Drone Detection Using Deep Learning: A Benchmark Study

Ahmed Hashem[1] and Thomas Schlechter[2](✉)

[1] Linz Center of Mechatronics GmbH, 4040 Linz, Austria
ahmed.hashem@jku.at
[2] University of Applied Sciences Upper Austria, 4600 Wels, Austria
thomas.schlechter@ieee.org
https://www.lcm.at/, https://www.fh-ooe.at/

Abstract. Since Unmanned Aerial Vehicles (UAVs) became available to the civilian public, it has witnessed dramatic spread and exponential popularity. This escalation gave rise to privacy and security concerns, both on the recreational and institutional levels. Although it is mainly used for leisure and productivity activities, it is evident that UAVs can also be used for malicious purposes. Today, as legislation and law enforcement federations can hardly control every incident, many institutions resort to surveillance systems to prevent hostile drone intrusion.

Although drone detection can be carried out using different technologies, such as radar or ultra-sonic, visual detection is arguably the most efficient method. Other than being cheap and readily available, cameras are typically a part of any surveillance system. Moreover, the rise of deep learning and neural network models rendered visual recognition very reliable [9,21].

In this work, three state-of-the-art object detectors, namely YOLOv4, SSD-MobileNetv1 and SSD-VGG16, are tested and compared to find the best performing detector on our drone data-set of 23,863 collected and annotated images. The main work covers detailed reportage of the results of each model, as well as a comprehensive comparison between them. In terms of accuracy and real-time capability, the best performance was achieved by the SSD-VGG16 model, which scored average precision (AP50) of 90.4%, average recall (AR) of 72.7% and inference speed of 58 frames per second on the NVIDIA Jetson Xavier kit.

Keywords: Drone detection · Neural network · Security · Artificial intelligence

1 Introduction

In 2013, after Amazon announced its plan to use drones for package delivery, hobby drones had become exponentially popular [5]. As per usual, legislative regulations were incapable of addressing the fast pace of advancements and inventions of technology. The widespread use of drones raised major privacy, security and safety concerns. These predicaments and safety concerns called for the need for drone, or rather anti-drone, surveillance systems that can detect drone intrusions and, if necessary, neutralize them.

In [21], Taha and Shoufan list and compare several drone detection modalities, namely RADAR, RF, IR and ultra-sonic, and compare their performance. Through that review, as well as many other papers, it can be concluded that visual detection, especially using recent advancements in Deep Neural Networks (DNNs) and Convolutional Neural Networks (CNNs), has become the most reliable and efficient method to achieve accurate and real-time object detection.

What makes drone detection a challenging task, however, is the characteristics of drone appearance and movement. There exists a myriad of drone models that can look very different from each other. From the number of rotors to the skeleton structure and color, drones constitute a real generalization challenge to detection algorithms. They can also come in a huge range of sizes, from a few centimeters to a couple of meters span. Finally, they have common shape characteristics with leafy branch endings of trees, birds and distant airplanes. In indoor environment, they can be hidden in complex backgrounds and can have similar appearance to other devices.

Drones have a very particular, yet very challenging, flight style as well. They can perform high acceleration and rotation movements in very small range. They can also fly at relatively high speeds for camera shutter to get clear resolution shots of them. This means they can appear in camera frames very small or very large, depending on the drone size and distance from the camera. They can also appear very blurred, depending on flight speed. Finally, they can stabilize in the air mimicking stationary objects.

In this work we utilize the recent advancements in deep learning and CNN to train state-of-the-art object detection models to detect drones. The paper sections are organized as follows: Sect. 2 presents a comprehensive review of the literature and related work with emphasis on the models tested in this work. In the third section, the methodology, a thorough description of the hardware setup, the software framework as well as the collected training data-set are presented. The results of training and testing of the models are presented and discussed in Sect. 4. Finally, Sect. 5 concludes the paper and presents its summary.

2 Related Work

Just like several other computer vision tasks, object detection can be carried out using DNNs or conventional computer vision algorithms. However, for drone detection, and due to the difficulties mentioned before, DNNs significantly outperform even the most sophisticated computer vision algorithms, such as Haar Cascades [17].

Take the YOLO (You Only Look Once) detector for instance; Version 4 of this detector has a "43.5% Average Precision (AP) (65.7% AP at 50% accuracy or AP50) on the MS COCO data-set at a real-time speed of 65 Frames Per Second (fps) on Tesla V100" [2]. YOLO was the first model of what is called

Single Shot Detectors. The name refers to the fact that these detectors can locate the object of interest and draw the bounding box around it in a single shot, as opposed to older models that had to do that on two stages with two different networks. This category of detectors gave hope in having real-time performance, which was previously unthinkable.

One year after the first version of YOLO was released, the Single Shot Multi-Box Detector (SSD) was published [11]. Although it was originally used with the "VGG-16" network as a "base network", the modularity of the framework allows for using other CNNs as base networks as well. In this paper, the YOLOv4 as well as the SSD-VGG-16 and SSD-MobileNet-v1 are tested and benchmarked on the collected data-set [6].

3 Methodology

The three state-of-the-art detectors, namely: YOLOv4, SSD-VGG-16 and SSD-MobileNet-v1, are all pre-trained on major detection datasets. YOLOv4 was pre-trained on the MS COCO data-set [10], while the other two networks were both trained on the PASCAL VOC data-set [3]. Here they are re-trained on the original drone data-set, that was manually collected and annotated during this work. The following sub-sections elaborate on the hardware and software framework that was used for training and testing, as well as the details of the dataset.

3.1 Hardware Setup

Three different drones, with different appearance, color and size, were chosen to be used in building the data-set. The Crazyflei drone [1] is the smallest of the three, just 6×6 cm. The Parrot Mambo Fly drone [16] is 18×18 cm, including the defenders. The biggest of the three is the black Syma X5SC drone [20], which spans 31×31 cm.

For our training, three different NVIDIA GPUs were used simultaneously to decrease the training time. They were used on separate machines and for training separate models though. The first is an NVIDIA GeForce RTX 2080-Ti with 11 GB GPU memory. The card is widely used for training DNNs and was used by the developers of YOLOv4 for their experiments [2]. Its relatively large memory and 4352 cores makes it a great choice for computer vision applications [13]. A training speed of 5 images/second was achieved using this card when training the YOLOv4 model with an input size of 512×512 and a minibatch size of 4.

The second GPU is the NVIDIA GeForce GTX 1060 with 6 GB of memory. Although this card is much less powerful than its 2080-Ti counterpart, it could still help with its 1280 cores [12]. The experiments that were performed on that card showed training speed half of that of the 2080-Ti but triple of that of the NVIDIA Jetson, which is our third and last GPU.

The NVIDIA Jetson AGX Xavier developer kit, was mainly used for inference. Being specifically designed to run AI and deep learning applications makes it a perfect choice for testing and implementing the chosen detectors [14]. As one of the most recent releases from NVIDIA, little research has been conducted using this powerful tool. This is why we present all the testing results using this card to evaluate its performance and limitations.

The 32-GB Jetson Xavier has a 512-core Volta GPU with a new technology called Tensor Cores [14]. Tensor Cores are GPU cores designed specifically for tensor multiplication. Tensor multiplication, which is the basic operation of almost all deep learning computations, is done instantly in chunks accumulating the results in high throughput registers. This results in significant speedup in performance [15]. The performance of the Jetson on drone detection is reported in the results section.

3.2 Software Framework

TensorFlow and PyTorch are arguably the biggest and strongest machine learning frameworks in Python. Both open libraries were developed by the tech giants Google and Facebook respectively. PyTorch, which was released in 2016 about a year after TensorFlow, was developed for machine learning and AI applications with emphasis on GPU utilization and optimization. Similar to TensorFlow, its paradigm enables lean and fast differential programming. However, PyTorch is more compliant with the Object Oriented Programming (OOP) paradigm and the familiar Python style [18]. The Tensor class, which enables handling multi-dimensional arrays and data-sets, is similar to the NumPy arrays data type with the additional capability to operate on GPUs [19].

Although TensorFlow is a very powerful library that enables great control over models and dataflow, the familiarity of PyTorch style renders it more attractive, especially that it also provides similar control and services. Overall, TensorFlow is still more widely used in industry and research than PyTorch. However, the trends show that PyTorch is becoming more and more popular and is forecasted to overtake TensorFlow in the near future, Fig. 1. For these reasons, and after the consultation of industry and research experts, PyTorch was the chosen platform for this project.

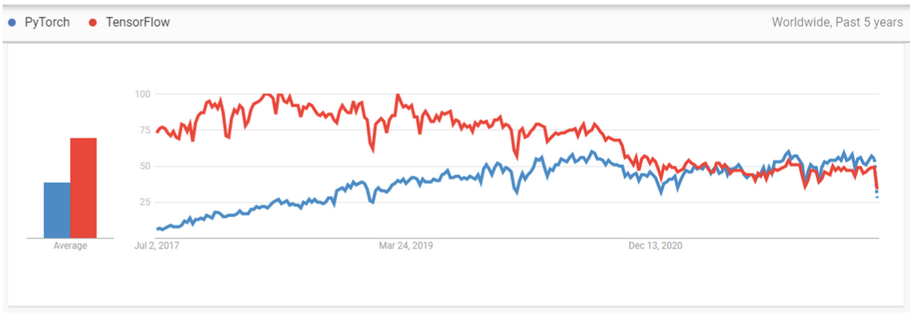

Fig. 1. Google search statistics for TensorFlow and PyTorch [4].

3.3 Data-Set

The data-set was recorded at first in videos using an Intel RealSense camera [7], and then filtered and converted to images. The final data-set, which consisted of 23,863 images was divided into training, validation and test subsets of 20,000, 1,863 and 2,000 images respectively. For annotation, Intel's image and video annotation tool, CVAT, was used to help accelerate the process [8].

Four out of the 22 videos that constituted the data-set included two drones instead of one to train the models on multiple detections. The videos were taken from different angles, elevations and lighting conditions to help the models generalize their training and avoid overfitting. The drones appear almost at every location in the image frame, including corners, and at different distances from the camera, from a few centimeters, to a few meters. Figure 2 shows a heat map for the statistical drone location in the image frame in the training set.

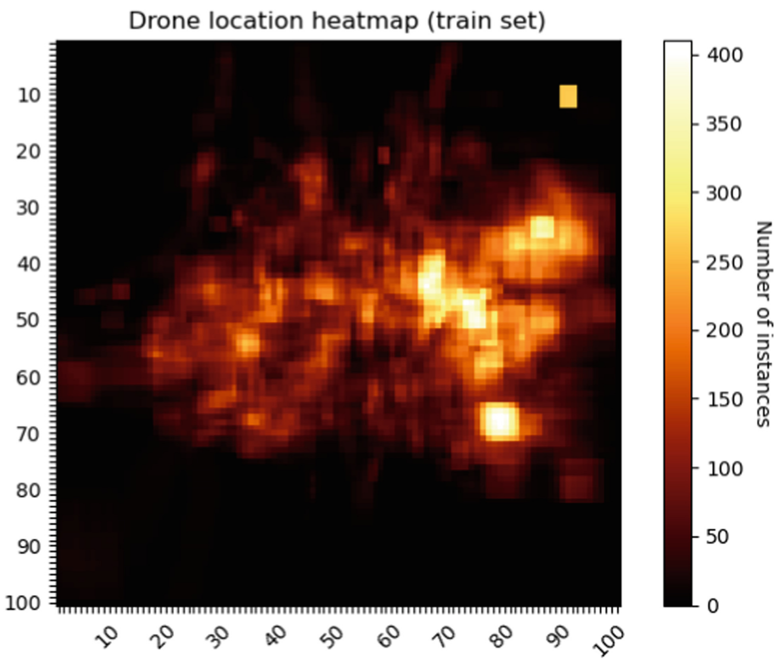

Fig. 2. A heatmap for the drone locations in the training set frames.

4 Results

Table 1 contains a summary of the main results of the three studied models. The first row refers to the size of the square images inputted to the models for training. Although YOLOv4 was tested for three different input sizes, only the results for the 416 input size are included. This is because it is the smallest size tested and the closest to the input size of the other two models.

The second row shows the Average Precision at 50% accuracy (AP50). As mentioned before, this refers to the average precision when the Intersection over Union (or IOU) of the generated bounding boxes is 50% minimum. This is a quite common and accepted standard for evaluating detection models. To be more precise, the mean AP (mAP) is also added in the third row of the table. In this figure, the AP values are evaluated and averaged for IoU values of [0.5:0.95]. In other words, the AP is evaluated at different IoU thresholds, from 0.50 to 0.95 with 0.05 steps, and then averaged to give more realistic values.

As the name suggests, recall refers to the portion of correct predictions relative to all the ground truth predictions that the model should have detected. In other words, while AP takes all model predictions as reference, recall takes all ground truth boxes as its denominator. AR is calculated by determining the recall at different IoU steps from 0.5 to 1 (regardless of the confidence) and then averaging those values [87].

Additionally, the F1 score for the models is computed and appended to the results. The F1 score is a suitable benchmarking metric as it is one number that represents precision and recall at the same time. Finally, the processing speed of the different models is added in the form of the number of frames processed per second on each of the used GPUs.

Table 1. Models' results summary.

Metric	YOLOv4	SSD-MobileNet1	SSD-VGG16
Input size	416	300	300
AP50	86.7%	88.1%	90.4%
mAP	53.0%	51.4%	62.1%
AR	70.3%	66.0%	72.7%
F1-Score	60.4%	57.8%	67.0%
Jetson fps	4	56	58
2080-Ti fps	21	143	170

Table 1 makes it clear that SSD-VGG is the best performance model. It surpasses the other models in every metric. The F1 overall score ranks it first as well. Not only does it possess the highest AP50 rate, but it also maintains formidable performance at more vigorous thresholds, i.e. AP75, mAP and AR. In addition, it is the fastest model with inference speed much higher than the real-time minimum requirement. Figure 3 shows a few samples of the drones from the testset detected by the model.

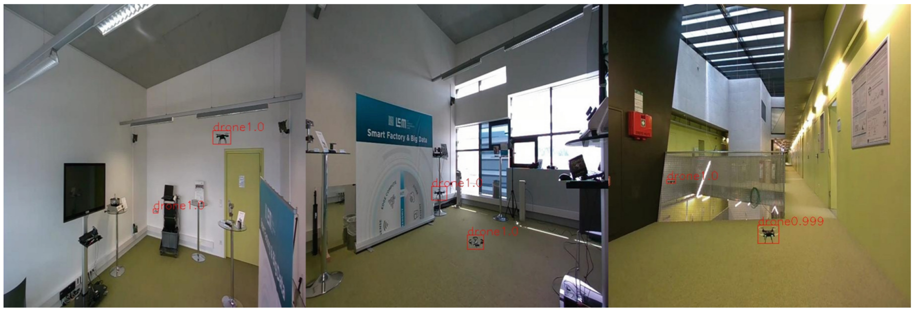

Fig. 3. A sample of the drone detections done by the SSD-VGG network on the testset.

5 Conclusion

In this work, an original drone detection data-set of 23,863 images is collected and annotated. Three DNN detection models, namely YOLOv4, SSD-MobileNetv1 and SSD-VGG16, are retrained and tested on the dataset. The goal was to train a high accuracy drone detection model that is to be used in real-time for surveillance or other purposes. This goal was achieved successfully by the SSD-VGG model, which scored an average precision (AP50) of 90.4%, average recall (AR) of 72.7% and inference speed of 58 frames per second on the NVIDIA Jetson Xavier kit.

References

1. Bitcraze: Home – Bitcraze. https://www.bitcraze.io/
2. Bochkovskiy, A., Wang, C.Y., Liao, H.Y.M.: YOLOv4: optimal speed and accuracy of object detection, April 2020. http://arxiv.org/abs/2004.10934
3. Everingham, M., Gool, L.V., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Homepage. http://host.robots.ox.ac.uk/pascal/VOC/
4. Google: Google Trends. https://trends.google.com/trends/?geo=US
5. Gruber, I.: The Evolution of Drones: From Military to Hobby & Commercial - Percepto. https://percepto.co/the-evolution-of-drones-from-military-to-hobby-commercial/
6. Howard, A.G., et al.: MobileNets: efficient convolutional neural networks for mobile vision applications, April 2017. https://arxiv.org/abs/1704.04861v1
7. Intel: Depth Camera D455 - Intel® RealSense™ Depth and Tracking Cameras. https://www.intelrealsense.com/depth-camera-d455/
8. Intel: openvinotoolkit/cvat: Powerful and efficient Computer Vision Annotation Tool (CVAT). https://github.com/openvinotoolkit/cvat
9. Lee, D.R., La, W.G., Kim, H.: Drone detection and identification system using artificial intelligence. In: Proceedings of the 9th International Conference on Information and Communication Technology Convergence: ICT Convergence Powered by Smart Intelligence, pp. 1131–1133, November 2018. https://doi.org/10.1109/ICTC.2018.8539442

10. Lin, T.Y., et al.: COCO - Common Objects in Context. https://cocodataset.org/#detection-eval
11. Liu, W., et al.: SSD: single shot MultiBox detector. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9905, pp. 21–37. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46448-0_2. https://arxiv.org/abs/1512.02325v5
12. NVIDIA: GeForce GTX 1060 Graphics Cards – NVIDIA GeForce. https://www.nvidia.com/en-in/geforce/products/10series/geforce-gtx-1060/
13. NVIDIA: GeForce RTX 2080 TI-Grafikkarte – NVIDIA. https://www.nvidia.com/de-at/geforce/graphics-cards/rtx-2080-ti/
14. NVIDIA: Jetson AGX Xavier Developer Kit – NVIDIA Developer. https://developer.nvidia.com/embedded/jetson-agx-xavier-developer-kit
15. NVIDIA: Programming Tensor Cores in CUDA 9 – NVIDIA Developer Blog. https://developer.nvidia.com/blog/programming-tensor-cores-cuda-9/
16. Parrot: Parrot Mambo drone downloads – Parrot Support Center. https://www.parrot.com/us/support/documentation/mambo-range
17. Pawełczyk, M., Wojtyra, M.: Real world object detection dataset for quadcopter unmanned aerial vehicle detection. IEEE Access **8**, 174394–174409 (2020). https://doi.org/10.1109/ACCESS.2020.3026192
18. PyTorch: PyTorch. https://pytorch.org/
19. PyTorch: PyTorch documentation - PyTorch 1.9.1 documentation. https://pytorch.org/docs/stable/index.html
20. Syma: SYMA X5SC EXPLORERS 2 - Drone - SYMA Official Site. http://www.symatoys.com/goodshow/x5sc-syma-x5sc-explorers-2.html
21. Taha, B., Shoufan, A.: Machine learning-based drone detection and classification: state-of-the-art in research. IEEE Access **7**, 138669–138682 (2019). https://doi.org/10.1109/ACCESS.2019.2942944