shown in figure 12. Another technique which is more robust to failures is training the model to distinguish between drone and other distractor classes such as bird and planes.

Figure 13: Drone not being detected

In figure 13 the bird is detected as a drone where as the drone is not detected at all. This also depends on orientation of the objects in frame and is very rare that the detection threshold is so high.

V. CONCLUSION

In this work, we have presented a drone detection framework capable of accurately identifying and locating drones at different scales in real time.

In future work, we plan to focus on using the information from object prediction in the preceding frame to localize the search and make the algorithm more efficient to run on systems with limited resources and train the model on more classes such as birds, planes to make it robust to misclassification with these objects.

VI. REFERENCES

Aker, C. & Kalkan, S., 2017. *Using Deep Networks for Drone Detection.* s.l.:14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS) (pp. 1-6). IEEE..

Huang, G., Liu, Z. & Maaten, L. v. d., 2017. *Densely Connected Convolutional Networks.* s.l.: In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4700-4708)..

Redmon, J. & Farhadi, A., 2017. *YOLO9000:Better, Faster, Stronger.* s.l.:In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 7263-7271).

Ren, S., He, K., Girshick, R. & Sun, J., 2015. *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.* s.l.:In Advances in neural information processing systems (pp. 91-99).

Rozantsev, A., Lepetit, V. & Fua, P., 2016. *Detecting Flying Objects Using a Single Moving Camera.* s.l.:IEEE transactions on pattern analysis and machine intelligence, 39(5), pp.879-892..

Schumann, A. et al., 2017. *Deep cross-domain flying object classification for robust UAV detection.* s.l.:In 2017 14th

IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS) (pp. 1-6). IEEE.
Simonyan, K. & Zisserman, A., 2014. *VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION.* s.l.:arXiv preprint arXiv:1409.1556.
Zeiler, M. D. & Fergus, R., 2014. *Visualizing and Understanding Convolutional Networks.* s.l.:In European conference on computer vision (pp. 818-833). Springer, Cham.
Bochkovskiy, A., Wang, C.-Y. & Liao, H.-Y. M., 2020. *YOLOv4: Optimal Speed and Accuracy of Object Detection.* s.l.:arXiv preprint arXiv:2004.10934.
Bochkovskiy, A., Wang, C.-Y. & Liao, H.-Y. M., 2020. *YOLOv4: Optimal Speed and Accuracy of Object Detection.* s.l.:arXiv preprint arXiv:2004.10934..
He, K., Zhang, X., Ren, S. & Sun, J., 2016. *Deep Residual Learning for Image Recognition.* s.l.:In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778)..
He, K., Zhang, . X., Ren, S. & Sun, J., 2015. *Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition.* s.l.:IEEE transactions on pattern analysis and machine intelligence, 37(9), pp.1904-1916..
Liu, S. et al., 2018. *Path Aggregation Network for Instance Segmentation.* s.l.:In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 8759-8768)..
Qiao, S., Chen, L.-C. & Yuille, A., 2020. *DetectoRS: Detecting Objects with Recursive Feature Pyramid.* s.l.:arXiv preprint arXiv:2006.02334.
Redmon, J. & Farhadi, A., 2018. *YOLOv3: An Incremental Improvement.* s.l.:arXiv preprint arXiv:1804.02767..
Rezatofighi, H. et al., 2019. *Generalized Intersection over Union: A Metric and A Loss for Bounding Box Regression.* s.l.:In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 658-666).
Russakovsky, O. et al., 2015. *ImageNet Large Scale Visual Recognition Challenge.* s.l.:International journal of computer vision, 115(3), pp.211-252.
Saqib, M., Khan, S. D., Sharma, N. & Blumenstein, M., 2017. *A study on detecting drones using deep convolutional neural networks.* s.l.:In 2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS) (pp. 1-5). IEEE.
Tan, M., Pang, R. & Le, Q. V., 2019. *Efficientdet: Scalable and efficient object detection.* s.l.:arXiv preprint arXiv:1911.09070..
Wang, C.-Y.et al., 2020. *CSPNet: A New Backbone that can Enhance Learning Capability of CNN.* s.l.:In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (pp. 390-391)..