



An Upgraded-YOLO with Object Augmentation: Mini-UAV Detection Under Low-Visibility Conditions by Improving Deep Neural Networks

Tijeni Delleji^{1,2} · Feten Slimeni¹ · Hedi Fekih^{1,2} · Achref Jarray¹ · Wadi Boughanmi¹ · Abdelaziz Kallel² · Zied Chtourou¹

Received: 13 December 2021 / Accepted: 4 August 2022 / Published online: 30 September 2022
© The Author(s), under exclusive licence to Springer Nature Switzerland AG 2022

Abstract

Over the last few years, the manufacturing technology of mini-unmanned aerial vehicles (mini-UAVs), also known as mini-drones, has been experiencing a significant evolution. Thus, the early warning optical drone detection, as an important part of intelligent surveillance, is becoming a global research hotspot. In this article, the authors provide a prospective study to prevent any potential hazards that mini-UAVs may cause, especially those that can carry payloads. Subsequently, we regarded the problem of detecting and locating mini-UAVs in different environments as the problem of detecting tiny and very small objects from an aerial perspective. However, the accuracy and speed of existing detection algorithms do not meet the requirements of real-time detection. For solving this problem, we developed a mini-UAV detection model called Upgraded-YOLO based on the state-of-the-art object detection method of YOLOv5. The proposed model is able to perform real-time tiny/small flying object detection. The main contributions of this research are as follows: firstly, an air image dataset of mini-UAVs was built using a Dahua multisensor camera. Secondly, a strategy of instance augmentation is proposed, in which we added small appearance of mini-drones to samples of the custom air image dataset. Thirdly, in addition to hyperparameter tuning and optimization operations, shallow layers are added to improve the model's ability to detect mini-UAVs. A comparative study with several contemporary object detectors proved that the Upgraded-YOLO performed better. Therefore, the proposed mini-UAV detection technology can be deployed in a monitoring center in order to protect a strategic installation even in low-visibility conditions.

Keywords Mini-UAV · Instance augmentation · Tiny/small object detection · Visibility conditions · Air image · YOLOv5 · Real-time · Dahua multisensor camera

Feten Slimeni and Hedi Fekih are authors contributed equally to this work.

Extended author information available on the last page of the article



Fig. 1 Two examples of mini-UAVs carrying payloads: (a) DJI Agars T16; (b) DJI MATRICE 600

1 Introduction

The International Civil Aviation Organization (ICAO) denotes by “drone” any unmanned aerial vehicle (UAV). Furthermore, the Air Force Special Operations Command (AFSOC) gave additional three names for a drone: a flying robotic system, an unmanned aircraft system (UAS), and a micro air vehicle (MAV) [1, 2]. So, to simplify, an UAV is an aircraft either controlled by a pilot via RF remote control or autonomously following a mission planner through a flight controller. In the same context, the NATO (North Atlantic Treaty Organization) classification [3] and Lykou et al. [4] mention that UAVs weighting between 2 and 25 kg are called mini-UAVs. So, a mini-UAV can carry an operating payload up to 15 kg, e.g., the DJI MATRICE 600 which weighs 10kg is capable of carrying a 6-kg payload for 16 min [5]. Over the last few years, the manufacturing technology of mini-unmanned aerial vehicles (mini-UAVs), also known as mini-drones, has been experiencing a significant evolution. There are multiple usages for mini-UAVs, including precision agriculture for spraying operation, professional aerial photography, and industrial applications [6]. Figure 1 illustrates two types of mini-UAVs that transported higher payloads. Figure 1(a) illustrates DJI Agars T16 mini-UAV equipped with a spray tank that can carry up to 16 L [7]. And, Fig. 1(b) shows DJI MATRICE 600 trying to carry and release a payload. However, the polyvalence of this type of flying gadgets made it accessible to everyone, particularly terrorist groups. Therefore, we can conclude that the detection of mini-UAVs before serious attacks is of the utmost interest.

Consequently, in this work, we will treat the issue of detecting and localizing mini-UAV in diverse environments as a problem of small object detection air image. To set the record straight, an air image or ground-to-aerial perspective image is mostly a picture of a flying object that must include sky background part, taken by a ground-based imaging system, typically used to monitor no-fly zones or restricted areas.

The real-time object detection applied to UAV monitoring is really crucial. Nevertheless, these applications need early detection of objects so that they can be used later as inputs for other reactions. Due to early detection, the appearance of the objects is generally small. In general, the aim of small object detection is to detect

objects that belong to the image and are small in size, which implies that the objects of interest either have a large physical appearance but occupy only a small area in an image, or have a really small appearance [32, 34, 35, 37]. Improvements in object detection algorithms allow faster and more accurate results.

The most recent methods using deep convolutional neural networks (deep CNN) usually involve several steps. First, specify the objects of interest in the image, then go through the deep CNN for feature extraction and afterwards classify them using supervised classification techniques. Finally, mix the results between the objects to properly mark the bounding box. In deep CNN models, there are mainly two types of state-of-art object detectors. The first type is the two-stage detectors, such as faster R-CNN (region-based convolutional neural networks) [8] that uses a region proposal network to generate regions of interests in the first stage, and mask R-CNN [9] that sends the region proposals down the pipeline for object classification and bounding box regression. Such models perform well in terms of accuracy, in particular the faster R-CNN with an accuracy of 73% mAP, but due to their very complex pipeline, they perform poorly in terms of speed with 7 frames per second (FPS), which restricts its application for real-time object detection. The second type of detectors is the single-stage detectors such as SSD (single-shot detector) [10] that runs a convolutional network on input image only once and calculates a feature map, and YOLO (you only look once) [11] that treats object detection as a simple regression problem by tacking an input image and learning the class probabilities and bounding box coordinates. Such models (SSD and YOLO) are proposed by considering both accuracy and processing time.

Especially YOLO performs well compared to previous region-based algorithms in terms of speed with 45 FPS while maintaining a good detection accuracy more than 63% mAP. Although the speed and accuracy were good, YOLOv1 (YOLO first version) [11] made some remarkable localization errors. In other words, the bounding boxes predicted by YOLOv1 are not accurate. So, to overcome the deficiencies of YOLOv1, the creators of YOLO launched YOLOv2 (YOLO second version) [12] where the two limitations of; (i) similarity of the predicted bounding box to the ground truth and (ii) the percentage of total relevant objects correctly classified were resolved without impairing the accuracy of the classification. Moreover, YOLOv2, called also YOLO9000 [12], gained a speed of 59 FPS and mAP of 77.8% in experiments on the PASCAL VOC 2007 dataset [13, 14]. Furthermore, in YOLOv3 (YOLO third version) [15, 16], the main improvement is the addition of multi-scale prediction. In addition, YOLOv3 brought further enhancements in terms of speed and accuracy. In experimenting with the MS COCO [17, 18] dataset, it obtained 33% AP score and achieved a real-time speed of approximately 75 FPS on Tesla V100. In February 2020, Joseph Redmon, the creator of YOLO, stopped researching in the field of computer vision research. However, YOLOv4 (YOLO fourth version) was released on 23 April 2020 and YOLOv5 on 10 June 2020 by other researchers. While YOLOv4 [19, 20] was released in the Darknet framework, YOLOv5 [20–25] has been released in the Ultralytics PyTorch framework. Despite the fact that YOLOv4 can reach 43% AP on MS COCO [26] and 65 FPS speed, the developers of YOLOv5 claim that in a YOLOv5 Colab notebook, running a Tesla P100, they found inference times of up to 0.007 s per image, meaning 140

frames per second (FPS) [24]. In contrast, YOLOv4 achieved 50 FPS after having been converted to the same Ultralytics PyTorch library [21]. Not only that, they also mentioned that YOLOv5 is smaller. Specifically, the YOLOv5 file weights 27 megabytes. However, it weights for YOLOv4 (with Darknet architecture) 244 megabytes. So, YOLOv5 is about 88% smaller than YOLOv4 [52].

The development of new versions of YOLO has not finished. On Oct 28, 2021, Yuxin et al. [53] have launched the YOLOS (you only look at one sequence). It is a series of object detection models based on the vanilla Vision Transformer with the fewest possible modifications and region priors, as well as inductive biases of the target task. In addition, Sahin and Ozer [55] recently released an improved YOLO framework, called YOLODrone, for detecting objects in aerial images taken by drone. To summarize, YOLOv5 claims to be fast, and has a very light model size compared to the contemporary series of YOLO [54].

This paper focuses on detecting mini-drones based on ground to aerial perspective images, more precisely on the AI techniques used for early detection and localization. The goal is to obtain a real-time and accurate deep-CNN object detector which will be able to correctly detect and locate mini-drones that may probably carry a payload, in order to start a neutralization system. The main contributions of this work can be summarized as follows:

1. We collect air images of flying mini-UAVs in a real environment using our Dahua multi-sensor camera [30], the majority of which contain flying mini-UAVs in poor visibility conditions. Subsequently, we build a custom dataset of air images of different types of flying mini-UAVs, called “Mini-UAVs air image dataset,” which provides a benchmark to evaluate the performance of the proposed detection model. The own custom air image dataset will be made public for future research.
2. We come up with a strategy of instance or object augmentation, in which we add tiny/small objects to the samples of our air image dataset. So, in our work, we denote by tiny/small objects, the mini-UAVs that occupy a small portion of the field of view in real world environment.
3. We develop a mini-UAV detection model by redesigning the YOLOv5 object detector [21, 23] from scratch. So, the redesigned model which is named “Upgraded-YOLO” (i.e., assuming that we will upgrade or mutate the model by modifying its internal structure and narrowing its depth in order to adapt it to the detection of tiny/small objects) aggregates more shallow feature information to focus on the detection of small flying objects in air images.

Experimental results show that our proposed Upgraded-YOLO outperforms the general YOLOv5 and other contemporary object detectors in detecting flying mini-UAVs in air images.

The remainder of this paper is organized as follows. We present an instance augmentation strategy in Sect. 3.1.3. A mini-UAV real-time detection algorithm is presented in Sect. 3.2, and the results are discussed in Sect. 4. Finally, Sect. 5 concludes the study.

2 Related Work

2.1 Overview of Low-Visibility Conditions in Aerial Perspective

The relatively high latitude of Tunisia and its geographical stretch from north to south give it contradiction climatic zones: sub-humid in the extreme north, and desert with dusty environment in the south. So, the Tunisian climate is characterized by a number of meteorological phenomena or meteors (see Fig. 2) which will consequently disturb the visibility and therefore have an impact on video surveillance and the detection of flying drones from the ground.

Under such conditions, many computer vision and image processing algorithms suffer from the visibility degradation, since most of them treat clear scenes under good weather. Therefore, in our work, we took these phenomena into consideration when constructing our training air image dataset (that contain flying mini-UAVs).

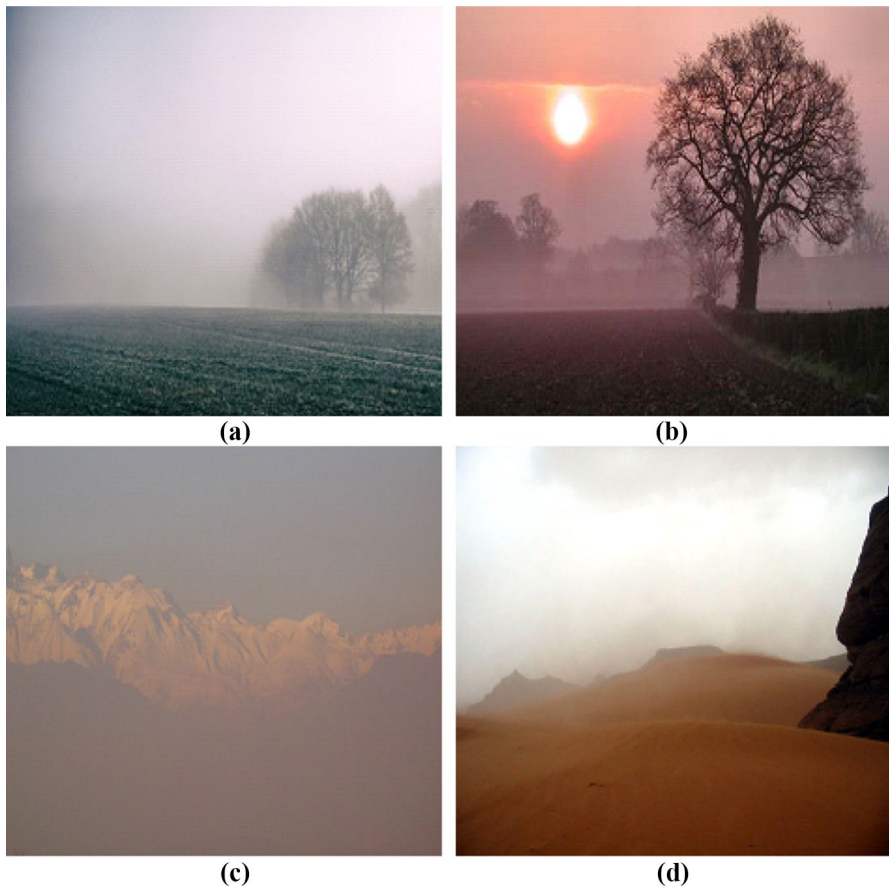


Fig. 2 Examples of phenomena observed in the Tunisian atmosphere. Most of them exhibit significant visibility degradation: (a) fog, (b) mist, (c) haze, (d) duststorm

Among these phenomena, the following are particularly noteworthy: (a) The fog is produced by the suspension in the atmosphere of very small water droplets or ice particles. This phenomenon reduces horizontal visibility at the earth's surface to less than 1km, and the humidity is close to 100% (see Fig. 2a). (b) The mist is produced by microscopic water droplets suspended in the atmosphere. The mist appears grayish and the visibility is between 1 and 5 km (see Fig. 2b). (c) The haze is the suspension in the atmosphere of dry, extremely small, and invisible particles which is the result of fumes or airborne dust, sand, or even sea salt particles. It appears bluish on a dark background and yellowish on a light background. Visibility in the haze is between 1 and 5 km, and relative humidity can reach 60% (see Fig. 2c). (d) The dust-storm is caused by dust or sand particles being powerfully lifted from the ground by a strong, turbulent wind, usually reducing visibility to less than 1km (see Fig. 2d).

2.2 Issues in Object Detection

The deep detectors generally consist of two parts: one is a skeleton pre-trained on ImageNet, and the other (called the head) is the main part used to predict the category and bounding box of the object. In addition, object detectors developed in recent years usually have some layers inserted between the skeleton and the head, and usually used to collect feature maps at different stages. We can call it the neck of the object detector [21, 24]. So, the detector needs to meet the following conditions:

1. Higher input network scale (resolution) used to detect multiple small objects;
2. Higher layers: higher receptive fields to cover the expanding scale of the input network;
3. More parameters improve the model's ability to detect multiple objects of different sizes in a single image.

In summary, the general object detector consists of the parts presented by Fig. 3.

Despite of these works, research in this area is far from complete and many difficulties remain. An interesting summary of some of the challenges is presented in the review by Agarwal et al. [28].

- Scale variance: the variation in size of the objects to be detected represents an important lock, especially when the gap is large. The image pyramid approach

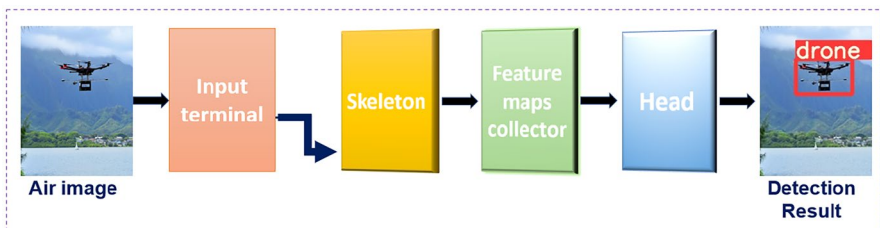


Fig. 3 Concept of architectural object detection for aerial perspective image

is one of the oldest and most effective methods for detecting objects at different scales. The disadvantage of this operation is the high computational cost of repeating the convolutions after each scaling step. The data augmentation method could be used as an alternative to enrich the training examples by applying transformations on the original images. However, the most used approach by state-of-the-art detection models (e.g., yolov5) is called anchors. Anchors are pre-calculated bounding boxes at different scales and aspect ratios, which are provided as a reference during training to ensure detection at different scales.

- Rotation variance: often solved by data augmentation by applying rotations to the training set images, thus showing the network rotated examples. This solution is limited to the rotation resolution of rigid objects (e.g., a drone) and does not apply to deformable objects (e.g., a cat).
- Domain adaptation: Most of known detection networks are pre-trained on huge datasets like ImageNet or COCO [17]. The models thus have a very high generalization capacity except in very specific use cases where the content is very different. So, domain adaptation application is therefore necessary. The most widely adopted solution is transfer learning, which re-train the network by changing its top layers with a small set of data corresponding to the task in question.
- Occlusions: This problem, which exists in most applications, is an issue since some of the information is hidden. Thus, providing examples containing occlusions in the training dataset may partially solve the problem but will not represent all forms of occlusion.
- Small objects: detecting small objects is more difficult than detecting medium or large ones. This is due to many factors such as lack of associated information, inaccurate localization, and confusion of objects with the background image. So, to overcome this problem, solutions vary in terms of complexity from simple scaling to the use of surface networks, coarse and fine networks to a super-resolution method that could be implemented with a GAN learning to represent small objects with higher resolutions. In addition, low image resolution could cause the same problems and thus require a super-resolution method.

2.3 Visualization of YOLOv5 Network Structure

YOLO is a technique based on regression. Instead of selecting the relevant part of an image, it predicts classes and bounding boxes for the entire image in a single run of the algorithm. So, the idea of YOLO is originated from the extension of the basic CNN idea for classification and detection tasks. The YOLO series (from YOLOv0 to YOLOv7) is a regression method based on deep learning. So, the latest release of YOLOv5 that we have modified in this work [20–24] is basically modified on the structure of YOLOv3 [15].

As shown in Fig. 4, the YOLO series architecture is divided into three functionally different parts, called backbone network, neck network, and head or detect network [19, 24]. This is a division found in the architecture of many recent image detection models [29].

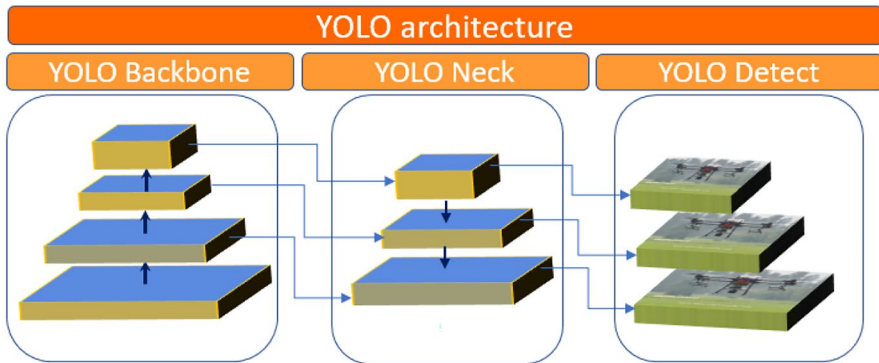


Fig. 4 Basic architecture of the YOLO series network presented as backbone, neck, and detect (head)

The backbone part is a convolutional neural network which extracts feature information from the input image by multiple convolution and pooling. So, it aggregates different fine-grained images and forms image features.

The backbone is the body of the network, which will enable all the decisions made by the network. In simple terms, it can be seen as a converter that converts the input image, a data format difficult to process by AI (artificial intelligence), into a set of information characterized by some features (such as the presence of shapes, colors, textures, ...) from which it is easy to recognize objects. It is thus composed of a series of successive layers.

As shown in Fig. 5, there are four different layers: focus structure, CBNS (convolution, batch normalization, and SiLU activation function), C3B (Bottleneck with 3 CBNS), and SPP (spatial pyramid pooling).

The backbone is usually trained separately on image classification competitions such as the ImageNet challenge [56], which include hundreds of thousands of images with a wide range of content such as animals, vehicles, and plants. This diversity of content forces the backbone to learn a wide variety of features in terms of size, color, and shape of the elements it observes and thus be more robust and able to extract useful features regardless of the image presented to the backbone.

The second part of Yolov5 is the neck network, a series of feature aggregation layers of mixed and combined image features. As shown in Fig. 5, there are four different layers in this part: CBNS, C3B, concatenation, and upsampling. The neck has the role of extracting the relevant features from all the layers of the backbone, and combining them into useful features for our detection task. Indeed, not all the layers included in the backbone learn the same information: the first set of layers, generally of higher spatial resolution, will detect features that are often simpler (the presence of lines, colors) and smaller. The last set are the lower resolution layers that tend to provide more complex features (e.g., the combination of specific shapes and colors such as a metal circle with a hole for a car rim) and large objects. The neck makes it possible to integrate and combine features of different resolutions and complexities, to allow detection of small and large, simple and complex features.

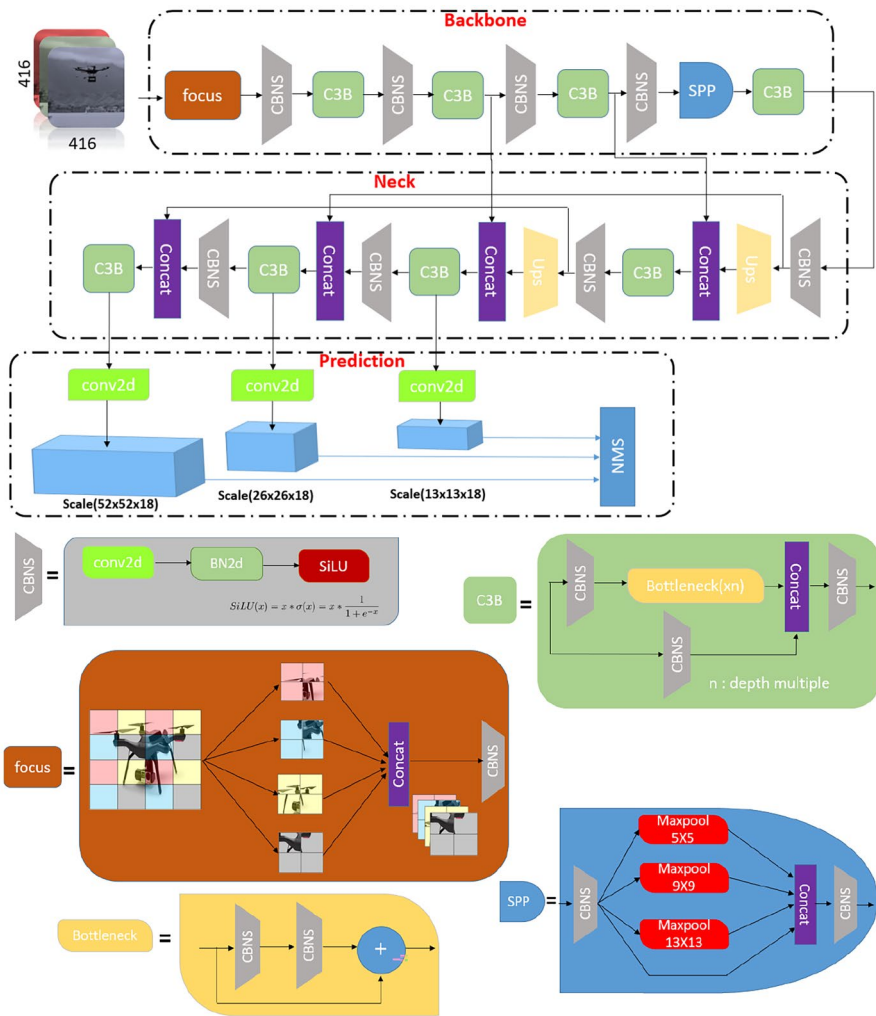


Fig. 5 YOLOv5 structure diagram

Finally, the head is responsible for the final decision of the network. Based on the information provided by the neck, it will detect the elements of interest by drawing bounding boxes around them and it will, furthermore, give the nature of every object present in each bounding box.

In terms of general architecture, Yolov5 is similar to its predecessors Yolo and other models in the literature. It is therefore time to see the real reason for the difference in performance. The bag of freebies is a set of enhancements with no impact on the architecture of a network, which can be used free of charge, with no cost of modification on an existing network. It gathers all the improvements that can be applied during the network learning such as the loss function, data augmentation, and cross mini-batch normalization. The bag of specials is, on the contrary, a bag containing

improvements that require specific modifications to the architecture of a network. It contains recent advances in the scientific literature that improve the performance of the network without decreasing its speed [23, 27, 29].

3 Method and Dataset

3.1 Custom Air Image Dataset Construction

To construct our own air image dataset, called “Mini-UAVs air image dataset,” we proceeded through the following steps: data collection, data augmentation, object augmentation, and data annotation.

3.1.1 Methodology of Collection

The dataset is collected using internet videos and our Dahua multisensor camera [30], mainly including rotor mini-UAV, like four-rotor UAV (i.e., DJI-Phantom4, DJI-Marvic), and six-rotor UAV (i.e., DJI-Matrice 600, DJI Agars T16). Some samples are shown in Fig. 6. A total of 4560 sample images are used in this experiment which are divided, randomly, into 3100 images for training and 1460 images for testing purposes.

3.1.2 Data Augmentation

Data augmentation is a technique that allows researchers to greatly increase the variety of data that is available for model training, without the need to gather new data. Thus, the purpose of data augmentation in the training dataset is to create diversity and overcome overfitting by artificially increasing the training samples [28]. In our work, traditional data augmentation methods such as adding noise, cropping, flipping, rotation, brightness, and contrast are used. Moreover, another technique is proposed, in which a number of meteorological phenomena such as dust, mist, and fog (see Subsect. 2.1) are added to images of our dataset (see Fig. 7).

3.1.3 Object Augmentation Strategy

The detection of tiny/small objects in an image or video is a research topic in computer vision that tries to develop technologies and techniques in order to detect lowercase instances [31]. The possibility of appearance of tiny/small objects is much more numerous than those of other objects due to their limited size, which creates confusion in the detector to locate these objects among several others located in the vicinity or having the same size (or appearance). Thus, it is difficult to distinguish tiny/small objects from the background. In general, a tiny/small object has two definitions [32–36]. The first is related to object dimension in the real world while the other is related to a threshold on the surface occupied by the object in the image [32–34, 37]. Although all modern detection models are effective for medium and large objects, they are not very efficient in detecting small objects. For example,



Fig. 6 Sample images of own custom “Mini-UAVs air image dataset”

it is really hard for a model to see a micro/mini-drone flying from 2000m away. This is because there are some obstacles in tiny/small object detection. First, lowcase objects need appearance information required to distinguish them from background or similar classes. Second, the places of low case objects have a lot of possibilities. That is to say, the required precision for accurate localization is higher [35]. So, in this paper, we are going to demonstrate how we can improve the performance of a detector to detect and classify small flying objects. In our contribution, based on the following references [32–39], we have grouped the flying objects according to the following distribution shown in Table 1.

The size of objects is measured as the number of pixels in each of the bounding boxes that describe the spatial location of a mini-UAV. To push our model to focus more on tiny/small objects, we perform an augmentation based on the copy-and-paste strategy that increases the number of tiny objects in each image of our dataset.

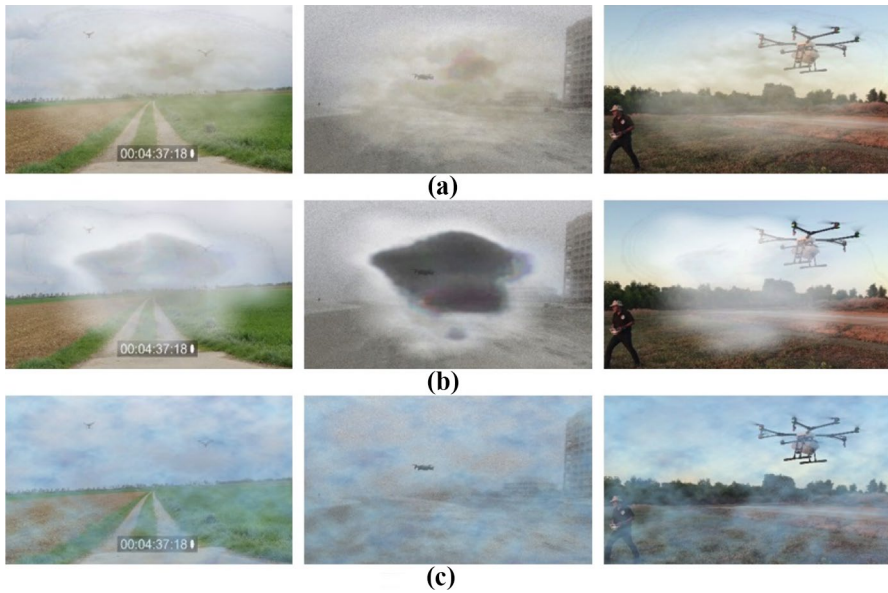


Fig. 7 Sample images of the Mini-UAVs air image dataset after data augmentation with meteorological phenomena usually observed in Tunisian climate. (a) Dust. (b) Mist. (c) Fog

As mentioned in Fig. 8, our augmentation strategy consists of two phases. The first phase consists in finding different appearances of tiny/small objects (micro/mini-drones in flight with low appearance) and generating their masks. The second phase consists in searching, in our dataset, the images that contain objects of bounding box (BB) size $< 32^2$.

Thus, the tiny/small objects that are prepared in the first phase will be pasted in random places of these images, as presented in Fig. 9, generating the labels of these new tiny/small objects in an automatic way.

Figure 10 shows the percentages (portions) of mini-UAVs (flying objects) of different sizes, for our own custom dataset: before and after object augmentation. So, this figure shows our contribution related to the increase of tiny/small objects in the air images of the “mini-UAVs dataset” (train and test sub-datasets).

As shown in Fig. 10, in the train and test datasets, there are more tiny/small objects than medium and large objects. Approximately more than 50% of objects are

Table 1 Size distribution of the mini-UAVs in air images by subheadings

Object	Size in pixels
Tiny	From 8^2 to $< 16^2$
Small	From 16^2 to $< 32^2$
Medium	From 32^2 to $< 96^2$
Large	$> 96^2$

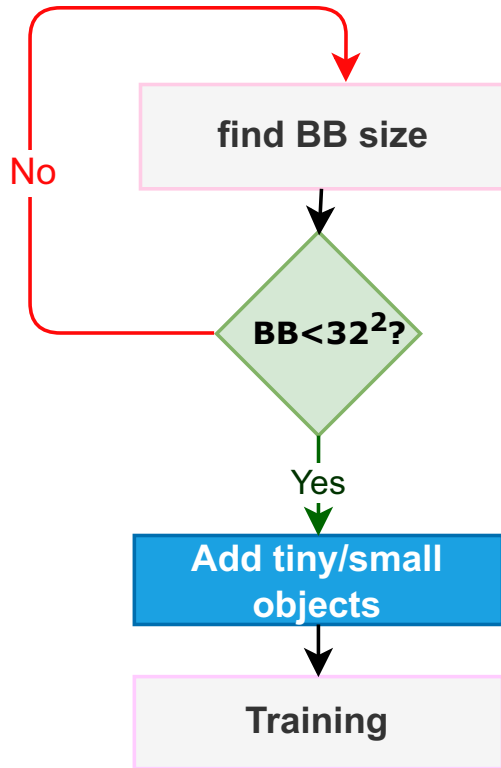


Fig. 8 Workflow of adding tiny/small instances; BB: bounding box contains an object

tiny ($area < 16^2$), more than 16% are small ($16^2 < area < 32^2$), more than 4% are medium ($32^2 < area < 96^2$, and less than 3% are large ($area > 96^2$).

3.2 Upgraded-YOLO for Mini-UAV Object Detection in Air Images

3.2.1 Model Architecture

In order to implement an optical early warning detection system, a flying target (i.e., unauthorized mini-UAV), which necessarily has a small or even tiny appearance, must be detected. The size of distant mini-UAVs in the sky background is very small; and the receptive field size of YOLOv5 is not enough to detect these tiny flying objects. This is the reason of improving the architecture of YOLOv5. As shown in Fig. 11, we did two improvements to the original YOLOv5 architecture:

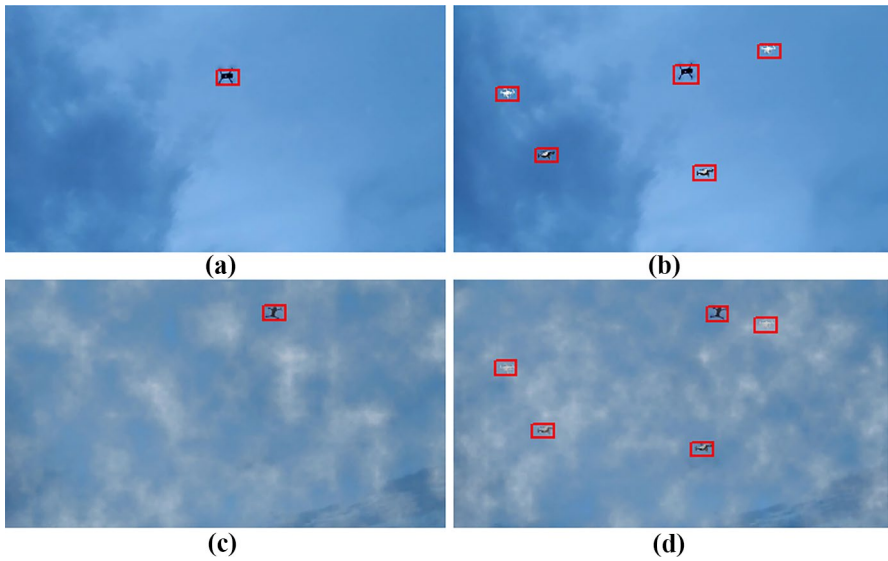


Fig. 9 Sample images of our dataset before (a and c) and after (b and d) tiny/small object augmentation

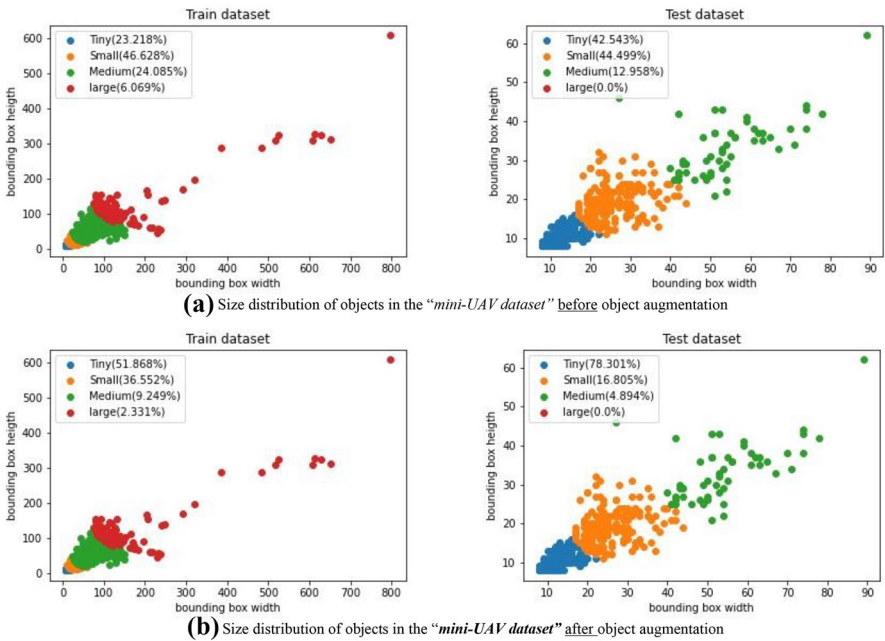


Fig. 10 Size distribution of tiny/small objects in air images before (a) and after (b) tiny/small object augmentation

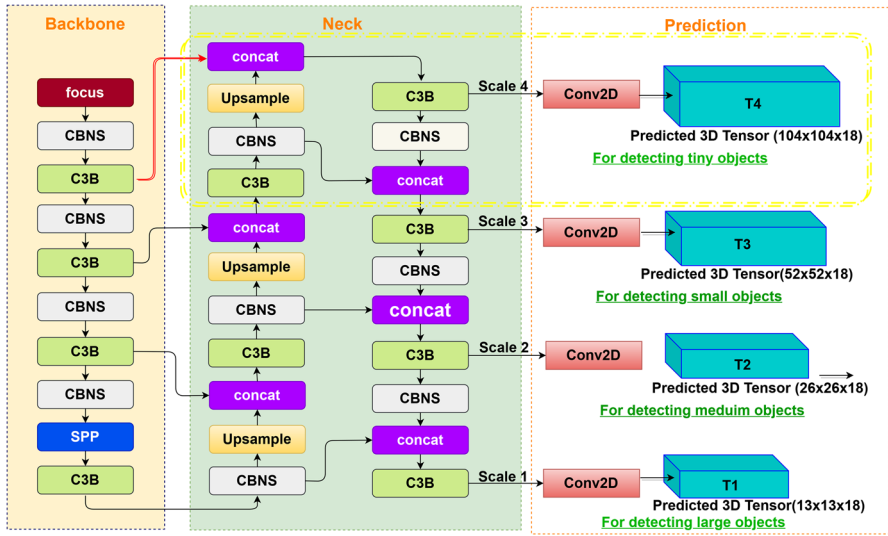


Fig. 11 Anatomy of the Upgraded-YOLO for mini-UAV detection in air images

- i. A fourth scale (marked with dashed yellow rectangle in Fig. 11) is added to the three scales of YOLOv5 feature maps to capture more texture and contour information of tiny/small objects as mini-UAVs.
- ii. Feature maps from the backbone network are brought into the added fourth scale (represented by the red line) to reduce feature information loss of mini-UAVs.

The YOLOv5 final part consists of three detection tensors. So, YOLOv5 applies 8, 16, and 32 downsampling of the initial image to detect objects at different resolutions. For example, given an image of resolution $416\tilde{A}\text{---}416$ as input to YOLOv5, features of input are extracted by the backbone network of YOLOv5. To precisely detect different sizes of the targets, 3 different scales of boxes are predicted, which are expressed as T1, T2, and T3 in the followed framework. In our mini-UAVs, air image dataset experiments, we predicted 3 boxes at each scale, which means the tensor is $N\tilde{A}\text{---}N\tilde{A}\text{---}[3*(1+1+4)]$ for 1 class probability (confidence score), 1 class (mini-UAV) predictions, and 4 surrounding box position coordinates. Here, N is feature maps size of T1, T2, and T3, which are 13, 26, and 52, respectively. So, the problem of lacking appearance information is related to different image resolutions. For example, if the image resolution is low, it may prevent the detector from detecting very small objects. In these cases, the information needed to detect very small objects will be very limited. Indeed, in YOLOv5, if the object of interest occupies a size of $8*8$ pixels on an image with a resolution of $416*416$, then it will be represented by only one pixel in the final feature maps. Therefore, any object smaller than $8*8$ will be disappeared. Subsequently, this architecture of YOLOv5 is insufficient for the detection of tiny objects. Therefore, the main idea of our proposal is to add

a detection level (scale 4 in Fig. 11) with a high resolution that is able to extract more features for tiny objects. For this purpose, we added a level that reduces the resolution only four times (i.e., the input image was downsampled with a stride of size 4). In fact, our proposed architecture aims in detecting tiny objects, that is why we have added a higher resolution detection level that generated a tensor T4 of size $104*104*18$. The addition of the later consists of adding seven layers as indicated in Fig. 11 by a yellow box, of which the upsample layer increases the resolution and then the output of this layer will be concatenated with the output of layer three of the backbone part. In addition, the connection represented by the red line is added to bring the feature information from the backbone network into the added fourth scale of the neck network. Based on the idea of residual networks, this connection can improve gradient back propagation, to prevent the gradient from being erased, and reduce the loss of the feature information of very small flying objects.

3.2.2 The Loss Function

The loss of YOLOv5 [22] is a multi-task loss that contains three terms: the first for localization loss or bounding box regression loss (denoted as $Loss^{box}$), the second for classification loss (denoted as $Loss^{class}$), and the third for object loss or confidence loss (denoted as $Loss^{obj}$) [29]. The total loss (Loss Total) can be written as:

$$Loss^{Total} = Loss^{box} + Loss^{obj} + Loss^{class} \quad (1)$$

Next, we will go into the details of the three losses we used in the proposed Upgraded-YOLO network [40].

$$Loss^{box} = \lambda_{box} * \sum_{i=0}^{S^2} \sum_{j=0}^B L_{i,j}^{obj} * (1 - CIoU_{pred_{box}}^{truth_{box}}) \quad (2)$$

$$Loss^{obj} = -\lambda_{obj} * \sum_{i=0}^{S^2} \sum_{j=0}^B L_{i,j}^{obj} * [\alpha * P_{truth}^{obj} * (1 - P_{pred}^{obj})^\gamma * \log(P_{pred}^{obj}) + (1 - \alpha) * (1 - P_{truth}^{obj}) * (P_{pred}^{obj})^\gamma * \log(1 - P_{pred}^{obj})] \quad (3)$$

$$Loss^{class} = -\lambda_{class} * \sum_{i=0}^{S^2} \sum_{c \in K} L_i^{obj} * [\alpha * P_{truth}^c * (1 - P_{pred}^c)^\gamma * \log(P_{pred}^c) + (1 - \alpha) * (1 - P_{truth}^c) * (P_{pred}^c)^\gamma * \log(1 - P_{pred}^c)] \quad (4)$$

where λ_{box} , λ_{obj} , λ_{class} are hyperparameters or scalars to weight each loss function,

- B is the number of bounding boxes predicted for each tile,
- S^2 is the number of cells (grids) that input images are divided into,
- K denotes the number of classes,

- P_{truth}^c equals 1 if the ground-truth belongs to the i -th class and 0 otherwise (binary indicator),
- P_{pred}^c is the predicted probability for the i -th class,
- P_{truth}^{obj} equals 1 if the ground-truth bounding box belongs an object (drone) and 0 otherwise,
- P_{pred}^{obj} is the probability the predicted bounding box contains an object inside,
- $\gamma \in [0, +\infty]$ is a focusing parameter or a modulating factor,
- $\alpha \in [0, 1]$ is a balancing parameter, is also useful for addressing class imbalance,
- The loss is similar to categorical cross entropy, and they would be equivalent if $\gamma = 0$ and $\alpha_i = 1$,
- Here $L_{i,j}^{obj}$ and L_i^{obj} are indicator functions such that: $L_{i,j}^{obj} = 1$ if box j and cell i are matched together, 0 otherwise (1 if object appears in cell i and j -th box detects it, 0 otherwise), $L_i^{obj} = 1$ if cell i has an object present, 0 otherwise.
- The $CIOU_{pred\ box}^{truth\ box}$ is called the complete-IoU between the predicted box and the ground-truth box [41].
- $(1 - CIOU_{pred\ box}^{truth\ box})$ is the complete IoU loss which ensures three geometric measures, i.e., overlap area, central point distance, and aspect ratio [41].

4 Experimental Results and Evaluation

4.1 Experimental Setting

Experiments in this paper have been performed using the machine learning framework PyTorch 1.9. At the beginning of our work, training trials were performed, with 100 epochs, on the Kaggle platform with a GPU NVIDIA TESLA P100, 16 GB of memory, Driver version: 450.119.04, and CUDA version: 11.0. The neural network training and testing were performed on a workstation equipped with an AMD Ryzen 9 5900X 12-Core Processor 3.70 GHz, NVIDIA Geforce RTX 3070 GPU, and NVIDIA TESLA T4 GPU. The model building, training, and result testing are all completed under the V1.9 of PyTorch framework, using the V11.1.0 of CUDA parallel computing architecture and at the same time integrating the v8.2.2 of cuDNN acceleration library into the PyTorch framework to accelerate computer computing capabilities. The ADAM optimizer with a learning rate of 0.001 was used for the training optimization. The training was performed with an input image size of 416*416, a batch size on a GPU of 8 frames, and a number 8 of dataloader's workers.

4.2 Evaluation Metrics for Mini-UAV Detection

The following standard criteria are exploited to quantitatively evaluate and compare the detection accuracy: Intersection over union (IoU) is one of the most used tools in machine learning to measure the accuracy of an object detection model. Moreover, this criterion compares the detected predicted region with the ground truth region in a way that is proportional to the size of the object being searched. Also, the regions

being compared can be bounding boxes that locate drones. Therefore, the overlapping ratio between the detected box (B_D) and the ground truth box (B_{GT}) is the measure of IoU [42–44].

$$IoU = \frac{|B_D \cap B_{GT}|}{|B_D \cup B_{GT}|} \quad (5)$$

For object detection, IoU is used to determine how many objects were detected correctly and how many false positives were generated. Generally, a $0.5 * IoU$ ratio for each prediction at the training stage is targeted. This means that if the network predicts an object with a detected box that overlaps with the ground truth box by at least 50%, it is considered as a true prediction. By defining the true positive (TP) as the number of correct detections with $IoU > 0.5$, the false positive (FP) as the number of false detections (like a bird that was detected as a drone) or detected more than once, and the false negative (FN) as the number of drones that are not detected or detected with $IoU \leq 0.5$, the precision and recall scores [45], which are used to measure the performance of a detection model, are calculated as:

$$P = \frac{TP}{TP + FP} \quad (6)$$

$$R = \frac{TP}{TP + FN} \quad (7)$$

where precision shows how accurately the model has detected the drones. Recall is described as the number of truly detected drones over the sum of truly detected drones and undetected drones in the image. In order to properly evaluate the performances of our object detector, average precision (AP) has been used, with precision (P) and recall (R). From all these indicators, it is now possible to draw the precision curve as a function of the recall. It will allow, thanks to the computing of the area under this curve, to define the average precision (AP) of the proposed model.

$$AP = \int_0^1 P(R) dR \quad (8)$$

Therefore, the mean average precision (mAP) is defined as the mean of AP across all categories (M):

$$mAP = \sum_{i=1}^M \frac{AP_i}{M} \quad (9)$$

If the IoU threshold has been set to 0.5 or 50%, the mAP is called $mAP_{0.5}$ or $mAP@50$. $mAP_{0.5} : 0.95$ means mAP with $0.5 < IoU < 0.95$.

4.3 Results

4.3.1 Parameters and Hyper-Parameters Optimization

Our proposed model has two different types of parameters:

Model parameters or network parameters: these are related to the size and topology of neural networks which have an influence on model performance. Therefore, for our model, they are initialized randomly (from scratch) to avoid symmetry, which could potentially affect the training process, and learned during model training.

Hyper-parameters are parameters that have an influence on the speed and quality of the learning process such as learning rate and decay. The determination of these hyper-parameters can be done manually by trying all possible values. But this is very time consuming as the number of possible combinations is very high. Therefore, in our work, we have applied the hyperparameter evolution that uses a genetic algorithm (GA) to automatically find the optimal hyper-parameter [46, 47]. Thus, crossover and mutation are the main genetic operators in the GA. So, the mutation is used, with a probability of 90% and a variance of 0.04, to create new offspring based on a combination of the best parents from all previous generations [48].

To evolve the hyper-parameters of our model (Fig. 12), we trained it, for 50 epochs, 300 times (300 generations) by maximizing the fitness score, which is defined as follows:

$$q_f = 0.1mAP_{0.5} + 0.9mAP_{0.5} : 0.95 \quad (10)$$

4.3.2 Experimental Analysis and Discussion

Figure 13 shows performance metrics and loss functions change curves, for the training validation. These curves correspond to the baseline (YOLOv5 small) and the Upgraded-YOLO, both trained on the Mini-UAVs air image dataset. The loss function indicates the performance of a given predictor in detecting the input data points in a dataset. The smaller the loss, the better the detector is at modeling the relationship between the input data and the output targets.

Line plots created in Fig. 13 show two different types of loss. Those represented in Fig. 13(a), (b) are associated to both of the losses related to the given cell containing an object during the training: the confidence loss or objectness loss (*obj_loss*) and the predicted bounding box loss (*box_loss*). In other words, the *box_loss* represents how well the model can locate the center of an object and how well the predicted bounding box covers an object, while the objectness loss determines whether there are objects in the predicted bounding box. Based on Fig. 13(a), we can conclude that as the number of iterations gradually increases, baseline and Upgraded-YOLO algorithm curves gradually converge, and the loss values become smaller and smaller. When the two models are iterated 250 times, the loss values are basically stable and the network basically converges. On one hand, Fig. 13(a) shows that the baseline (blue curve) had the smallest objectness loss. Thus, for 300 training epochs, the loss of confidence in the baseline is 0.0007, whereas for the Upgraded-YOLO (red curve) it reached 0.0014. So, when we added a fourth level to

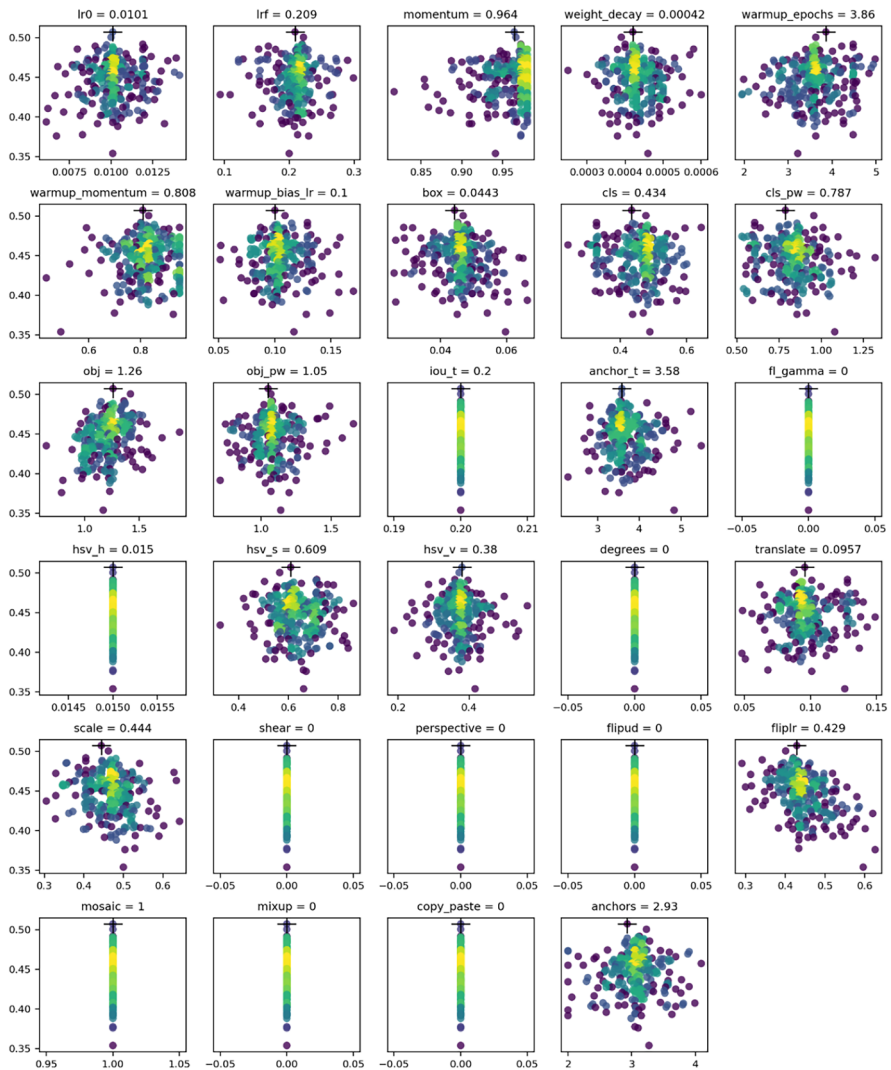


Fig. 12 Fitness (y axis) vs hyper-parameter values (x axis) of the base scenario trained on “mini-UAV dataset.” Yellow indicates higher concentrations. Vertical distributions indicate that a parameter has been disabled and does not mutate

our proposed model, we actually increased the number of detected object parts. Consequently, this leads to an increase in the number of bounding boxes predicted by each cell. Moreover, as mentioned by the authors in reference [15]: “If a bounding box prior is not assigned to a ground truth object it incurs no loss for coordinate or class predictions, only objectness”; it means that all the box predictions contribute to the objectness loss which is an accumulation of losses related to each given cell containing an object during the training, whereas the Upgraded-YOLO had a slightly superior loss.

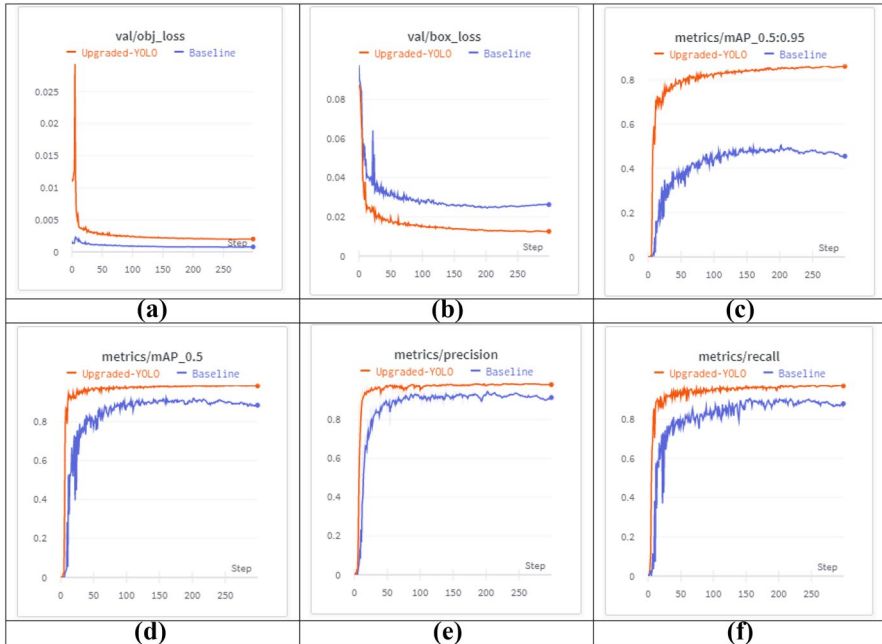


Fig. 13 Comparison of common evaluation indicators and loss curves between baseline (YOLOV5), given in blue curve, and Upgraded-YOLO, given in red curve. The first two plots are (a) bounding box loss (measured by CIoU) and (b) confidence loss in the validation dataset. The remaining four curves represent performance metrics of object detection task, and they are (c) mAP_0.5:0.9, (d) mAP_0.5, (e) precision, and (f) recall

On the other hand, Fig. 13(b) shows that the box loss value of our Upgraded-YOLO was much lower than that of the baseline model because the mutation (the changes in the anatomy of YOLO to detect tiny/small objects) of our proposed model has brought an improvement in terms of prediction of the bounding box that covered the target object. So, the mutation of YOLO, by injecting a neural structure capable of detecting the most part of an object, reduces errors between the ground truth and the predicted bounding box. Hence, we have pushed the box loss to better teach the network to predict a better CIoU. Since only the best-fitting boxes in each spatial cell contribute to the box loss, we get better results with our model. Also, it can be seen that the number of true positive (TP) or the number of detections with *IoU* > 0.5 is augmented. Hence, the better results were obtained by our model in Fig. 13(c)–(f). So, the general trend of performance metrics (*mAP*_0.5 : 0.9, *mAP*_0.5, precision and recall) was roughly the same for the two models (Upgraded-YOLO and the baseline). But, the curves of the baseline have a jitter range almost during the whole training process. However, the timing of jitter for the Upgraded-YOLO during the early stage was shorter than that of the baseline, with a smaller jitter amplitude. Generally, the observation of the jitter phenomenon during the training of a detection model is due to the added noise, and it is due to local minima. So, every time the optimizer converges towards the local minimum, the performance metrics increases. But with good learning rate, the model learns to jump from these points and the optimizer will

converge toward the global minimum which is the solution. Hence, our model is better for the convergence towards the global solution. In terms of speed, the Upgraded-YOLO is faster than the baseline; e.g., for $mAP_{0.5}$, our model reached 91% after 11 epochs; however, the baseline did not exceed 5%. In terms of accuracy, the curves of the evaluation metrics show that our model is more accurate. Furthermore, the mAP is used to measure the quality of the detection model. Thus, the higher the value is, the higher the average detection accuracy and the better the performance will be. Moreover, the high mAP also denotes a great performance of the training models. So, Fig. 13(c) shows that from the start, the $mAP_{0.5} : 0.9$ of our model is higher than that of the baseline. Moreover, the $mAP_{0.5} : 0.9$ of our model reaches about 86% after 300 epochs, while the baseline reached 47% after 150 epochs and started overfitting until epoch 300, with a $mAP_{0.5} : 0.9$ of 45%. As the graphs in Fig. 13(d) show, the $mAP_{0.5}$ of our model reached 92% whereas the baseline reached 88%. As shown in Fig. 13(e), the precision or the exactness of the Upgraded-YOLO reached 0.98 after 300 epochs; however, for the baseline it did not exceed 0.91. The recall tells us what proportion of objects was predicted to be a mini-UAV. The curves of Fig. 13(f) show that the completeness of our model is better than the baseline. The improvement in precision and recall rate of our model compared to the baseline can be attributed to the increase in the true positive (the number of correctly classified objects). Therefore, an improvement in prediction accuracy can be seen.

Furthermore, Table 2 shows mAP , precision, and recall of four models: YOLOv3-tiny, YOLOv3, YOLOv5, and ours after a training on our air image dataset. It can be seen that, after 300 epochs, our method has better performance.

Compared with the results of the baseline (YOLOv5 model), the precision of the Upgraded-YOLO model is increased by 6.84% and the recall rate is increased by 9.04%. Moreover, the $mAP_{0.5:0.95}$ is increased by 40.57% and the $mAP_{0.5}$ has improved by 9.9%. These results confirm what was mentioned at the beginning of this interpretation, that the performance of our model is higher than that of the baseline.

Moreover, Table 3 shows that after 300 epochs of training, our model has the lowest total loss value, which makes it more accurate and perform better than the three contemporary object detectors: YOLOv3-tiny, YOLOv3, YOLOv5.

To highlight the performance of our detector, we compare it to the baseline. The results of the test are based on 400 frames from YouTube video sequences captured in an outdoor environment with different drone models, and from

Table 2 Model performance evaluation with the custom air image dataset

Mini-UAV detection model	Performance metrics			
	$mAP_{0.5}$	$mAP_{0.5:0.95}$	Precision	Recall
YOLOv3-tiny	0.4312	0.6936	0.7827	0.6989
YOLOv3	0.4641	0.8811	0.8982	0.8691
YOLOv5	0.4549	0.8846	0.912	0.8789
Upgraded-YOLO (ours)	0.8606	0.9836	0.9804	0.9693

Table 3 Comparison of loss functions between our model and other contemporary detectors

Mini-UAV detection model	Loss function		
	<i>Box_loss</i>	<i>Obj_loss</i>	Total loss
YOLOv3-tiny	0.07621	0.0127704	0.08898
YOLOv3	0.02878	0.0008904	0.0301704
YOLOv5	0.02621	0.0007704	0.0269804
Ours	0.01257	0.001986	0.014556

visible video clips shot with our Dahua multi-sensor camera. An illustration of the detected results related to both of the baseline model and the Upgraded-YOLO for some samples in air images (i.e., ground to aerial perspective images) is shown in Fig. 14 where the red and green bounding boxes correspond to detections by the Upgraded-YOLO detector and the baseline detector, respectively.

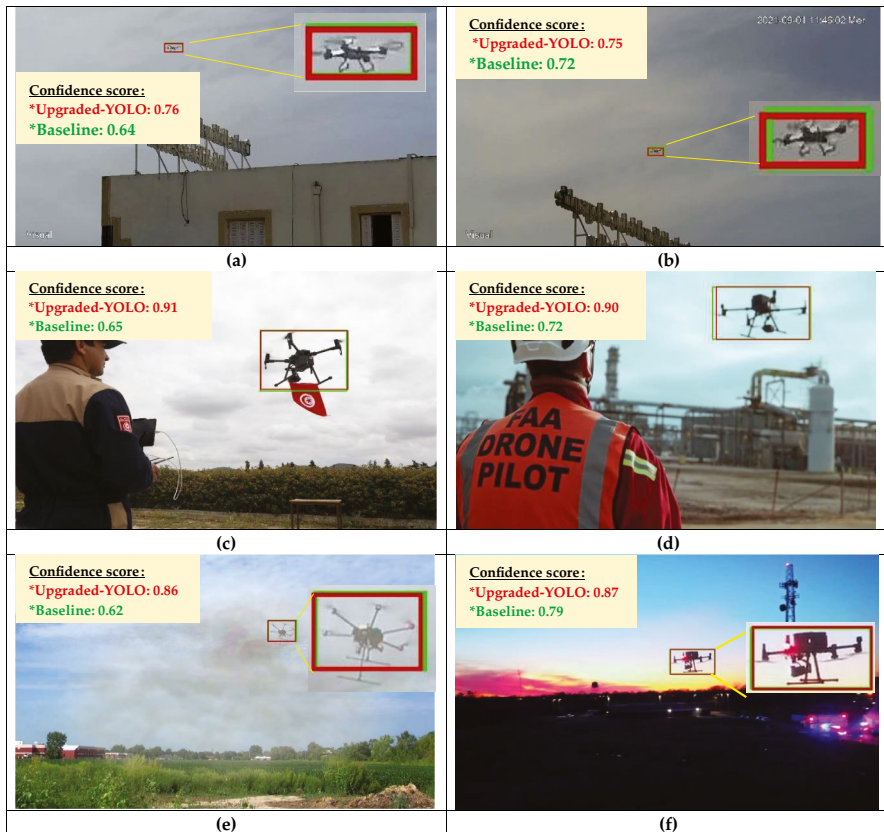


Fig. 14 Comparison of the detection results in air images at diverse distances and with different visibility conditions: (a) and (b) mini-UAVs in very small appearance; (c) and (d) mini-UAVs with medium appearance; (e) and (f) mini-UAVs flying in low visibility conditions

For instance, in Fig. 14(a), (b), we used two frames of size 1920*1080 taken by our Dahua camera, which contain very far mini-drones. Indeed, the Upgraded-YOLO has detected the far mini-drones with a confidence score higher than 0.76, which is superior than that of the baseline (i.e., between 0.64 and 0.72). Accordingly, Fig. 14(a), (b) show that our model was efficient and outperformed the baseline (original YOLOv5) in the detection of mini-UAVs of tiny and small appearance.

Furthermore, the results in Fig. 14(c), (d) show that the bounding boxes of our model (red bounding boxes) are more adjusted with the detected mini-UAVs than the original YOLOv5. This was consistent with the previous evaluation, and this shows that our method has the lowest box loss. Finally, the last figures (lack of lighting for Fig. 14(e) and fog phenomena for Fig. 14(f)) show that our model performs well even under low-visibility conditions.

5 Conclusion

In this research, deep learning technology was applied to tiny/small flying object detection in air image. And based on the YOLOv5 object detector [21], a high-precision mini-UAV detection model was proposed. So, we firstly collected images of mini-UAVs in a real environment, using our Dahua Thermal Network PTZ Camera. Most of them consist of mini-UAVs flying in poor visibility conditions. Then, we constructed an own custom dataset designed by “Mini-UAVs air image dataset,” which provides a benchmark to evaluate the performance of the proposed detection model, especially under low-visibility condition. In addition, we proposed a new strategy of instance augmentation in order to increase the accuracy of our model. This strategy consists of adding tiny/small objects (mini-UAVs) to the images of our own custom dataset. As a result, in order to reduce the total loss, we implemented a mini-UAV detection model based on the state-of-the-art object detection method of YOLOv5, which has recently appeared. In the proposed detector, called Upgraded-YOLO, a new feature fusion layer was added to capture more feature information about tiny and small flying objects, detected in air images. This paper mainly researches and develops drone-related threats under the requirement of a real-time flying object detector. However, fast detection still needs specific hardware configuration. In the future, we will continue to optimize Upgraded-YOLO especially with a huge multi-class dataset of drones. At the same time, we will try to deploy and integrate our model with a flying object tracker such as DeepSORT in order to set up an anti-UAV system [49–51].

Acknowledgements This research is supported by the Tunisian Ministry of National Defense, Science and Technology for Defense Lab (STD), and Military Research Center through a research and development project.

Author Contribution T.D. (Tijeni Delleji) presented the ideas, carried out the experiments, and written the paper. F.S. (Feten Slimeni) contributed to programming, writing, and review. H.F. (Hedi Fekih) contributed to review and helped in obtaining the real-time images of flying mini-UAVs. A.J. (Achref Jarry) and W.B. (Wadi Boughanmi) contributed to the original draft preparation. A.K. (Abdelaziz Kallel)

contributed to review and editing the final version of the manuscript. Z.C. (Zied Chtourou) took the responsibility of supervision. All authors have read and agreed the published version of the manuscript.

Funding This research was funded by the Military Research Center, Tacib Mhiri, Aouina, 2045, Tunis, Tunisia.

Data Availability The data presented in this study are available on request from the corresponding author.

Declarations

Conflict of Interest The authors declare no competing interests.

References

1. USAF (2009) Unmanned Aircraft Systems Flight Plan 2009-2047, Technical report, unclassified, United States Air Force, Washington DC, pp. 24-27
2. Doyle DD (2013) Real-time, multiple, pan/tilt/zoom, computer vision tracking, and 3D position estimating system for small unmanned aircraft system metrology. DEPARTMENT OF THE AIR FORCE, AIR UNIVERSITY, Wright-Patterson Air Force Base, Ohio, USA. Jeffrey Maddalon
3. Maddalon J, Hayhurst KJ, Koppen DM, Upchurch JM (2013) Perspectives on unmanned aircraft classification for civil airworthiness standards. Langley Research Center, Hampton, Virginia. NASA/TM-2013-217969
4. Lykou G, Moustakas D, Gritzalis D (2020) Defending airports from UAS: a survey on cyber-attacks and counter-drone sensing technologies. *Sensors* 20(12):3537
5. Official DJI website: <https://www.dji.com/matrice600-pro/info> (last Accessed on 23 Mar 2021)
6. Seidaliyeva U, Akhmetov D, Ilipbayeva L, Matson ET (2020) Real-time and accurate drone detection in a video with a static background. *Sensors* 20(14):3856
7. Official DJI website: <https://www.dji.com/t16/info#downloads> (Last Accessed on 23 Jun 2021)
8. Ren S, He K, Girshick R, Sun J (2015) Faster R-CNN: towards real-time object detection with region proposal networks. arXiv preprint [arXiv:1506.01497](https://arxiv.org/abs/1506.01497)
9. He K, Gkioxari G, Dollár P, Girshick R (2018) Mask R-CNN. arXiv preprint [arXiv:1703.06870](https://arxiv.org/abs/1703.06870)
10. Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu C-Y, Alexander C (2015) Berg. SSD: single shot MultiBox detector. arXiv preprint [arXiv:1512.02325](https://arxiv.org/abs/1512.02325)
11. Redmon J, Divvala S, Girshick R, Farhadi A (2016) You only look once: unified, real-time object detection. Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 779-788
12. Redmon J, Farhadi A (2017) YOLO9000: better, faster, stronger. arXiv preprint [arXiv:1612.08242](https://arxiv.org/abs/1612.08242)
13. Everingham M, Ali Eslami SM, Van Gool L, Williams CKI, Winn J, Zisserman A (2015) The Pascal Visual Object Classes challenge: a retrospective. *Int J Comput Vis* 111:98-136
14. Everingham M, Van Gool L, Williams CKI, Winn J, Zisserman A (2010) The PASCAL Visual Object Classes (VOC) challenge. *Int J Comput Vis* 88:303-338
15. Redmon J, Farhadi A (2018) Yolov3: an incremental improvement. arXiv preprint [arXiv:1804.02767](https://arxiv.org/abs/1804.02767)
16. Lawal MO (2021) Tomato detection based on modified YOLOv3 framework. *Sci Rep* Jan 14;11(1):1447. <https://doi.org/10.1038/s41598-021-81216-5>. PMID: 33446897; PMCID: PMC7809275
17. Lin T-Y, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollár P, Lawrence Zitnick C (2014) Microsoft COCO: common objects in context. in 13th European Conference on Computer Vision, pp. 740-755
18. Kim D-H (2019) Evaluation of COCO Validation 2017 Dataset with YOLOv3. *Journal of Multidisciplinary Engineering Science and Technology* 6(7)
19. Bochkovskiy A, Wang CY, Liao HYM (2020) YOLOv4: optimal speed and accuracy of object detection, arXiv preprint [arXiv: 2004.10934](https://arxiv.org/abs/2004.10934)
20. Wang Z, Wu Y, Yang L, Thirunavukarasu A, Evison C, Zhao Y (2021) Fast personal protective equipment detection for real construction sites using deep learning approaches *Sensors* 21(10):3478

21. Ultralytics YOLOv5 and Vision AI, Madrid, Spain. Available online: <http://www.ultralytics.com> (Last Accessed on 03 Aug 2021)
22. Kharel S, Ahmed KR (2021) Potholes detection using deep learning and area estimation using image processing, Proceedings of SAI Intelligent Systems Conference, IntelliSys 2021: Intelligent Systems and Applications 296:373-388
23. Wang X, Wei J, Liu Y, Li J, Zhang Z, Chen J, Jiang B (2021) Research on morphological detection of FR I and FR II radio galaxies based on improved YOLOv5. *Universe* 7(7):211
24. Yan B, Fan P, Lei X, Liu Z, Yang F (2021) A real-time apple targets detection method for picking robot based on improved YOLOv5. *Remote Sensing* 13(9):1619
25. Yang G, Feng W, Jin J, Lei Q, Li X, Gui G, Wang W (2020) Face mask recognition system with YOLOV5 based on image recognition. Proceedings of 2020 IEEE 6th International Conference on Computer and Communications, IEEE Xplore , pp. 1398-1404
26. COCO dataset. Available online: <https://cocodataset.org/#home> (Last Accessed on 17 Sept 2021)
27. Adibhatla VA, Chih H-C, Hsu C-C, Cheng J, Abbod MF, Shieh J-S (2021) Applying deep learning to defect detection in printed circuit boards via a newest model of you-only-look-once. *Math Biosci Eng* 18(4):4411-4428
28. Agarwal S, Du Terrail JO, Jurie F (2018) Recent advances in object detection in the age of deep convolutional neural networks. arXiv preprint [arXiv: 1809.03193](https://arxiv.org/abs/1809.03193)
29. Yao J, Qi J, Zhang J, Shao H, Yang J, Li X (2021) A real-time detection algorithm for kiwifruit defects based on YOLOv5. *Electronics* 10(14):1711
30. Dahua Technology. Available online: <https://www.dahuasecurity.com/products/All-Products/Thermal-Cameras/Wizmind-Series/TPC-8-Series/TPC-PT8621C> (Last Accessed on 04 Aug 2021)
31. Zhang Y, Yongliang S, Jun Z (2019) An improved tiny-yolov3 pedestrian detection algorithm. *Digital Signal Processing* 183:17-23
32. NguyenN-D, Do T, Ngo TD, Le D-D (2020) An evaluation of deep learning methods for small object detection. *J Electr Comput Eng* 2020(3189691):18
33. Delleji T, Fekih H, Chtourou Z (2020) Deep learning-based approach for detection and classification of micro/mini drones. In 2020 4th International Conference on Advanced Systems and Emergent Technologies (IC_ASET), pp. 332-337
34. Kisantant M, Wojna Z, Murawski J, Naruniec J, Cho K (2020) Augmentation for small object detection. arXiv preprint [arXiv: 1902.07296](https://arxiv.org/abs/1902.07296)
35. Tong K, Wu Y, Zhou F (2020) Recent advances in small object detection based on deep learning: a review. *Image and Vision Computing, Science Direct, ELSEVIER* 97
36. Pang J, Li C, Shi J, Xu Z, Feng H (2019) R2-CNN: fast tiny object detection in large-scale remote sensing images. *IEEE Trans Geosci Remote Sens* 57(8)
37. Zhang Y, Bai Y, Ding M, Ghanem B (2020) Multi-task generative adversarial network for detecting small objects in the wild. *Int J Comput Vis* pp. 1810-1828
38. Chen C, Liu M-Y, Tuzel O, Xiao J (2016) R-CNN for small object detection. *Asian Conference on Computer Vision ACCV*, pp.214-230
39. Du Z, Yin J, Yang J (2019) Expanding receptive field YOLO for small object detection. *Journal of Physics. Conference Series, 3rd International Conference on Electrical, Mechanical and Computer Engineering, Guizhou, China* 1314
40. Lin T-Y, Goyal P, Girshick R, He K, Dollár P (2018) Focal loss for dense object detection. *IEEE Transactions on PatternAnalysis and Machine Intelligence* 42(2):318-327
41. Zheng Z, Wang P, Liu W, Ye R, Ren D (2020) Distance-IoU loss: faster and better learning for bounding box regression. *AAAI Conference on Artificial Intelligence* 34(07)
42. Rezatofighi H, Tsoi N, Gwak JY, Sadeghian A, Reid I, Savarese S (2019) Generalized intersection over union: a metric and a loss for bounding box regression. arXiv preprint [arXiv: 1902.09630](https://arxiv.org/abs/1902.09630)
43. Madasamy K, Shanmuganathan V, Kandasamy V, Lee MY, Thangadurai M (2021) OSDY: embedded system-based object surveillance detection system with small drone using deep YOLO. *EURASIP Journal on Image and Video Processing* 2021:19
44. Wang X, Song J (2021) CIIoU: improved loss based on complete intersection over union for bounding box regression. *IEEE Access* 9:105686-105695
45. Zhou J, Tian Y, Yuan C, Yin K, Yang G, Wen M (2019) Improved UAV opium poppy detection using an updated YOLOv3 model. *Sensors* 19(22):4851
46. Wicaksono AS, Supianto AA (2018) Hyper parameter optimization using genetic algorithm on machine learning methods for online news popularity prediction. *International Journal of Advanced Computer Science and Applications(IJACSA)* 9(12)

47. Chawla S (2016) Application of genetic algorithm and backpropagation neural network for effective personalise web search-based on clustered query sessions. *International Journal of Applied Evolutionary Computation (IJAE)* 7(1):33–49
48. Kingma DP, Ba JL (2014) A method for stochastic optimization. arXiv preprint [arXiv: 1412.6980](https://arxiv.org/abs/1412.6980)
49. Nicolai W, Bewley A, Paulus D (2017) Simple online and realtime tracking with a deep association metric. arXiv preprint [arXiv: 1703.07402](https://arxiv.org/abs/1703.07402)
50. Jiang N, Peng X, Yu X, Wang Q, Xing J, Li G, Zhao J, Guo G, Han Z (2021) Anti-UAV: a large multi-modal benchmark for UAV tracking. arXiv preprint [arXiv: 2101.08466](https://arxiv.org/abs/2101.08466)
51. Zhao J, Wang G, Li J, Jin L, Fan N, Wang M, Wang X, Yong T, Deng Y, Guo Y, Ge S, Guo G (2021) The 2nd Anti-UAV Workshop & Challenge: methods and results. arXiv preprint [arXiv: 2108.09909](https://arxiv.org/abs/2108.09909)
52. website Roboflow. PyTorch Object Detection, YOLOv5 is Here, <https://models.roboflow.com/object-detection/yolov5> (Last Accessed on 03 Dec 2021)
53. Yuxin F, Liao B, Wang X, Fang J, Qi J, Wu R, Niu J, Liu W (2021) You only look at one sequence: rethinking transformer in vision through object detection. arXiv preprint [arXiv: 2106.00666](https://arxiv.org/abs/2106.00666)
54. Adibhatla VA, Chih H-C, Hsu C-C, Cheng J, Abbod MF, Shieh J-S (2021) Applying deep learning to defect detection in printed circuit boards via a newest model of you-only-look-once. *Math Biosci Eng* 18
55. Sahin O, Ozer S (2021) YOLODrone: improved YOLO architecture for object detection in drone images. In 2021 44th International Conference on Telecommunications and Signal Processing (TSP), pp. 361–365. IEEE
56. ImageNet. ImageNet Large Scale Visual Recognition Challenge 2017 (ILSVRC2017). Available online: <https://image-net.org/challenges/LSVRC/2017/> (Last Accessed on 22 Jul 2022)

Springer Nature or its licensor holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.

Authors and Affiliations

Tijeni Delleji^{1,2} · **Feten Slimeni¹** · **Hedi Fekih^{1,2}** · **Achref Jarray¹** · **Wadi Boughanmi¹** · **Abdelaziz Kallel²** · **Zied Chtourou¹**

✉ Tijeni Delleji
tjeni_dalleji@yahoo.fr

Feten Slimeni
feten.slimeni@gmail.com

Hedi Fekih
fekihhedi2007@gmail.com

Achref Jarray
yassinejarray312@gmail.com

Wadi Boughanmi
Boughanmi.wadi@gmail.com

Abdelaziz Kallel
abdelaziz.kallel@crns.rnrt.tn

Zied Chtourou
ziedchtourou@gmail.com

¹ Science and Technology for Defense Lab (STD), Military Research Center, Taeib Mhiri, Aouina 2045, Tunis, Tunisia

² SM@RTS Laboratory, Digital Research Centre of Sfax, Tunis road km 10, El-Ons City 3021, Sfax, Tunisia