# Vehicle Detection From UAV Imagery With Deep Learning: A Review

Abdelmalek Bouguettaya, Hafed Zarzour, Ahmed Kechida, and Amine Mohammed Taberkit

***Abstract*— Vehicle detection from unmanned aerial vehicle (UAV) imagery is one of the most important tasks in a large number of computer vision-based applications. This crucial task needed to be done with high accuracy and speed. However, it is a very challenging task due to many characteristics related to the aerial images and the used hardware, such as different vehicle sizes, orientations, types, density, limited datasets, and inference speed. In recent years, many classical and deep-learning-based methods have been proposed in the literature to address these problems. Handed engineering- and shallow learning-based techniques suffer from poor accuracy and generalization to other complex cases. Deep-learning-based vehicle detection algorithms achieved better results due to their powerful learning ability. In this article, we provide a review on vehicle detection from UAV imagery using deep learning techniques. We start by presenting the different types of deep learning architectures, such as convolutional neural networks, recurrent neural networks, autoencoders, generative adversarial networks, and their contribution to improve the vehicle detection task. Then, we focus on investigating the different vehicle detection methods, datasets, and the encountered challenges all along with the suggested solutions. Finally, we summarize and compare the techniques used to improve vehicle detection from UAV-based images, which could be a useful aid to researchers and developers to select the most adequate method for their needs.**

***Index Terms*— Autoencoders, computer vision, convolutional neural networks (CNNs), deep learning, generative adversarial networks (GANs), recurrent neural networks (RNNs), unmanned aerial vehicle (UAV), vehicle detection.**

## I. Introduction

THE unmanned aerial vehicles (UAVs) are flying robots that fly without an onboard pilot, which are controlled by a person remotely or make them navigate autonomously using an onboard/on-ground computer system. Today, UAVs are almost everywhere in our life from photography to very complex applications due to their small size, low cost, and high mobility against other available technologies. Artificial intelligence techniques, especially deep-learning-based computer vision, make UAVs smarter and can achieve tasks that are impossible before their arrival. Computer vision is one of the most important topics in the field of autonomous robots, including UAVs. It is the science that gives UAVs the ability to analyze, process, and understand the content of digital images/videos to make the right decision at the right time. Detecting vehicles in UAV images and videos is one of the tasks which attracted great attention from researchers. It plays an important role in a wide range of fields, such as transportation, military, search and rescue, and surveillance. Traditional computer vision techniques are basically based on handed-engineering-based features extractor or shallow-learning algorithms followed by classifiers. Recently, these techniques were surpassed by deep learning architectures. Many factors lead to such fast evolution, including deep neural networks, a large amount of data, and GPUs for processing power. Among the most known deep neural network architectures, we find the convolutional neural network (CNN), the generative adversarial network (GAN), autoencoders, and the recurrent neural network (RNN), which are used in many computer visions tasks, including object detection. These architectures have been contributed to improve computer vision abilities and performance. Today, most of the state-of-the-art object detection algorithms are based on deep learning techniques as their backbone. Vehicle detection from UAV imagery represents a big challenge due to many properties concerning aerial images, such as small size, occlusion, orientation, and shadow. Over the last years, many challenging vehicle benchmarks datasets have played a crucial role to develop and evaluate vehicle detection algorithms, such as UAVDT [1], UAV123 [2], Stanford Drone Dataset (SDD) [3], VisDrone [4], and DLR 3K Munich Vehicle Aerial Image Dataset [5]. In this article, we summarize the latest development in deep-learning-based vehicle detection frameworks in which we examine more than 200 publications about the vehicle and other object detection algorithms, deep learning architectures, and the different benchmark datasets.

There are some survey papers in the literature, which have discussed different object detections using deep learning techniques. Most of them have investigated object detection in natural scenes. However, none of these works have focused on deep-learning-based vehicle detection from UAV imagery. Carrio *et al.* [43] presented different deep learning techniques and their applications for UAVs. However, their review is

Manuscript received 24 June 2020; revised 4 October 2020 and 18 January 2021; accepted 11 May 2021. Date of publication 24 May 2021; date of current version 28 October 2022. *(Corresponding author: Hafed Zarzour.)*

Abdelmalek Bouguettaya, Ahmed Kechida, and Amine Mohammed Taberkit are with the Signal Processing and Imagery Division, Analysis and Control of Embedded Drone Systems Research Team, Research Center in Industrial Technologies (CRTI), Algiers 16014, Algeria (e-mail: a.bouguettaya@crti.dz; a.kechida@crti.dz; a.taberkit@crti.dz).

Hafed Zarzour is with the LIM Research, Department of Mathematics and Computer Science, University of Souk Ahras, Souk-Ahras 41000, Algeria (e-mail: hafed.zarzour@gmail.com).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2021.3080276.

Digital Object Identifier 10.1109/TNNLS.2021.3080276

relatively old and did not present the detection task in detail. Another review is presented in [52]. The authors targeted only one dataset which is VisDrone. It is a very large-scale dataset that consists of different objects including vehicles. However, the authors presented the performance of different algorithms only on the VisDrone dataset and reported the results from the summary papers of the ECCV and ICCV challenge competitions. They presented six datasets for object detection from UAV-based images/videos, where only three of them targeted vehicle detection. Only CNN-based algorithms have been discussed in their paper. Ayalew [79] has focused on some R-CNN and YOLO variants with so few details. Moreover, Ayalew [79] did not provide explanations about lightweight detectors nor benchmark datasets.

To the best of our knowledge, this is the first review article that focuses on the vehicle detection task with most of the necessary materials with the main objective to provide an overview on the different deep learning techniques to improve vehicle detection from aerial images, especially from UAVs systems, while determining where each technique can be used. Many techniques are covered including multiscale bounding boxes, hard negative mining, and fusing feature maps from different backbone layers. Also, we take into account the lightweight detectors that are very convenient for small devices with limited computation power. Moreover, the most popular benchmark datasets for vehicle detection tasks are presented all along with different evaluation metrics. This study could not only help researchers and developers to understand the concepts of vehicle detection from UAV imagery but also assist them to choose the appropriate architecture and dataset for their applications. The main contributions of this article are given as follows.

1) We introduce the most known and powerful deep learning architectures that contribute to improve vehicle detection performance in UAV-based images and videos, where different techniques are introduced to enhance small size, dense, and oriented vehicle detection.
2) We describe the most used aerial image/video benchmark datasets and their properties while presenting for what application each dataset is better for.
3) We present the best available vehicle detection frameworks in the last years all along with their accuracy and speed evaluation. Also, lightweight detectors for small devices are presented.
4) We discuss various UAV-based vehicle detection challenges all along with the impact of different parameters on the detection accuracy and speed. The parameters include the image resolution, altitude, and view angle.

The remainder of this article is organized as follows. In Section II, different deep learning architectures are presented. Section III is devoted to the vehicle detection algorithms and the encountered challenges. Different techniques used to improve vehicle detection in UAV-based images/videos are presented in Section IV. Section V is dedicated to present some of the important benchmarks and evaluation metrics. Some discussions are presented in Section VI. We conclude this article in Section VII.

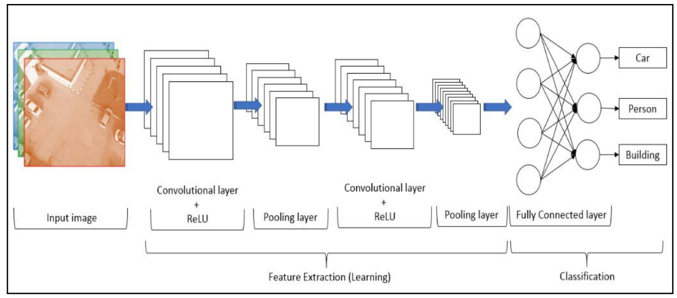


Fig. 1. Typical CNN architecture.

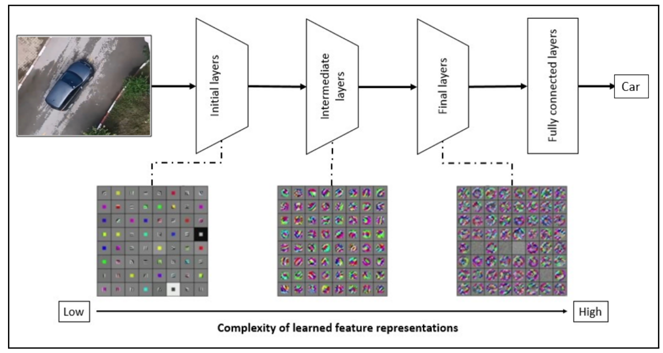


Fig. 2. Feature visualization in different layers [18].

## II. Overview of Deep Learning Techniques

In this section, we present an overview of the most relevant deep learning architectures that are developed to solve different problems in a large number of domains, such as computer vision, speech recognition, system recommender, and natural language processing (NLP). Among these architectures, we find CNNs, autoencoders, GANs, RNNs, and long short-term memory (LSTM). All of these techniques are employed not only for object detection but also to enhance the detection performance.

### A. Convolutional Neural Networks

CNN is one of the most popular and successful feedforward neural network algorithms for deep learning, which achieves state-of-the-art results in many problem areas, such as NLP, speech recognition, and recommender systems. However, it is mostly used to handle computer vision and image analysis problems, including image classification [7]–[10], object detection [11]–[14], and image segmentation [15]–[17]. Fig. 1 shows the typical CNN architecture.

As shown in Fig. 2, CNNs learn low-level vehicle features in the initial layers (edges and corners), followed by more complex intermediate and high-level feature representations in deeper layers (wheels, windscreen, car doors, and cars), which are used for a classification task. In the last years, CNN architectures achieve incredible performance on some computer vision tasks, including vehicle detection and classification [82], [83]. This is happening thanks to the combination of recent powerful computing systems (GPUs), software

TABLE I

PERFORMANCE COMPARISON OF DIFFERENT CNN ARCHITECTURES (CITATION TILL APRIL 20, 2020)

| Ref | CNN architectures | Year | Number of parameters (million) | Number of MACs (million) | Number of Layers | Top-5 error rate (%) | Top 1 accuracy (%) | Top 5 accuracy (%) | Number of Citations |
|---|---|---|---|---|---|---|---|---|---|
| [6] | LeNet | 1998 | 0.060 | 0.341 | 5 | - | - | - | 25872 |
| [19] | AlexNet | 2012 | 60 | 724 | 8 | 16.4 | 57.2 | 80.3 | 60329 |
| [18] | ZFNet | 2013 | 60 | - | 8 | 11.7 | 64 | 85.3 | 8896 |
| [21] | VGGnet | 2014 | 138 | 15.5 G | 16 | 7.3 | 71.4 | 91.9 | 36459 |
| [20] | GoogLeNet | 2014 | 5 | 1.43 G | 22 | 6.7 | 69.8 | 89.9 | 20752 |
| [171] | Inception v3 | 2015 | 23.8 | - | 159 | 3.5 | 78.8 | 94.4 | 7465 |
| [22] | ResNet-152 | 2015 | 60 | 11.3 G | 152 | 3.57 | 80.62 | 95.51 | 43982 |
| [172] | Inception-ResNet v2 | 2016 | 55.8 | - | - | 3.08 | 80.1 | 95.1 | 4294 |
| [174] | ResNeXt | 2016 | 83.6 | 8 G | 101 | 3.03 | 80.9 | 95.6 | 2026 |
| [173] | DenseNet | 2016 | 20 | 1.15 G | 201 | 5.54 | 78.54 | 94.46 | 8351 |
| [26] | SqueezeNet | 2016 | 1.25 | 837 | - | - | 57.5 | 80.3 | 2276 |
| [175] | SENet | 2017 | 145.8 | 42.3 G | 154 | 4.47 | - | - | 2724 |
| [23] | MobileNet v1 | 2017 | 4.24 | 569 | 28 | 10.5 | 70.6 | 89.5 | 3928 |
| [27] | ShuffleNet v1 | 2017 | 5.2 | 524 | 50 | - | 70.9 | 89.8 | 1173 |
| [24] | MobileNet v2 | 2018 | 3.47 | 300 | 50 | - | 72.0 | - | 1549 |
| [28] | ShuffleNet v2 | 2018 | - | 591 | - | - | 75.4 | - | 446 |
| [25] | MobileNet v3 | 2019 | 5.4 | - | - | - | 75.2 | - | 125 |
| [29] | PeleeNet | 2019 | 2.8 | 508 | - | - | 72.1 | 90.6 | 75 |

(Tensorflow, Keras, and Pytorch) and also the large labeled dataset (ImageNet, MS COCO, and VisDrone) used to train CNN models.

Deeper, wider, and more efficient CNN architectures are proposed, over the years, after the success of the famous LeNet-5 [6]. AlexNet (SuperVision) [19], ZFNet (Clarifia) [18], GoogLeNet [20], VGGnet [21], and ResNet [22] are some examples of deep CNN architectures. Many other deep CNN architectures have been proposed as shown in Table I. However, they involve a large number of learnable parameters corresponding to the weights and biases learned during the training process (see Table I), which make them computationally very exhausting, especially in the field of embedded systems where resources are very limited. For that reason, lighter versions have been proposed to achieve real-time processing even on mobile platforms while preserving the same accuracy as deep CNN or even better, in some cases, such as MobileNet [23]–[25], SqueezeNet [26], ShuffleNet [27], [28], and PeleeNet [29].

### *B. Recurrent Neural Networks and Long Short-Term Memory*

In feedforward networks, the activations flow only in one single direction, from the input to the output layer. In such architecture, an output at a certain time "$t$" is totally independent of the output at a time "$t-1$." However, RNN is a special type of deep neural network architecture that looks like a feedforward network but with a feedback connection to save the output of a layer and feed it back to its input for the next prediction (see Fig. 3). Thus, RNN uses inputs from previous stages to help a model to remember its past, which makes it very useful to handle sequential data such as video, audio, and text. RNN architecture and its variants have the ability to save previous states and predict the next position of the vehicle. Combining their characteristics with CNN-based detectors could improve the detection accuracy, and they are mostly used for object tracking.

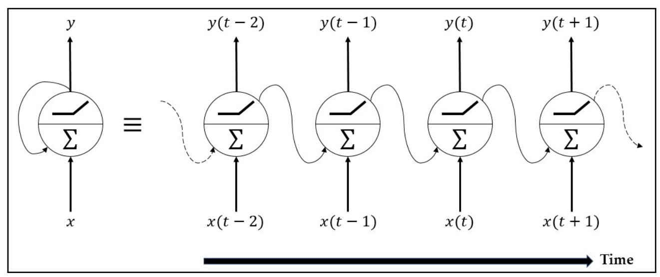


Fig. 3. Basic RNN structure (right) unrolled through time (left).

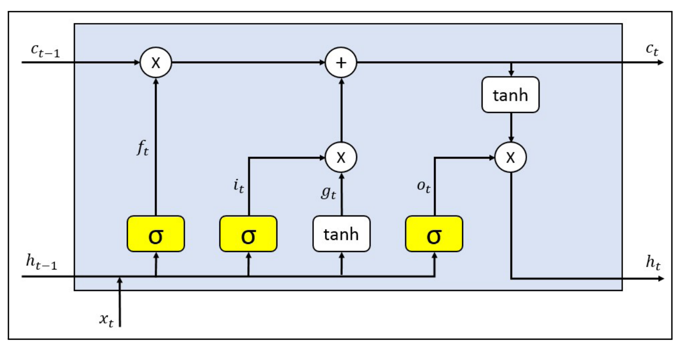


Fig. 4. Typical LSTM architecture.

In the last years, RNNs have been utilized in a wide range of applications, such as speech recognition [30], [31], image captioning [32], [33], time series prediction [34]–[36], NLP [37]–[39], and machine translation [40], and even to enhance object detection and trajectory prediction performances in videos [41], [42], which is very useful for autonomous UAV.

Hochreiter and Schmidhuber [44] proposed LSTM to solve some problems encountered in classical RNN architecture, such as vanishing gradient. LSTM network works much better than the standard RNN version due to its characteristics of remembering patterns for a long-time duration. Fig. 4 shows the main architecture of LSTM.

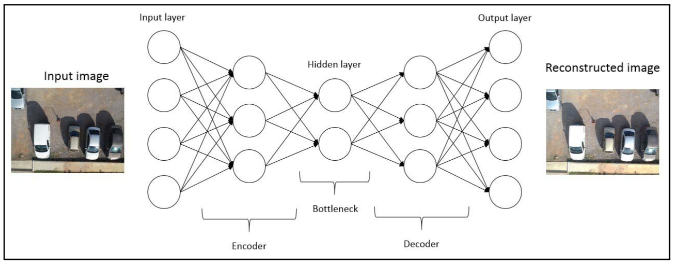


Fig. 5. Typical autoencoder architecture.

### C. Autoencoders

Data compression is a big topic in the computer vision field. The main role of data compression is to convert our input data into a smaller representation, which is used to reconstruct the original input data. Autoencoder [46] is another type of artificial neural network that is based on unsupervised algorithms to do this compression and reconstruct the input data after applying a series of operations. Basically, it is designed to reproduce its input at the output. Autoencoders are mainly composed of three components: an encoder, a bottleneck, and a decoder (see Fig. 5). The output of an autoencoder represents the reconstructed input with the minimum possible errors. The encoder part can also serve as an interesting feature extractor, as shown in [47], where stacked denoising autoencoders (SDAEs) are used. There are various autoencoders types, such as convolutional autoencoder (CAE) [48], sparse autoencoder [49], deep autoencoder [50], and variational autoencoder [51]. The upsampling operation in the decoder part of an autoencoder can provide better accuracy in detecting small objects, as shown in [127], in which their proposed detector is based on an autoencoder-like architecture.

Autoencoders are only able to compress data similar to what they have been trained on. For example, an autoencoder that has been trained on human faces will not perform well on images of vehicles. Moreover, they are worse than classical compression techniques, such as JPEG and MPEG. However, they work really well on other tasks, such as image denoising [53]–[55], dimensionality reduction [56]–[58], reconstructing missing parts of an image [59], [60], and also colorizing images [61].

### D. Generative Adversarial Networks

Goodfellow *et al.* [62] proposed a new unsupervised deep neural network, called GANs, in 2014. It is described as the most interesting idea in the last decade by Emmert-Streib *et al.* [63]. GANs reached great success to generate new data instances that are similar to the ones in the given dataset. GANs consist of two networks, generator and discriminator networks, competing against each other (see Fig. 6). The generator network tries to generate new realistic samples from random noise (Gaussian distribution). The discriminator network is a regular neural network-based classifier, which takes both the real data and the generated one as inputs then decide whether the generated samples are considered real or fake samples. The output of a discriminator is the probability that

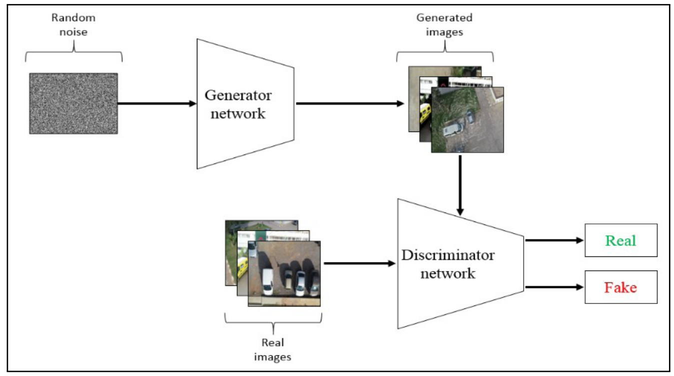


Fig. 6. GAN architecture.

the input images belong to the real dataset or not, where it affects a probability near to “1” for real images and a probability near to “0” for fake images. Thus, the generator must produce samples that are similar as possible to real ones to fool the discriminator.

GAN architecture can enhance the poor representations of tiny objects to super-resolved ones in the generator part to fool the discriminator part, which could improve small vehicle detection performance from high-altitude UAV imagery [45]. However, GAN architecture is usually used to generate new data samples making it one of the best solutions for data augmentation.

## III. UAV-Based Vehicle Detection Algorithms and Challenges

The continuously increasing number of vehicles has forced transport management agencies and researchers to develop new accurate methods for traffic monitoring, parking securing, accident detection in difficult areas, and parking lot occupation. One available solution is to use some on ground fixed sensors, such as radars and fixed cameras [64]–[66], which provide only a partial overview and miss a lot of information. However, aerial image sensors seem to be a better solution because they provide a larger overall overview of areas of interest.

UAVs have emerged as a new image acquisition platform, which presents several advantages over satellites and airborne for solving vehicle detection problems. This is due to their low cost, high flying flexibility, small size, and easier and faster data acquisition. Moreover, the most important of all, UAVs provide extremely high-resolution images encompassing abundant spatial and contextual information, which makes them a center of many new UAV image analysis applications. As UAV applications become widespread, implementing sensors, powerful processors, artificial intelligence, and computer vision techniques is becoming indispensable. The combination of deep learning and computer vision techniques allows UAVs to analyze, process, and understand the content of these images and videos, which enables them to navigate autonomously, detect and track objects, and even provide analytical feedback in real-time. These technologies can provide a higher level of autonomy and make the UAVs smarter and more efficient in many tasks, including navigation in GPS-denied areas.

As a hotspot of deep learning and computer vision studies, vehicle detection from UAV-based images has attracted increasing attention in many studies in the last years. It plays a vital role in a large number of applications, such as road traffic information [67]–[70], tracking specific types of vehicles [71], [72], transportation monitoring [73], [74], intelligent parking [75], speed control [76], [77], and social distancing detection in the context of COVID-19 pandemic.

### *A. Deep-Learning-Based Detectors*

The development of different deep learning techniques, the availability of very large datasets, and the continuous improvement of processing power have been led to impressive progress in the object detection field, especially for vehicle detection from UAV-based aerial images.

Recently, there has been significant progress in vehicle detection due to deep learning techniques, particularly the emergence of CNN, which is an effective method that can improve detection performance. Most of the deep-learning-based vehicle detection research studies are based on CNN architectures as a backbone. Unlike the traditional feature extractor techniques, CNN has a strong ability to extract relevant features automatically. Moreover, many CNN-based algorithms for vehicle detection in UAV imagery have been proposed as an end-to-end framework. These algorithms can be categorized into two main groups: two-stage object detectors and one-stage object detectors. Two-stage object detectors, also called region-based object detectors, are basically R-CNN [89] and its variants, including SPPnet [90], Fast R-CNN [91], Faster R-CNN [92], R-FCN [93], and Mask R-CNN [94]. They consist of two main stages: the first stage is responsible to generate candidate regions, while the second stage predicts the class that each region belongs to. Over the years, many region proposal techniques were proposed [95], [96], and it seems that the selective search algorithm [97] provides better results than the other nondeep learning techniques. For that reason, the selective search algorithm was adopted in R-CNN and Fast R-CNN to extract particular region proposals, which may contain an object, with different scales and positions. However, the selective search algorithm is not a learnable algorithm, which can lead to bad region selection that makes it very slow and less accurate. Although R-CNN and Fast R-CNN achieved impressive results over classical methods, they are still very slow due to the limits of the selective search algorithm. These two architectures are not often used anymore because of their lack of speed and accuracy.

For the above reasons, most of the studies on vehicle detection in aerial images, using a region-based detector, have relied on another improved version of R-CNN called Faster R-CNN. Instead of using the selective search algorithm for region proposal generation, the Faster R-CNN network applied a separate deep CNN network to generate a predefined number of high-quality region proposals using the region proposal network (RPN). These proposals are then used by Fast R-CNN for the detection, which achieves remarkable detection performance. In order to predict regions of different aspect ratios and scales within an image, Faster R-CNN introduces the concept of anchors boxes to detect overlapping objects. These anchors can be denoted by (cx, cy, w, and h), where: “cx” and “cy” are the coordinates of the center point of the default box, and “w” and “h” are the width and height of the box.

Even with the impressive accuracy achieved by the region proposal framework, their architecture is complex and still noticeably very slow due to the candidate region generation module. Thus, it makes them not suitable for real-time object detection, particularly on mobile devices. For UAVs, it is indispensable to detect all objects in its visual field, such as vehicles, with high speed. Thus, the UAV can accomplish the required task in real time. One-stage detectors were proposed to improve vehicle detection speed. They are different from two-stage detectors. Instead of dividing the detection framework into two parts, single-shot detectors fuse the recognition and detection steps into one deep neural network, which avoids spending too much time on candidate region proposal generation. Many one-stage detectors have been proposed over the last years to handle real-time object detection that could be implemented even on small devices. One-stage object detection algorithms mainly include You Only Look Once (YOLO), Single-Shot Multibox Detector (SSD), and RetinaNet. In 2016, Redmon *et al.* [98] developed the first version of YOLO for real-time object detection. It consists of dividing the input image into a fixed number of grid cells. Each cell is considered as a proposal that may contain an object. However, it has two main drawbacks over the region-based methods: high localization error and low recall rate. Liu *et al.* [99] proposed another one-stage detector called SSD to overcome some limitations of YOLOv1 achieving better accuracy for real-time processing. It has the same principle as YOLO dividing the input image into grid cells. Moreover, it generates multiple anchor boxes of different scales and aspect ratios. Afterward, Redmon and Farhadi [100], [101] proposed YOLOv2 and v3 in 2017 and 2018, respectively. These methods improve the detection accuracy further while maintaining detection speed. They adopted a more powerful CNN framework, which is Darknet. In addition, they also adopted a better anchor strategy, using k-means clustering from the training data instead of set them manually, to enhance detection accuracy. Lin *et al.* [102] developed RetinaNet that adopts the FPN concept and focal loss function to improve the detection accuracy of small-sized and dense objects. In 2020, Bochkovskiy *et al.* [103] proposed YOLOv4 enriched with many new features to provide the best tradeoff between speed and accuracy. These frameworks provide a remarkable tradeoff between detection accuracy and speed. However, all of them are developed for natural benchmark datasets that contain only one or few objects.

More deep-learning-based object detection algorithms are presented in [52]. The authors mentioned different algorithms for object detection from VisDrone2018 and VisDrone2019 datasets, where they received 34 and 47 object detection algorithms in VisDrone2018 and VisDrone2019 challenges, respectively. Most of the proposed algorithms are based on the aforementioned methods, such as YOLOv3, FPN, Faster R-CNN, and RetinaNet. Also, other approaches are proposed, such as HRDet+ [84], TridentNet [85], DBCL [86], DetNet [87], and CornerNet [88]. However, HAL-Retina-Net and DPNet [52] are the best methods on the VisDrone-DET 2018 dataset achieving more than 30% AP.

Many other deep learning architectures, such as GAN, autoencoder, and LSTM, are combined with CNN-based detectors to improve the detection performance. In order to handle the UAV-based dataset lack of our targeted object (vehicle), GANs are mostly used to increase the sizes of the datasets. Also, autoencoders can be used to generate new unseen images to increase dataset size. Moreover, researchers can use autoencoder-like architectures to improve small vehicle detection, as presented in [127]. RNN and LSTM are mostly used to improve the detection performance in video sequences. Kompella and Kulkarni [144] proposed a 1-D RNN architecture called RVS to improve salient object detection in videos. Most of the time RNN/LSTM are combined with different CNN architectures to achieve better performance on vehicle detection in video sequences, as shown in [42] and [150]. More details are presented in Section IV.

### *B. Vehicle Detection From UAV Imagery Challenges*

UAVs can fly at different altitudes with different view angles resulting in many challenges compared with vehicle detection in ordinary view images and videos. Automated and accurate vehicle detection in UAV-based images remains a challenging problem due to many issues, such as, but not limited to, scale variations, different vehicles orientations and types, illumination variations, high density, appearance similarity of the vehicles with other object types, partial occlusions of vehicles, complex background, and the image qualities. Moreover, limited annotated training datasets and real-time detection are other problems that must be handled.

*1) Small Object:* High-resolution aerial images contain a smaller and large number of objects beside vehicles than those in ordinary scene images, which causes the lack of information in feature extraction. This consequently increases the difficulty of localization and the missed detections.

*2) Scale Diversity:* The same object can be of different sizes, which can be ambiguous and confusing for a UAV when it is captured at different altitudes. It makes the detection more challenging due to the different scales of the vehicles throughout the captured images/videos.

*3) Vehicles Orientations and Types:* The mobility degree of UAVs leads to many view angles of the vehicles: top view, side view, and front view. The captured images/videos consist of vehicles of various types in different orientations and shapes, which make the detection more and more challenging.

*4) Illumination Condition Variations:* Light is an important factor in any classification/detection task, and it is one of the most difficult problems to solve. The reliable and accurate system cannot be achieved without taking this factor into account, hence the need for some image preprocessing to minimize the effects of lighting and illumination.

*5) Background Complexity and Appearance Similarity:* The large size of aerial images and the wide view angle produce a very complex and clutter background with several objects in the same image, thus leading to confuse the vehicle's appearance with other similar object types, such as trash bins and air conditioner units. This increases the difficulty of vehicle detection and classification.

*6) Limited Datasets:* The limited annotated datasets are another challenge facing vehicle detection from aerial images. The available annotated vehicles datasets in UAV imagery are very few, which makes the detectors less accurate, especially for deep-learning-based ones where we need a very large number of annotated images/videos to train them. Most of the available datasets are based on satellite images. For this reason, we find most of the studies are based on their own collected UAV-based datasets.

*7) Real-Time Issue:* In order to manage road traffic or parking lot, UAVs must detect and classify vehicles in a clutter environment in real time. However, the UAV platform has limited onboard hardware resources to execute such complex tasks and heavy data, which makes detection and classification operations in real time the biggest challenges. Also, aerial images are larger than ordinary scene images, which decreases the detection speed. Therefore, accurate and fast vehicle detection algorithms are indispensable for low-power and low processing UAV platforms to accomplish their mission at the right time.

In order to overcome these challenges, there is a critical need to develop robust algorithms that are able to enhance the detection performance from aerial images/videos.

## IV. Techniques to Improve Vehicle Detection From UAV-Based Images and Videos

Deep-learning-based vehicle detection is not transferable directly on high-altitude aerial imagery, which contains a large number of smaller and randomly located objects. To handle these challenges, many approaches, based on improved versions of region-based and one-stage detector algorithms, are developed. In the last years, many techniques have been proposed to improve vehicle detection from aerial views. Among the adopted techniques to extend CNN-based detector to UAV imagery, we find redesigning the anchor boxes properties and the feature extractor structure, adopting data augmentation techniques, and combine different deep learning approaches. In the following, we describe the most effective techniques used to improve the performance of vehicle detection in UAV-based images.

### *A. Training Strategy and Parameter Adaptation*

To detect vehicles from UAV-based images, researchers [78], [81], [96], [104]–[108] adopted the original detectors frameworks. In order to show how these architectures are accurate, they just trained them on low altitude and/or top view UAV imagery while adapting some parameters to handle multiscales vehicle detection, including the number of layers, anchor boxes scales, loss function, and input image size. Training the original detectors on low-altitude images achieved remarkable results. However, a big drawback of such architecture is their lack of small vehicle detection from very high altitudes.

Many deep-learning-based detectors studies show the impact of the framework settings' choice on vehicle detection from UAV images. Anchor box is one of the most important parameters, which is set in different scales and aspect ratios in a way to fit various sizes and shapes of the vehicles. Larger anchors are applied to detect large vehicles while smaller ones to detect tiny vehicles. Thus, changing the anchors' sizes to fit the used dataset is a good idea to improve multisize vehicle detection in UAV images. Combining the anchor box concept with the right framework parameters and a suitable dataset can achieve remarkable results.

Hsieh *et al.* [109] showed the effect of anchor box sizes and some other parameters on the performance of vehicle detection in UAV images. Tang *et al.* [110] and Shen *et al.* [111] showed the impact of anchors scales and aspect ratios on small vehicle detection accuracy. Xu *et al.* [81] demonstrated that Faster R-CNN provides good results on vehicle detection from low altitude UAV-based images. However, the authors did not extend the detection to higher altitude and multi-types of vehicles, where the feature extraction and detection become more difficult. In the same way, Ammar *et al.* [104] showed the impact of the chosen dataset on Faster R-CNN and YOLOv3 performance while using different hyperparameters, including input size, feature extractor, and score threshold. Sommer *et al.* [96] investigated the impact of the adapted RPN and Fast R-CNN on small object detection. They showed the effect of the chosen CNN architecture and the network parameters (number of layers, number and size of filters, and dropout) on small object detection accuracy. Radovic *et al.* [105] showed that a trained YOLO, with a suitable dataset and parameter settings, is able to detect vehicles and aircraft from low altitude and top view UAV images. They trained YOLO on a dataset that consists of satellite and UAV images. Tang *et al.* [107] trained the original SSD, YOLOv1, and YOLOv2 structures on UAV imagery to detect vehicles. YOLOv2 had the best results in detection accuracy and speed achieving 77.12% and 0.048 s, respectively, against (67.99% and 0.056 s) on YOLOv1 and (72.95% and 0.055 s) on the SSD detector.

The ordinary bounding box contains some background features in the case of vehicles in multiple orientations. Moreover, it has difficulties separating vehicle targets effectively. In order to overcome these issues, oriented bounding boxes can improve vehicle detection and counting in dense and sparse UAV scenes. For this reason, Tang *et al.* [112], Guo *et al.* [113], and Li *et al.* [114] adopted rotated bounding boxes. The loss function is another important parameter to improve vehicle detection in UAV images. Many studies [115]–[118] have shown the impact of the chosen loss function on the detector performance. Some researchers used the focal loss function instead of the cross-entropy function, in the training process, to handle small-size vehicle detection because it concentrates on hard features, which makes it a good solution that can improve the performance of small vehicle detection [115], [118]. A more advanced training strategy is adopted in [119] and [120], which trains the model alternately with cross-entropy loss (focus much on easy examples during training) and focal loss (focus on hard examples during training) for more discriminative features extraction; thus, more improvements are provided.

Most of these techniques have remarkable results on top view and low altitude UAV images. However, they have poor performance when the UAV provides us images/videos from different altitudes, where the vehicle size varies from a few pixels (e.g., $10 \times 20$) to medium and larger sizes. This is because the UAV-based image has different characteristics than natural on-ground, top views, and low altitude scenes.

### *B. Using Feature Maps From Different Layers*

Most of the previous CNN-based object detection techniques use a single feature map, which is the last convolutional layer output, to detect vehicles in an image. Deep CNN that has a large number of convolutional and pooling layers results in feature maps, from deeper layers, with a receptive field larger than that of shallower ones. Consequently, deep layers produce low spatial resolution with strong semantics, which makes them more suitable for large vehicle detection, while shallower layers are semantically very strong, which provides high feature map resolution, which is more suitable for small-size vehicle detection. Therefore, deep layers in CNN architectures provide a high recall rate with poor localization accuracy while shallower layers otherwise [121].

Many researchers have proposed to use shallower CNN architectures or feature maps from earlier layers to enhance small-size vehicle detection. Wang *et al.* [122] showed the impact of the chosen backbone architecture on Faster R-CNN accuracy. Sommer *et al.* [123] replaced VGG-16, which is the main CNN backbone of Faster R-CNN, with another CNN architecture with fewer layers that are more suitable to handle small-size vehicle detection. Sommer *et al.* [96] proposed optimized versions of Fast and Faster R-CNN detectors. High-resolution feature maps from earlier layers of the original VGG architecture were used resulting in significant improvement of Faster R-CNN performance on small vehicle detection in UAV images. Also, they replaced the original VGG network with another network inspired by the network in [124] to show the effect of shallow networks on the detector performance. However, only one vehicle category was considered in their work. The same authors extended the adapted Faster R-CNN for multicategory vehicle detection [123], but, even in this work, the authors are limited to only two vehicle categories detection due to the limitation of annotated object classes. Moreover, they showed the impact of data augmentation to improve detection accuracy. However, even with these improvements for small-size vehicle detection in UAV images, using feature maps provided by shallower layers and shallow CNN architectures result in a high number of false detection due to the apparent similarity of some objects with vehicles.

Most of the recent studies on object detection using only one scale feature, almost the last layer feature map, to enhance the tradeoff between detection speed and accuracy.

However, the techniques used in these studies have a lack of detecting tiny objects. Quite recently, considerable attention has been paid to fuse feature maps from different layers.

Many techniques have been gaining importance in recent years proposing to combine heretical feature maps from shallow and deeper layers. Many researchers demonstrate that combining feature maps from different layers provides more suitable results for multiscales vehicle detection, especially for small-size vehicle detection. Sommer *et al.* [125] extended Faster R-CNN to detect small vehicles, where they applied the deconvolutional layer on a deeper layer to upsample low-dimensional feature maps. Then, the upsampled feature maps are combined with feature maps from shallower layers to detect small-size vehicles from aerial images. Acatay *et al.* [126] presented an extended YOLOv2, called DYOLO, which achieved remarkable accuracy combining the upsampled feature maps of deep layers (obtained by deconvolutional module) with those of shallow layer. Tayara *et al.* [127] proposed the fully convolutional regression network (FCRN), which is a detector based on autoencoder-like architecture to detect small vehicles in UAV images. Tang *et al.* [110] proposed a modified Faster R-CNN version to overcome the limits of RPN network performance on small object localization and the differentiation between vehicles and complex background features in the classifier. They replaced the original RPN with a Hyper-RPN (HRPN), which is based on ZFNet architecture. It combines feature maps from the last three convolutional layers while using a cascade of boosted classifiers to improve the classification accuracy reducing false detection by mining hard negative examples. Also, Tang *et al.* [128] and Deng *et al.* [129] improved Faster R-CNN doing the same thing as [110] combining the feature maps from the last three convolutional layers of ZFNet architecture to generate a hyper feature map while adopting different anchor scales and aspect ratios. Xie *et al.* [120] proposed a detector based on RefineDet [130] architecture adapted to detect small vehicles in aerial images combining weak and strong semantic features.

The feature pyramid network (FPN) is one of the most recommended techniques to enhance multiscale vehicle detection, which was first introduced by Lin *et al.* [131]. It aims to combine shallow layers low-level features and deep layers high-level features to produce a set of new feature maps with different spatial resolutions that give us the possibility to detect multiscale vehicles, especially in UAV scene cases. Several publications have appeared in recent years documenting the improvements brought by adopting FPN architecture [115], [122], [132], [133]. Li *et al.* [114] showed the impact of FPN comparing different backbone architectures, where they achieved better results adopting ResNet-101 with FPN. Wang *et al.* [122] studied vehicle detection from UAV images and showed that FPN improves its detection in very challenging scenes, including vehicles in shadow and/or occluded by trees and buildings.

### C. Training Data and Data Augmentation

The training process does not work the same way that it does with humans, where we find that a little boy of three- or four-year-old learns from very little data. Deep-learning-based methods always require large amounts of training data to perform very well. Thus, the performance of the model depends on the datasets' size, type, and quality, as shown in [78], [106], [108], and [134], where the authors showed the impact of the chosen dataset on the performance of the algorithm. Generally, more data improve the model accuracy better than complex algorithms.

Most of the well-annotated datasets are those of natural scene images, such as ImageNet [135], PASCAL VOC [136], and MS COCO [137]. Unfortunately, the available datasets that aim to detect vehicles from aerial images are very limited, which makes it one of the major problems for vehicle detection from UAV imagery. Moreover, most of the available datasets are based on satellites, airborne and fixed cameras. In order to provide the necessary datasets to train a model on UAV imagery, we are forced to use non-UAV-based images all along with other methods. One of the most used methods is to collect our datasets by capturing thousands of images through UAV platforms [75], [78], [80], [81], [138], [139], and the annotation will be done manually. However, this method is very exhausting and time-consuming. Another solution is to use an available dataset [96], [106], [140], but most of these datasets are specialized for only one task. For these reasons, we find ourselves forced to combine collected and available datasets, as in [107] and [129], or to use some data augmentation techniques to increase the number of training images to make the detector more efficient.

There are two approaches for data augmentation: classical techniques and deep-learning-based techniques. Using these data augmentation techniques in the right way, we can improve the performance of vehicle detection algorithms. For the classical data augmentation methods, we just apply some basic image manipulations, such as flipping, rotations, cropping, scaling, color space transformation, add some noise, and more [117], [123], [132], [141], [142]. However, deep-learning-based data augmentation is a more advanced technique, which is based generally on neural style transfer, autoencoders, and GANs to produce new unseen samples [143]–[145]. Many data augmentation techniques are presented very well in [146].

GAN is the most used deep-learning-based architecture for data augmentation due to its high ability to generate fake images that are very similar to real ones. Various GAN-based architectures were proposed over the few past years. Some researchers benefit from the GAN architecture to improve the performance of deep learning object detectors. Shen *et al.* [143] showed how multicondition constrained GAN (MCGAN) improved vehicle detection from aerial images. Chen *et al.* [147] proposed a data augmentation framework based on classification-oriented super-resolution GANs (CSRGANs) combined with a flexible object proposal generation to detect small-scale vehicles in UAV-based images. In order to generate new realistic remote sensing labeled vehicle images, Zheng *et al.* [148] proposed another data augmentation approach called vehicle synthesis GANs (VS-GANs).

Before doing data augmentation, we must consider the application that we want to develop. For example, we cannot do flipping for digit recognition because it will be confusing to distinguish between 9 and 6. Moreover, we cannot generate new unreal images in the healthcare field because it gives false and undesirable results.

### *D. Other Deep-Learning-Based Techniques to Improve Vehicle Detection Performance From UAV Videos*

In video sequences, CNN-based detectors have some drawbacks, such as classifying the same detected object into different classes between consecutive frames. Many studies have shown that recurrent networks achieve better results in the field of object detection from videos. In addition, other studies have tried to combine CNN and RNN characteristics to improve detection accuracy. RNN network and its types, including LSTM, show good performance on object tracking tasks in video sequences. Thus, their characteristics help to improve the detector performance by predicting the location of the vehicle in the next frames resulting in correct classifications between successive frames. Lakhal *et al.* [149] proposed the CNR framework that combines CNN and LSTM architectures to improve vehicle classification performance from remote sensing images. Ning *et al.* [150] developed ROLO architecture stacking LSTM layers on the top of the YOLOv1 detector to improve object detection accuracy, including vehicles. Lu *et al.* [42] proposed a new approach combining a CNN-based detector (SSD) with an associated LSTM network to improve vehicle detection accuracy. Zhang *et al.* [151] combine fully CNNs (FCN) and LSTM to estimate the number of vehicles in dense scenes from low-resolution videos captured through city cameras.

### *E. Lightweight Detectors for Small Devices*

The visual data produced by UAVs need a strong combination of reliable software and powerful hardware to realize complex applications. Most of today's deep-learning-based vehicle detection algorithms are executed on sophisticated GPUs, which are much faster than GPPs and CPUs. The heavy operations of detection algorithms also lead to huge energy consumption and large storage. Real-time vehicle detection from UAVs has very strict requirements, such as huge computing requirements, low latency, power efficiency, and security.

UAV's data processing could be divided into two main groups, which are off-board processing and onboard processing. For the first one, the UAVs collect data over their sensors and send them to on-ground powerful workstations or on the cloud. This gives to the edge device, UAV in our case, the ability to save energy by offloading intensive compute operations. Even when we use workstations equipped with powerful GPUs, it still very hard to process some tasks in real time, while time is very crucial for any autonomous machine, including UAVs, cars, and robots. Maybe, this issue will be solved with the advance of the fourth industrial revolution, the 5G, and even with quantum computing. Other problems with the off-board data processing are the security issue and the denied-signal areas, where the transmitted data could be hacked or cannot be transmitted to the processing unit. Moreover, the wireless transmission takes additional time for data sending, which leads to an additional cost to the latency of the system.

In the second group, all the calculations are done inside the drone itself, and we call it also edge computing. However, the heavy detection algorithms have a lack of computing power and energy managing on small and low-power devices, while they are crucial for deep-learning-based vehicle detection. Much research has been done to solve these problems where several publications have appeared, in recent years, documenting lightweight versions of deep-learning-based detectors to improve the detection speed while keeping a competitive accuracy.

Ringwald *et al.* [152] proposed a powerful lightweight vehicle detector (UAV-Net), which is implemented on Nvidia Jetson TX2 achieving an average precision (AP) of 97.2% and an inference speed of 15.9 frames per second (fps). The UAV-Net is based on SSD architecture replacing VGG-16 with ZynqNet [153] and adapted to the unique characteristics of UAV imagery. Shen *et al.* [143] proposed a modified version of Faster R-CNN architecture, which is based on a lightweight deep CNN feature extractor (LD-CNN) and an MCGAN to improve the vehicle detection from aerial images. The LD-CNN achieved better performance than other deep CNN and lightweight CNN architectures (VGG-16, ResNet-50, DenseNet-121, MobileNet, and ShuffleNet-v2) while reducing the model size and the computation cost. Kyrkou *et al.* [154] developed four single-shot detectors, including SmallYoloV3, TinyYoloVoc, TinyYoloNet, and DroNet. They are based on Tiny-YOLO architectures while applying different: filters sizes and numbers, layers number, and input image sizes. They proved that these architectures work efficiently on lightweight embedded systems achieving good results even on Raspberry Pi 3. Azimi [155] proposed a modified version of the SSD detector, called ShuffleDet, which uses ShuffleNet architecture as a backbone network. The author showed that ShuffleDet can be executed in real-time achieving 14 fps on Nvidia Jetson TX2 while keeping a competitive accuracy.

## V. Benchmark Datasets and Evaluation Metrics

The dataset size, type, and quality play a vital role in the development of deep-learning-based vehicle detection algorithms from aerial images. There is a lack in the number of well-annotated aerial images, and most of them are based on satellites and/or airborne. However, the available UAV-based datasets are very limited although large-scale and challenging aerial vehicle detection datasets are crucial to develop this field. In this section, we present some common benchmark datasets and various evaluation metrics used to evaluate deep-learning-based vehicle detection algorithms.

### *A. Datasets*

*1) DLR 3K Munich Vehicle Aerial Image Dataset:* The DLR 3K Munich Vehicle Aerial Image Dataset [5] is one of the most used and challenging datasets for small-size vehicle detection in aerial images annotated with oriented bounding boxes. The DLR 3K dataset consists of 20 top views aerial images with a resolution of $(5616 \times 3744)$ pixels and

TABLE II

AERIAL IMAGES/VIDEOS DATASETS FOR VEHICLE DETECTION TASK (CITATION TILL APRIL 20, 2020)

| Dataset | View | Bounding Box | Platform | Altitude (m) | Number of Images | GSD (cm/pixel) | Number of Instances | Image size | Cites |
|---|---|---|---|---|---|---|---|---|---|
| DLR 3K Munich Vehicle Aerial Image Dataset [5] | Top view | Oriented | Airborne | 1000 | 20 | 13 | 14235 | 5616×3744 | 137 |
| VEDAI-512 [156] | Top view | Oriented | Satellite | - | ∼ 1200 | 25 | ∼ 12.5 | 512×512 | 146 |
| VEDAI-1024 [156] | | | | | | 12.5 | | 1024×1024 | |
| COWC [159] | Top view | Center point | Aerial | - | 53 | ∼ 15 | 32716 | 2000×2000 to 19,000×19,000 | 105 |
| DOTA [158] | Top view | Oriented | Different sensors platforms | - | 2806 | Various | 188282 | 800×800 to 4000×4000 | 245 |
| Stanford Drone [3] | Top view | Axis-aligned | UAV | ∼ 80 | 60 videos | - | - | 1400×1904 | 243 |
| PSU [78] | Top view | Axis-aligned | UAV | - | 270 | - | 4102 | 684×547 to 4000×2250 | 5 |
| CARPK [109] | Top view | Axis-aligned | UAV | ∼ 40 | 1448 | - | 89777 | 1280×720 | 70 |
| PUCPR+ | Front and side views | Axis-aligned | Camera from high building | ∼ 35 | 125 | - | 192216 | | |
| UAV123 [2] | Different views | Axis-aligned | UAV | 5 to 25 | 123 videos | - | - | 1280×720 to 3840×2160 | 384 |
| UAVDT [1] | Front, side, top views | Axis-aligned | UAV | 10 to more than 70 | 80000 | - | 840000 | 1080×540 | 72 |
| VisDrone2018 [4] | Different views | Axis-aligned | UAV | - | 10209 images and 263 videos | - | 2.5 M | 2000×1500 | 98 |
| VeRi [160] | Front and side views | Axis-aligned | Camera | - | 49325 | - | - | 1920×1080 | 160 |
| CyCAR [142] | Top view | Axis-aligned | UAV | 20 to 500 | 450 | - | ∼ 5000 | 1920 x 1080, 2704×1520, 3840×2160 | 1 |
| NWPU VHR-10 [161] | Top view | Axis-aligned | Satellite | - | 715 | ∼ 8 to ∼ 200 | ∼ 3800 | 533×597 to 1728×1028 | 241 |
| OIRDS [162] | Top view | Axis-aligned | Satellite | - | 900 | ∼ 8 to ∼ 30 | ∼ 1800 | - | 47 |

a ground sampling distance (GSD) of around 13 cm/pixel. It is acquired at an altitude of around 1000 m through an airplane over Munich (Germany) urban and residential areas using Canon Eos 1Ds Mark III camera. It contains seven vehicle categories, but only two classes are generally used: cars with 9300 instances and trucks with 160 instances. The other categories are neglected because of the very small number of annotated instances. Due to the limited number and the very large size of the dataset images, all the researchers divide these images into a group of smaller subimages. Moreover, data augmentation techniques are used to increase the small number of truck instances [123]. The DLR 3K dataset is considered as one of the most important datasets to develop and evaluate different object detection algorithms, as shown in Tables II and III.

*2) Vehicle Detection in Aerial Imagery (VEDAI) Dataset:* Razakarivony and Jurie [156] create another well-annotated dataset, called the VEDAI dataset, by cropping the very large satellite images provided by the Utah AGRC database [157] into more than 1200 smaller images with two different resolutions, VEDAI-512 and VEDAI-1024, and a GSD of around 25 and 12.5 cm/pixel, respectively. The VEDAI-512 is just the downscaled version of VEDAI-1024, which makes the target vehicles smaller and more challenging. This dataset consists of nine different classes of sparse small vehicles all along with various backgrounds and confused objects captured from the same distance to the ground and with no oblique views. VEDAI is basically used to train and test small-size vehicle detection algorithms. Many state-of-the-art studies are based on the VEDAI dataset as a baseline for object detection algorithm development (see Tables II and III).

*3) DOTA Dataset:* The Object deTection in Aerial images (DOTA) dataset was created recently by Xia *et al.* [158], which contains 15 object categories, including vehicles of different scales and orientations. Moreover, each object is annotated manually by experts with an oriented bounding box. It consists of 2806 aerial images captured through different platforms in multiple cities. Thus, the size of each image is varying between (800 × 800) and (4000 × 4000) pixels with multiple GSDs. DOTA dataset provides a good balance between small- and middle-size objects, which makes it very similar to real-world scenes.

*4) COWC Dataset:* Mundhenk *et al.* [159] created the Cars Overhead with Context (COWC) dataset, which consists of 53 TIFF images, from six various locations, with different resolutions varying from (2000 × 2000) to (19 000 × 19 000) pixels, and a GSD of around 15 cm/pixel. Instead of labeling the images with bounding boxes, the authors followed another

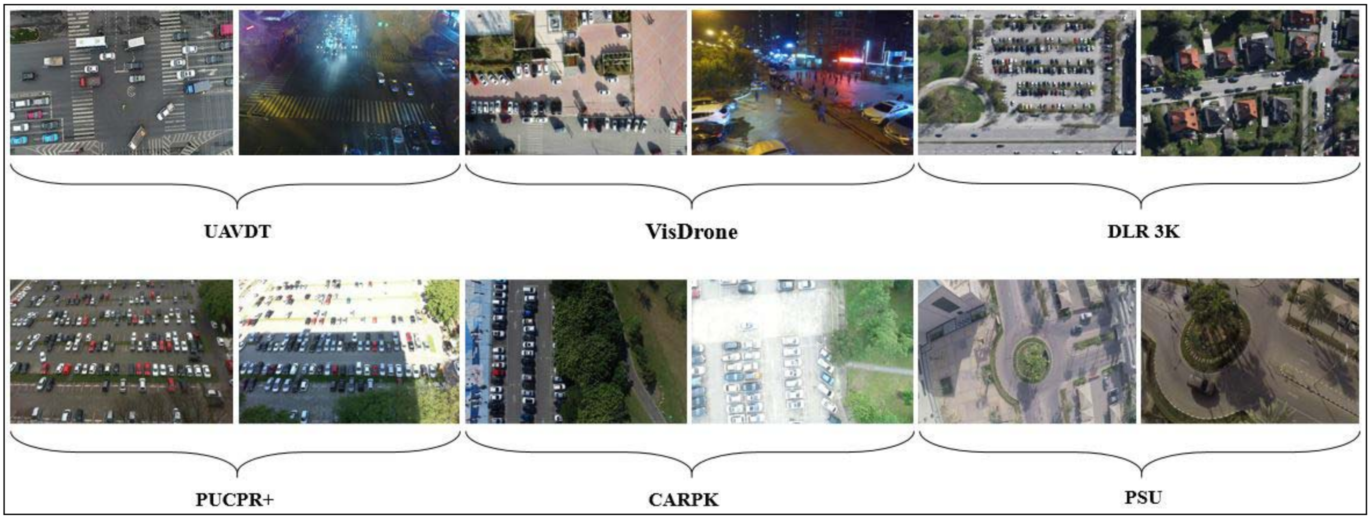


Fig. 7. Examples of some aerial datasets.

style of annotation, where they used the center pixel point of the vehicles. COWC dataset can be used for vehicle detection and counting tasks. However, it is a very challenging dataset due to the small size of cars that vary between 24 and 48 pixels.

*5) Stanford Drone Dataset:* The SDD [3] is one of the most popular UAV-based datasets that consists of around 60 annotated UAV video sequences with a resolution of (1400 × 1904) pixels, from eight unique scenes over the Stanford University campus. Each of these scenes is captured through a 3DR solo UAV equipped with a 4k camera at an altitude of 80 m. These videos contain different object categories, including 11 200 pedestrians, 6400 bicyclists, 1300 cars, 300 skateboarders, 200 golf carts, and 100 buses.

*6) CARPK and PUCPR+ Datasets:* The CARPK and PUCPR+ are publicly available datasets for vehicle detection tasks in different parking lots, which are published in the same paper [109]. CARPK is a large-scale UAV-based car parking dataset, which consists of around 90 000 cars in high-resolution images. It is captured through a UAV platform from four different parking lots with an altitude of around 40 m. However, the PUCPR+ dataset is captured through cameras installed at around 30 m on the top of a store building.

*7) VisDrone 2018 Dataset:* VisDrone 2018 [4] is a large-scale dataset that consists of 263 videos of around 179 000 frames and 10 209 static images with resolutions of (3840 × 2160) and (2000 × 1500) pixels, respectively. It contains ten object categories, including vehicles, pedestrians, and bicycles. All the videos and images are captured through different UAV platforms across many Chinese cities under different lighting and weather conditions. The VisDrone dataset targets four tasks, including object detection in images and videos, and single- and multiple-object trackings.

*8) CyCAR Dataset:* Kouris *et al.* [142] created a CyCAR dataset that consists of high-resolution images captured through a UAV platform over Nicosia city (Cyprus). It consists of 450 images that contain more than 5000 annotated vehicles collected from low (20 m) to high (500 m) altitudes.

*9) UA123 Dataset:* Mueller *et al.* [2] proposed, in their study, one of the largest low altitude multiclass UAV-based datasets, called UAV123. It contains 123 video sequences with 110 000 fully annotated frames, including vehicles. The videos are captured through different UAV platforms with different resolutions, varying between (1280 × 720) and (3840 × 2160) pixels, from various altitudes varying from 5 to 25 m. It is basically created for the single-object-tracking task.

*10) UAVDT Dataset:* Du *et al.* [1] created a new dataset, called UAVDT, which consists of 100 videos with 80k UAV-based images, with a resolution of (1080 × 540) pixels. These images are captured through UAV in complex scenarios from different altitudes varying from 10 to more than 70 m. All the images are annotated carefully with axis-aligned bounding boxes and some helpful attributes, including flying altitude, occlusion, vehicle category, weather condition, and camera view. This information makes UAVDT dataset useful for three main computer vision tasks: vehicle detection, single-vehicle tracking, and multiple-vehicle tracking.

The choice of the appropriate dataset depends on the problem that we are trying to solve. Other datasets are summarized in Table II, and Fig. 7 highlights the differences between some benchmark datasets.

### *B. Evaluation Metrics*

After developing the vehicle detection algorithms, their performance should be evaluated. Nowadays, there are various metrics to evaluate the quality of object detection algorithms, including mean AP (mAP), Intersection over Union (IoU), recall rate, precision rate, F1-score, and inference speed. In this section, we present the most important evaluation metrics.

*1) Intersection Over Union:* IoU is a technique used to determines whether detection is valid or not by comparing how good two different bounding boxes are for the detected object. As shown in (1), IoU is the ratio of the area of overlapping to the area of union between predicted (red box) and ground-truth (green box) boxes. Blue represents the intersection region, while orange represents the union of the two bounding boxes

$$\text{IoU} = \frac{\text{Area of overlap}}{\text{Area of union}}. \tag{1}$$

*2) True Positive, False Positive, True Negative, and False Negative:* The true positive (TP) value represents the number of vehicles correctly identified by the detector. Each bounding box is considered as a positive detection if the IoU is greater than a certain threshold. In addition, if there are many bounding boxes for the same detected object, only the one with the highest overlap ratio is taken as positive detection. However, false positive (FP) means the number of vehicles incorrectly identified by the detector. In other terms, the number of bounding boxes with an IoU is less than the chosen threshold, while false negative (FN) means the undetected vehicles, and true negative (TN) represents the correct misdetection.

*3) Recall Rate (Sensitivity):* The recall rate is an important metric to evaluate the detector performance. It represents how many of the predictive positive vehicles are really vehicles as follows:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{\text{TP}}{\text{All ground truth}}. \tag{2}$$

*4) Precision Rate:* The precision rate is another important evaluation metric, which determines the number of vehicles that are predicted to be truly positives over all detections as follows:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{\text{TP}}{\text{All detections}}. \tag{3}$$

*5) F1-Score:* The F1-score combines precision and recall rates into a single metric, which is the harmonic mean of these combined metrics as follows:

$$\text{F1-score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{4}$$

*6) Accuracy:* The accuracy metric represents how much correct detection over the total number of detections [see (5)]. The accuracy metric does not work well in the case of imbalanced data. In such a case, the F1-score metric is more reliable than the accuracy metric

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP}} = \frac{\text{Correct detections}}{\text{All detections}}. \tag{5}$$

*7) Average Precision:* The AP metric represents the area under the precision–recall curve; the higher AP the better performance is, and vice versa. Moreover, the mAP is obtained by calculating the AP for each class and then calculating the average between all the classes. It was first adopted by the PASCAL VOC 2007 dataset. The mAP metric is used to evaluate the object detection algorithm accuracy. Most of the time, the mAP is noted as mAP@0.5 or AP50. In some other cases, like in the case of COCO competition, AP@[.50:.95] or AP@[.50:0.05:.95] represents the same thing as mAP, where AP@[.50:0.05:.95] is the average value over multiple IoU of the results from 0.5 to 0.95 with a step size of 0.05.

*8) Frame Rate and Time/Image:* One of the biggest challenges of object detection algorithms is to make them work in real time. In order to express how fast the detector is, we use one of the most known evaluation metrics, called frame rate. The frame rate is the number of displayed frames in 1 s, which is measured in frame per second (FPS). The speed of a detection algorithm depends on the used hardware (GPU and CPU), the video/image resolution, and the detector architecture. The higher the frame rate, the better the detector we will get. Another frequently used evaluation metric is time per Image that expresses how fast a detector can process an input image of a certain resolution.

*9) Mean Absolute Error and Root Mean Squared Error:* The mean absolute error (MAE) and the root mean squared error (RMSE) are two metrics that are employed in object detection algorithms evaluation. MAE represents the average absolute difference between true values "$f_i''$" and predicted values "$y_i$" [see (6)]. Similarly, RMSE represents the average of the square root of the squared difference between the true and predicted values [see (7)]

$$\text{MAE} = \frac{1}{n}\sum_{i}^{n} |f_i - y_i| \tag{6}$$

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i}^{n}(f_i - y_i)^2}. \tag{7}$$

Table III shows the performance evaluation of various vehicle detection algorithms on the most used datasets, all along with the main backbone for each detector. In addition, the listed performances are achieved for a certain dataset that contains characteristics (the number of classes, the number of instances in each image, and resolution).

## VI. Discussion

In the last years, vehicle detection from UAV-based aerial images/videos systems gained high importance due to their versatility use in many studies. In this article, we presented several vehicle detection algorithms and the improvements that came with the different deep learning architectures. Since there are many challenges related to the type of used hardware, the UAV-based images/videos characteristics, the quality, the type, and the size of the dataset, we need to choose the appropriate system for the required task. Based on the challenges presented in Section III-B, there is a wide range of research directions that merits exploring. Table III shows various metrics related to the performances of vehicle detection algorithms on different datasets.

### A. Impact of Altitude

Unlike natural scene images/videos, UAV-based images contain vehicles of different sizes, scales, and orientations that increase the detection difficulty and make it more challenging. The flying altitude is an essential parameter, which affects the sizes of vehicles. Flying from low and medium altitudes provides vehicles' sizes very similar to the ones in natural scene images. However, the higher the distance from the

TABLE III
DIFFERENT DETECTORS WITH THEIR PERFORMANCES

| Ref | Method | Dataset | Backbone | Platform | MAE | RMSE | AP (%) | Recall (%) | Precision (%) | F1-score | Time/Image (s) | FPS |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| [127] | FCRN | DLR | VGG-16 | TITAN X | - | - | - | 90.51 | 93.3 | 0.92 | 9.7 | - |
| | | OIRDS | | | | | | 93.72 | 95.09 | - | - | |
| [119] | Modified RefineDet | Stanford | VGG-16 | GTX-1080Ti | - | - | 90.4 | - | - | - | - | 58 |
| | Faster R-CNN | | VGG-16 | | - | - | 75.3 | - | - | - | - | 7 |
| | | | ResNet-101 | | - | - | 75.9 | - | | - | - | 2.4 |
| | R-FCN | | ResNet-101 | | - | - | 86.6 | - | - | - | - | 9 |
| | FPN | | ResNet-101 | | - | - | 88.8 | - | - | - | - | 6 |
| | SSD | | VGG-16 | | - | - | 78.2 | - | - | - | - | 59 |
| | | | ResNet-101 | | - | - | 71.9 | - | - | - | - | 11.2 |
| | DSSD | | ResNet-101 | | - | - | 76.6 | - | - | - | - | 9.5 |
| | YOLOv2 | | Darknet-19 | | - | - | 55.8 | - | - | - | - | 64 |
| | YOLOv3 | | Darknet-53 | | - | - | 80 | - | - | - | - | 76 |
| | RefineDet | | VGG-16 | | - | - | 86.1 | - | - | - | - | 40 |
| | RON | | VGG-16 | | - | - | 69.2 | - | - | - | - | 15 |
| [110] | Faster R-CNN | DLR 702x624 | ZFNet | GTX-960 | - | - | 79.54 | 78.3 | 89.2 | 0.83 | 3.93 | - |
| [96] | Faster R-CNN | DLR 936x624 | Their own CNN | TITAN X | - | - | 91.4 | 85 | 92.8 | - | 0.537 per MPixel | - |
| | | VEDAI-512 | | | | | 89.5 | - | - | - | | |
| | | VEDAI-1024 | | | | | 95.2 | - | - | - | | |
| [123] | Faster R-CNN | DLR 936x624 | Their own CNN | - | - | - | 82.5 | 87,4 | 91,3 | - | - | - |
| [112] | Oriented_SSD300 | DLR 702x624 | VGG-16 | GTX-1060 | - | - | ∼71 | 72 | 81.25 | 0.76 | 4.50 | - |
| | | VEDAI-512 | | | | | - | 52.60 | 78.36 | 0.63 | 0.06 | - |
| | Oriented_SSD512 | DLR 702x624 | | | | | ∼76 | 78.84 | 85.53 | 0.82 | 5.17 | - |
| | | VEDAI-512 | | | | | - | 60.12 | 80.46 | 0.69 | 0.10 | - |
| [129] | AVPN_basic | DLR 702x624 | ZFNet | GTX-960 | - | - | ∼72 | 75.59 | 85.93 | 0.8 | 3.65 | - |
| | AVPN_basic+fast R-CNN | | | | | | ∼74 | 74.73 | 91.98 | 0.82 | 4.05 | - |
| | AVPN_large | | | | | | ∼76 | 77.02 | 87.81 | 0.82 | 3.65 | - |
| [152] | UAV-Net | DLR 936x624 | ZynqNet | TITAN X | - | - | 97.2 | - | - | - | - | 194.1 |
| | | | | GTX-1060 | | | | | | | | 83.8 |
| | | | | Jetson TX2 | | | | | | | | 15.9 |
| | | VEDAI-1024 | | TITAN X | - | - | 95.7 | - | - | - | - | 123.5 |
| | | | | GTX-1060 | | | | | | | | 50.2 |
| | | | | Jetson TX2 | | | | | | | | 9.9 |
| | | UAVDT | | TITAN X | - | - | 26.2 | - | - | - | - | 214 |
| | | | | GTX-1060 | | | | | | | | 98.8 |
| | | | | Jetson TX2 | | | | | | | | 18.3 |
| [155] | ShuffleDet | DLR 512x512 | ShuffleNet | Jetson TX2 | - | - | 62.9 | - | - | - | 0.524 | 14 |
| | | CARPK | | | 26.75 | 38.46 | - | - | - | - | - | |
| | | PUCPR+ | | | 41.58 | 49.68 | - | - | - | - | - | |
| [163] | SSD | UAV123 | VGG16 | TITAN X | - | - | 88.7 | - | - | - | - | 43 |
| [143] | Modified Faster R-CNN | DLR 702x624 | LD-CNN | TITAN XP | - | - | 86.9 | 83.1 | 92.5 | 0.875 | 1.64 | - |
| [111] | Faster R-CNN | DLR 702x624 | VGG-16 | GTX-1060 | - | - | 85.5 | 82.2 | 90.2 | 0.861 | 4.56 | - |
| [164] | Faster R-CNN | VEDAI-1024 | VGG-16 | - | - | - | 96 | 97 | 95 | 96 | - | - |
| [125] | DFRCNN | DLR 964x624 | VGG-16 with deconvolution Module | - | - | - | 98 | - | - | - | - | - |
| | | VEDAI-512 | | | | | 95 | | | | | |
| | | VEDAI-1024 | | | | | 97.8 | | | | | |
| [165] | Proposed detector | CARPK | - | TITAN X | 5.42 | 7.38 | 89.8 | - | - | - | - | 9 |
| | | PUCPR+ | | | 3.92 | 5.06 | 92.9 | | | | | |
| | | UAVDT | | | 7.67 | 10.95 | 60.1 | | | | | |
| | | VisDrone-2018 | | | 9.04 | 17.49 | 70.3 | | | | | |
| [128] | Coupled R-CNN | DLR | ZFNet | GTX-960 | - | - | - | 74 | 86.2 | 0.8 | 3.65 | - |
| [118] | Faster R-CNN | VEDAI-1024 | ZFNet | - | - | - | 66 | - | - | - | - | 14 |
| [166] | YOLOv3 | UAVDT | Darknet-53 | GTX-1080Ti | - | - | - | 88.6 | 96.5 | 0.92 | - | - |

TABLE III

*(Continued.)* DIFFERENT DETECTORS WITH THEIR PERFORMANCES

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| [114] | R3-Net | DLR 512x512 | ResNet-101 with FPN | GTX-1080Ti | - | - | 87 | 81.6 | - | 0.91 | - | 10.3 |
| | | VEDAI-512 | | | | | 64.8 | 59.2 | - | - | - | |
| [115] | Faster R-CNN + FPN | VEDAI-512 | ResNet-101 | - | - | - | 77.2 | 86.4 | 89.3 | 0.88 | - | - |
| [108] | YOLOv3 | CARPK | Darknet-53 | Jetson TX2 | 3.73 | 5.11 | - | 95 | 97 | - | - | 4 |
| | | PUCPR+ | | - | 1.8 | 2.74 | - | 86 | 95 | - | - | |
| [167] | GANet | UAVDT | ResNet-50 | TITAN XP | 5.09 | 8.16 | 46.8 | - | - | - | - | - |
| | | CARPK | | | 4.61 | 6.55 | 90.1 | | | | | |
| | | PUCPR+ | | | 3.28 | 4.96 | 91.4 | | | | | |
| [142] | Faster R-CNN | CyCAR | Inception-ResNet-v2 | Jetson TX2 | - | - | 89.9 | - | - | - | - | - |
| | | | Resnet-50 | | | | 78.95 | - | - | - | 0.439 | - |
| [126] | DYOLO | DOTA | Darknet-19 | TITAN X | - | - | 78.05 | - | - | - | - | - |
| [168] | SAR + Faster R-CNN | VEDAI-1024 | VGG-16 | TITAN X | - | - | 97.3 | - | - | - | 0.194 | - |
| [1] | Faster R-CNN | UAVDT | VGG-16 | GTX-1080Ti | - | - | 22.32 | - | - | - | - | 2.75 |
| | SSD | | - | | | | 33.62 | - | - | - | - | 41.55 |
| | R-FCN | | ResNet-50 | | | | 34.35 | - | - | - | - | 4.65 |
| | RON | | - | | | | 21.59 | - | - | - | - | 11.11 |
| [132] | SAIC-FPN | VisDrone-2018 DET | ResNet-50 | GTX-1080Ti | - | - | 27.1 | - | - | - | $0.252 \sim 2.568$ | - |
| [169] | Faster R-CNN | VEDAI | ZFNet | TITAN XP | - | - | 64.7 | - | - | - | 0.09 | - |
| [170] | EOVNet-basic | DLR 448x448 | ResNet-50 | GTX-1080Ti | - | - | 52.6 | 87.8 | 88.6 | 0.882 | - | - |
| | EOVNet-HEM | | | | | | 53.4 | 86.4 | 91.9 | 0.891 | - | - |
| | EOVNet++ | | | | | | 53.6 | 87 | 92.3 | 0.896 | 0.109 | - |
| [116] | DFPN | VisDrone-2018 | ResNet | Jetson TX2 | - | - | 30.6 | - | - | - | - | 8.6 |
| | | | MobileNet-v1 | | | | 29.2 | - | - | - | - | 14 |
| [78] | YOLOv3 | PSU | Darknet-53 | GTX-1080 | - | - | 90.6 | 93 | 93 | - | - | 23 |
| | | Stanford | | | | | 13.8 | 15 | 87 | - | - | |
| | Faster R-CNN | PSU | - | | | | 70.4 | 71 | 97 | - | - | - |
| | | Stanford | | | | | 18.9 | 21 | 44 | - | - | - |
| [117] | Pyramid styled CNN | NWPU VHR-10 | VGG-16 | TITAN X | - | - | 85.14 | - | - | - | - | - |
| | | | ResNet-50 | | | | 83.15 | - | - | - | - | - |
| | | | ResNet-101 | | | | 83.59 | - | - | - | 0.088 | - |

ground, the smaller the size of the vehicles. From higher altitudes, the detector confuses the vehicles with other similar objects that have the same shape, resulting in a large number of false alarms (FP). As shown in Tables II and III, mostly, as the altitude increases, the detection performance decreases, and vice versa. The resolution of the captured images and chosen algorithms should be taken into account as shown in [96], [125], [152], and [165]. For example, Li *et al.* [114] showed that the altitude could affect the detector accuracy, where they applied the same architecture (R3Net) on two different datasets DLR and VEDAI, achieving 87% and 64.8%, respectively. These datasets have similar image resolutions ($1080 \times 540$) but captured at different altitudes, where the DLR was captured at an altitude of 1000 m, and the VEDAI was captured through satellites. Most vehicle detection algorithms perform very well on low altitude images, but this performance decreases on images from very high altitudes. In many works, such as [78], [165], and [167], the authors showed that vehicle detection from lower altitudes achieves better accuracy than from higher altitudes. In Sections IV-A and IV-B, we presented the most powerful techniques to improve small vehicle detection in aerial images. The best results on small-sized vehicle detection are based on the detection algorithms that use FPN architectures, narrow CNN architectures, or adding deconvolution modules to increase the feature map resolution. Yang *et al.* [119] achieved the best AP result of 88.8% on the Stanford dataset among the detectors that are based on ResNet-101 as a feature extractor by combining it with FPN architecture. Similarly, Li *et al.* [114] and Gu *et al.* [115] show the performance of FPN-based detector on small vehicle detection from VEDAI and DLR datasets. However, applying a deconvolution module could provide better results as shown in the paper of Sommer *et al.* [125], achieving an AP varying between 95% and 98% on the VEDAI and DLR datasets. Moreover, Sommer *et al.* in their studies [96] and [123], show the impact of shallow architecture on small vehicle detection performance achieving an AP that varies between 82.5% and 95.2% on the VEDAI and DLR datasets (see Table III).

### B. Impact of View Angles

One other property that defers UAV from on-ground vehicles is the degree of liberty that has an impact on the view angles (top view, side view, and front view) and orientation. As shown in Table III, most of the detectors achieved very acceptable results on datasets with top view scenes, such as in the case of DLR and Stanford datasets, where the AP can achieve around 98% [125] and 91% [119], respectively. However, most of them gave poor results when the used dataset consists of images/videos captured from different angles, such as in the case of UAVDT where they achieved AP from 21% [1] to 60% [165]. In addition, the diversity of view angles is not the only reason for the bad performance because there are many other parameters that can cause the low performance of detectors, including dataset size and quality.

### C. Impact of Image/Video Resolution

The input image resolution plays a crucial role in vehicle detection performance. For example, the same detector trained on the same datasets [96], [125], but with different resolutions (VEDAI-512 and VEDAI-1024), gives different results. The results presented in Table III shows that high-resolution images/videos give better results in term of accuracy, where the AP improves from 89.5% to 95.2% in [96] and from 95% to 97.8% in [125]. Also, as the resolution decreases, the vehicle size decreases, which led to an increase in the number of FP detection. In addition, as the image resolution increases, the detection speed decreases.

### D. Impact of Dataset Properties

As shown in Table II, each dataset has different properties, such as occlusion, shadow, illumination, resolution, and viewpoint variation. These properties have an impact on the performance of the detection algorithm. Table III shows that vehicle detection from datasets that have multiple properties achieved lower accuracy compared to other datasets with fewer properties. As presented in Section IV-C, data augmentation techniques also have a significant impact on the detection performance. The number of images used to train the model and the scene complexity are other parameters that we should take into consideration because they are able to affect vehicle detection performance. The dataset is usually divided into training and testing sets, where their sizes depend on the total number of images in the dataset. Amato *et al.* [108], Yang *et al.* [119], and Chen *et al.* [163] split the dataset into a training set of 80% and a testing set of 20%. Also, Amato *et al.* [108] split the PUCPR+ dataset into 70% and 30% for training and testing sets, respectively.

### E. Impact of Backbone Architecture

Another important parameter is the chosen CNN architecture, which has an impact on the detection performance in terms of accuracy and speed. Table III shows that the backbone network architectures have an obvious impact on detection accuracy. Kyrkou *et al.* [154] showed the impact of changing CNN architecture on the detection performance. Yang *et al.* [119] showed that ResNet-101 architecture provides slightly better accuracy than VGG-16 in the case of Faster R-CNN detector. In the same paper, they showed that VGG-16 provides better results than ResNet-101 in the case of an SSD detector. However, in both cases, VGG-16 provides a better inference speed than ResNet-101. Table III shows that the backbone architectures' selection has a direct impact on vehicle detection performance. As an example, in Section IV-B, we presented that FPN and shallow CNN architectures provide better results on small vehicle detection. Yang *et al.* [119] and Kouris *et al.* [142] got better accuracy just by changing the backbone architecture, but it could affect the detection speed. Yang *et al.* [119] achieved seven and two fps in the case of Faster R-CNN using VGG-16 and ResNet-101, respectively. The same thing also happened with SSD using VGG-16 and ResNet-101 as backbone architectures achieving 59 and 11.2 fps.

### F. Impact on Real-Time Detection

The majority of works sacrifice speed to improve detection accuracy. However, any robotic system, including UAVs, needs to do the perception and decision-making and take actions in real time, especially in the case of small devices-based platforms. Image/video resolution and backbone architecture have serious impacts on the inference speed. As shown in Table III, two-stage detectors achieved interesting accuracies in vehicle detection from UAV tasks. However, they have a lack in terms of inference speed, which makes them not suitable for vehicle detection in real time even when we use powerful GPUs. One-stage detectors are solutions to achieve vehicle detection in real time while keeping a competitive accuracy result. Most detectors achieve real-time detection speed only on powerful GPUs. The SSD detector in [163] achieved 43 FPS while keeping a good accuracy of around 89% on the UAV123 dataset. A modified version of RefineDet [119] achieved 58 FPS and an accuracy of around 90% on Nvidia GTX1080Ti GPU. In [152], the proposed UAV-Net that is based on lightweight CNN architecture achieved competitive accuracy on three different datasets. It achieved 97.2%, 95.7%, and 26.2% on DLR, VEDAI, and UAVDT datasets, respectively, while providing a remarkable speed even on a small device (Jetson TX2) achieving 15.9, 9.9, and 18.3 FPS on DLR, VEDAI-1024, and UAVDT datasets, respectively. Moreover, Ringwald *et al.* [152] showed that, as the image resolution increases, the detection speed decreases. The resolutions of the cropped DLR 3K ($936 \times 624$) and UAVDT ($1024 \times 540$) images are almost the same, which gave an approximate speed detection of 194.1 and 214 fps on TITAN X GPU, respectively. However, the VEDAI-1024 ($1024 \times 1024$) has a bigger resolution that causes more processing time, resulting in less speed detection of 123.5 fps on the same GPU platform (TITAN X).

The different algorithms developed with the aim to improve the performance of vehicle detection from UAV imagery/videos (see Section IV) have some advantages and drawbacks. One of these algorithms is anchors' sizes changing, which could be a good solution for small vehicle detection in UAV-based scenes. However, it cannot be generalized for multiscale vehicle detection when the UAV flights at different altitudes. This is the case when Cai *et al.* [167] applied it on the UAVDT dataset (see Table III). Setting a large number of anchors with different scales could be a solution to minimize the flexibility issue. However, it still not sufficient to enhance detection accuracy.

Shallow architectures and feature maps from early layers could be another solution to improve small vehicle detection, but they still suffer from a high number of false-positive due to the appearance resemblance of some objects with vehicles. These drawbacks can lead to low recall and precision rates, as shown in [96], [129], and [123] (see Table III). Fusing feature maps from different layers could be a very good solution to improve the recall rate. However, it is not the only reason for the low recall rate, where it could be a compilation of different parameters, such as the image resolution, GSD, and flight altitude. It seems that FPN-based architectures provide the best results for multiscale vehicle detection.

Also, two- and one-stage algorithms have advantages and drawbacks. The two-stage detectors achieved higher accuracy of more than 88% using the FPN detector on the Stanford dataset compared to around 80% applying YOLOv3 on the same dataset. However, YOLOv3 achieved 76 fps against six fps of the FPN detector. Recently, one-stage detectors compete with two-stage detectors in terms of accuracy, especially with the advance of the new YOLOv4 that achieved remarkable results both in accuracy and detection speed.

Most reliable detectors are unsuitable for small devices and have real-time detection issues. For this reason, lightweight models are developed, which provide remarkable results concerning real-time detection, even on small devices. However, they still some critical issues concerning accuracy. For example, ShuffleNet [155] and MobileNet [116] achieve a detection speed near-real-time of 14 fps on Jetson TX2 but with low accuracy of 62.9% on the DLR dataset and 29.2% on the VisDrone-2018, respectively.

In order to realize an application, we need to take all of these parameters into consideration according to the chosen application. For these reasons, many factors should be taken into account, including the used platform, the targeted application, and the detection speed. For example, we should select PUCPR+ or CARPK datasets for parking-related applications. We must select lightweight architectures for real-time vehicle detection from autonomous UAVs or other small systems. Deeper and heavier algorithms could be more efficient for offline applications that demand more accuracy.

Object detection from UAV imagery is still an open research field, where the performance of the developed detectors evolves continuously. The vehicle detection performance depends on several parameters, including the complexity of the scene, the size of the vehicles, and the dataset quality and size, among others.

The choice of the right dataset is one of the most important steps in vehicle detection from UAV imagery. The number of images in a specific dataset could affect the performance of the detector, especially in complex environments. A large number of training sets may improve the detection accuracy, but the model takes a longer time in the training process. However, the trained model on small datasets can suffer in terms of accuracy due to the lack of information about the targeted object. Similarly, the image resolution could affect the speed and accuracy of the detector, where low-resolution images are processed faster, but they achieve poor accuracy due to the lack of valuable information. High-resolution images provide more valuable information, while they take a very long time to be processed. Also, the flight altitude is another fundamental parameter that can affect the detection accuracy due to the variation of vehicle size and scene complexity, where detecting vehicles from high altitude is more difficult than from low altitude. Several methods were proposed in the literature to solve such problems from image resolution choice [96] to the choice of the right architecture [114], [119], [125].

## VII. Conclusion

The UAV is one of the indispensable systems in the new intelligent era. They are being adopted in several sensitive areas and fields, including agriculture, search and rescue, military, and even discovering Mars. Artificial intelligence and deep learning techniques play an important role to develop these systems and make them smarter to facilitate many operations. In this review article, we provided an overview of some powerful deep learning architectures, including CNN, RNN, autoencoders, and GANs. These architectures are considered as the cornerstone of modern vehicle detection algorithms, including region-based detectors (Faster R-CNN and RFCN) and one-stage detectors (SSD, YOLO, and RefineDet). We showed that deep-learning-based vehicle detection from UAV images achieved interesting results in many tasks. Moreover, we presented different benchmark datasets all along with their characteristics to evaluate the developed models. In this review article, we tend to help researchers choosing the appropriate architecture and dataset for their applications.

Future works should focus on improving lightweight detectors for real-time vehicle detection, which could be implemented on the UAV itself instead of sending captured videos to the on-ground workstation or doing the processing on the cloud. Moreover, improving vehicle detection in the dense environment and vehicle tracking are fundamental issues that should be addressed in the future.

## References

[1] D. Du *et al.*, "The unmanned aerial vehicle benchmark: Object detection and tracking," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 370–386, doi: 10.1007/978-3-030-01249-6_23.

[2] M. Mueller, N. Smith, and B. Ghanem, "A benchmark and simulator for UAV tracking," in *Proc. Eur. Conf. Comput. Vis.*, vol. 9905, 2016, pp. 445–461, doi: 10.1007/978-3-319-46448-0_27.

[3] A. Robicquet, A. Sadeghian, A. Alahi, and S. Savarese, "Learning social etiquette: Human trajectory understanding in crowded scenes," in *Proc. Eur. Conf. Comput. Vis.*, in Lecture Notes in Computer Science, vol. 9912, 2016, pp. 549–565, doi: 10.1007/978-3-319-46484-8_33.

[4] P. Zhu, L. Wen, X. Bian, H. Ling, and Q. Hu, "Vision meets drones: A challenge," 2018, *arXiv:1804.07437*. [Online]. Available: http://arxiv.org/abs/1804.07437

[5] K. Liu and G. Mattyus, "Fast multiclass vehicle detection on aerial images," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 9, pp. 1938–1942, Sep. 2015, doi: 10.1109/LGRS.2015.2439517.

[6] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998, doi: 10.1109/5.726791.

[7] S. Mujawar, D. Kiran, and H. Ramasangu, "An efficient CNN architecture for image classification on FPGA accelerator," in *Proc. 2nd Int. Conf. Adv. Electron., Comput. Commun. (ICAECC)*, Feb. 2018, pp. 2018–2021, doi: 10.1109/ICAECC.2018.8479517.

[8] Y. Peng *et al.*, "FB-CNN: Feature fusion-based bilinear CNN for classification of fruit fly image," *IEEE Access*, vol. 8, pp. 3987–3995, 2020, doi: 10.1109/ACCESS.2019.2961767.

[9] M. Blot, M. Cord, and N. Thome, "Max-min convolutional neural networks for image classification," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2016, pp. 3678–3682.

[10] A. G. Howard, "Some improvements on deep convolutional neural network based image classification," in *Proc. 2nd Int. Conf. Learn. Represent. (ICLR)*, 2014, pp. 1–6.

[11] J. Li, X. Liang, S. Shen, T. Xu, J. Feng, and S. Yan, "Scale-aware fast R-CNN for pedestrian detection," *IEEE Trans. Multimedia*, vol. 20, no. 4, pp. 985–996, Apr. 2018, doi: 10.1109/TMM.2017.2759508.

[12] T. Agrawal and S. Urolagin, "Multi-angle parking detection system using mask R-CNN," in *Proc. 2nd Int. Conf. Big Data Eng. Technol.*, Jan. 2020, pp. 76–80, doi: 10.1145/3378904.3378914.

[13] W. Zhang, S. Wang, S. Thachan, J. Chen, and Y. Qian, "Deconv R-CNN for small object detection on remote sensing images," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2018, pp. 2491–2494.

[14] B. N. K. Sai and T. Sasikala, "Object detection and count of objects in image using tensor flow object detection API," in *Proc. Int. Conf. Smart Syst. Inventive Technol. (ICSSIT)*, Nov. 2019, pp. 542–546, doi: 10.1109/ICSSIT46314.2019.8987942.

[15] B. Kayalibay, G. Jensen, and P. van der Smagt, "CNN-based segmentation of medical imaging data," *CoRR*, vol. abs/1701.03056, pp. 1–24, Jan. 2017. [Online]. Available: http://arxiv.org/abs/1701.03056

[16] M. Vardhana, N. Arunkumar, S. Lasrado, E. Abdulhay, and G. Ramirez-Gonzalez, "Convolutional neural network for bio-medical image segmentation with hardware acceleration," *Cognit. Syst. Res.*, vol. 50, pp. 10–14, Aug. 2018, doi: 10.1016/j.cogsys.2018.03.005.

[17] Y. Tian, G. Yang, Z. Wang, E. Li, and Z. Liang, "Instance segmentation of apple flowers using the improved mask R-CNN model," *Biosyst. Eng.*, vol. 193, pp. 264–278, May 2020, doi: 10.1016/j.biosystemseng.2020.03.008.

[18] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, vol. 8689, 2014, pp. 818–833, doi: 10.1007/978-3-319-10590-1_53.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 1, pp. 1097–1105, May 2012, doi: 10.1145/3065386.

[20] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9, doi: 10.1109/CVPR.2015.7298594.

[21] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, 2015, pp. 1–14.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778, doi: 10.1109/CVPR.2016.90.

[23] A. G. Howard *et al.*, "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*. [Online]. Available: http://arxiv.org/abs/1704.04861

[24] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4510–4520, doi: 10.1109/CVPR.2018.00474.

[25] A. Howard *et al.*, "Searching for MobileNetV3," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1314–1324, doi: 10.1109/ICCV.2019.00140.

[26] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5 MB model size," *CoRR*, vol. abs/1602.07360, pp. 1–13, Nov. 2016. [Online]. Available: http://arxiv.org/abs/1602.07360

[27] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6848–6856, doi: 10.1109/CVPR.2018.00716.

[28] N. Ma, X. Zhang, H. T. Zheng, and J. Sun, "ShuffleNet V2: Practical guidelines for efficient CNN architecture design," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 116–131, doi: 10.1007/978-3-030-01264-9_8.

[29] R. J. Wang, X. Li, and C. X. Ling, "Pelee: A real-time object detection system on mobile devices," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 1963–1972.

[30] J. Li, R. Zhao, H. Hu, and Y. Gong, "Improving RNN transducer modeling for end-to-end speech recognition," in *Proc. IEEE Autom. Speech Recognit. Understand. Workshop (ASRU)*, Dec. 2019, pp. 114–121.

[31] A. Amberkar, P. Awasarmol, G. Deshmukh, and P. Dave, "Speech recognition using recurrent neural networks," in *Proc. Int. Conf. Current Trends Towards Converging Technol. (ICCTCT)*, Mar. 2018, pp. 2018–2021, doi: 10.1109/ICCTCT.2018.8551185.

[32] H. Wang, H. Wang, and K. Xu, "Evolutionary recurrent neural network for image captioning," *Neurocomputing*, vol. 401, pp. 249–256, Aug. 2020, doi: 10.1016/j.neucom.2020.03.087.

[33] M. Wang, L. Song, X. Yang, and C. Luo, "A parallel-fusion RNN-LSTM architecture for image caption generation," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2016, pp. 4448–4452, doi: 10.1109/ICIP.2016.7533201.

[34] Y. Li and J. Yang, "Hydrological time series prediction model based on attention-LSTM neural network," in *Proc. 2nd Int. Conf. Mach. Learn. Mach. Intell.*, Sep. 2019, pp. 21–25, doi: 10.1145/3366750.3366756.

[35] Y. Chen and K. Wang, "Prediction of satellite time series data based on long short term memory-autoregressive integrated moving average model (LSTM-ARIMA)," in *Proc. IEEE 4th Int. Conf. Signal Image Process. (ICSIP)*, Jul. 2019, pp. 308–312, doi: 10.1109/SIPROCESS.2019.8868350.

[36] X. Song *et al.*, "Time-series well performance prediction based on long short-term memory (LSTM) neural network model," *J. Petroleum Sci. Eng.*, vol. 186, Mar. 2020, Art. no. 106682, doi: 10.1016/j.petrol.2019.106682.

[37] L. Yao and Y. Guan, "An improved LSTM structure for natural language processing," in *Proc. IEEE Int. Conf. Saf. Produce Informatization (IICSPI)*, Dec. 2018, pp. 565–569, doi: 10.1109/IICSPI.2018.8690387.

[38] J. Li, Y. Xu, and H. Shi, "Bidirectional LSTM with hierarchical attention for text classification," in *Proc. IEEE 4th Adv. Inf. Technol., Electron. Automat. Control Conf. (IAEAC)*, Dec. 2019, pp. 456–459, doi: 10.1109/IAEAC47372.2019.8997969.

[39] A. F. Ganai and F. Khursheed, "Predicting next word using RNN and LSTM cells: Stastical language modeling," in *Proc. 5th Int. Conf. Image Inf. Process. (ICIIP)*, Nov. 2019, pp. 469–474, doi: 10.1109/iciip47207.2019.8985885.

[40] C. Su, H. Huang, S. Shi, P. Jian, and X. Shi, "Neural machine translation with Gumbel tree-LSTM based encoder," *J. Vis. Commun. Image Represent.*, vol. 71, Aug. 2020, Art. no. 102811, doi: 10.1016/j.jvcir.2020.102811.

[41] C. Zhang and J. Kim, "Modeling long- and short-term temporal context for video object detection," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2019, pp. 71–75, doi: 10.1109/ICIP.2019.8802920.

[42] Y. Lu, C. Lu, and C.-K. Tang, "Online video object detection using association LSTM," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2344–2352, doi: 10.1109/ICCV.2017.257.

[43] A. Carrio, C. Sampedro, A. Rodriguez-Ramos, and P. Campoy, "A review of deep learning methods and applications for unmanned aerial vehicles," *J. Sensors*, vol. 2017, pp. 1–13, Aug. 2017, doi: 10.1155/2017/3296874.

[44] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[45] J. Li, X. Liang, Y. Wei, T. Xu, J. Feng, and S. Yan, "Perceptual generative adversarial networks for small object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1222–1230.

[46] C.-Y. Liou, W.-C. Cheng, J.-W. Liou, and D.-R. Liou, "Autoencoder for words," *Neurocomputing*, vol. 139, pp. 84–96, Sep. 2014, doi: 10.1016/j.neucom.2013.09.055.

[47] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, no. 12, pp. 3371–3408, Dec. 2010.

[48] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, "Stacked convolutional auto-encoders for hierarchical feature extraction," in *Proc. Int. Conf. Artif. Neural Netw.*, 2011, pp. 52–59, doi: 10.1007/978-3-642-21735-7_7.

[49] J. Xu *et al.*, "Stacked sparse autoencoder (SSAE) for nuclei detection on breast cancer histopathology images," *IEEE Trans. Med. Imag.*, vol. 35, no. 1, pp. 119–130, Jan. 2016, doi: 10.1109/TMI.2015.2458702.

[50] M. Schreyer, T. Sattarov, D. Borth, A. Dengel, and B. Reimer, "Detection of anomalies in large scale accounting data using deep autoencoder networks," 2017, *arXiv:1709.05254*. [Online]. Available: http://arxiv.org/abs/1709.05254

[51] Y. Pu *et al.*, "Variational autoencoder for deep learning of images, labels and captions," in *Proc. 30th Int. Conf. Neural Inf. Process.*, 2016, pp. 2360–2368.

[52] P. Zhu, L. Wen, D. Du, X. Bian, Q. Hu, and H. Ling, "Vision meets drones: Past, present and future," 2020, *arXiv:2001.06303*. [Online]. Available: http://arxiv.org/abs/2001.06303

[53] K. Bajaj, D. K. Singh, and M. A. Ansari, "Autoencoders based deep learner for image denoising," *Procedia Comput. Sci.*, vol. 171, pp. 1535–1541, Jan. 2020, doi: 10.1016/j.procs.2020.04.164.

[54] Z. Fang, T. Jia, Q. Chen, M. Xu, X. Yuan, and C. Wu, "Laser stripe image denoising using convolutional autoencoder," *Results Phys.*, vol. 11, pp. 96–104, Dec. 2018, doi: 10.1016/j.rinp.2018.08.023.

[55] Y. Qiu, Y. Yang, Z. Lin, P. Chen, Y. Luo, and W. Huang, "Improved denoising autoencoder for maritime image denoising and semantic segmentation of USV," *China Commun.*, vol. 17, no. 3, pp. 46–57, Mar. 2020, doi: 10.23919/JCC.2020.03.005.

[56] R. K. Keser and B. U. Toreyin, "Autoencoder based dimensionality reduction of feature vectors for object recognition," in *Proc. 15th Int. Conf. Signal-Image Technol. Internet-Based Syst. (SITIS)*, Nov. 2019, pp. 577–584, doi: 10.1109/sitis.2019.00097.

[57] C. Hu, X. Hou, and Y. Lu, "Improving the architecture of an autoencoder for dimension reduction," in *Proc. IEEE 11th Int. Conf. Ubiquitous Intell. Comput., IEEE 11th Int. Conf. Autonomic Trusted Comput., IEEE 14th Int. Conf. Scalable Comput. Commun. Associated Workshops*, Dec. 2014, pp. 855–858, doi: 10.1109/UIC-ATC-ScalCom.2014.50.

[58] J. Zabalza *et al.*, "Novel segmented stacked autoencoder for effective dimensionality reduction and feature extraction in hyperspectral imaging," *Neurocomputing*, vol. 185, pp. 1–10, Apr. 2016, doi: 10.1016/j.neucom.2015.11.044.

[59] V. Kuppili, D. R. Edla, and A. Bablani, "Novel fitness function for 3D image reconstruction using bat algorithm based autoencoder," in *Proc. 23rd Int. ACM Conf. 3D Web Technol.*, Jun. 2018, pp. 2–3, doi: 10.1145/3208806.3211218.

[60] C. C. Tan and C. Eswaran, "Reconstruction of handwritten digit images using autoencoder neural networks," in *Proc. Can. Conf. Electr. Comput. Eng.*, May 2008, pp. 465–470, doi: 10.1109/CCECE.2008.4564577.

[61] Z. Dong and W. Qu, "Infrared image colorization using an edge aware auto encoder decoder with the multi-resolution fusion," in *Proc. Chin. Automat. Congr. (CAC)*, Nov. 2019, pp. 1011–1016, doi: 10.1109/CAC48633.2019.8996588.

[62] I. J. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst.*, vol. 2, 2014, pp. 2672–2680.

[63] F. Emmert-Streib, Z. Yang, H. Feng, S. Tripathi, and M. Dehmer, "An introductory review of deep learning for prediction models with big data," *Frontiers Artif. Intell.*, vol. 3, pp. 1–23, Feb. 2020, doi: 10.3389/frai.2020.00004.

[64] M. Bugdol, Z. Segiet, M. Kręcichwost, and P. Kasperek, "Vehicle detection system using magnetic sensors," *Transp. Problems*, vol. 9, no. 1, pp. 49–60, 2014.

[65] S. S. M. Ali, B. George, L. Vanajakshi, and J. Venkatraman, "A multiple inductive loop vehicle detection system for heterogeneous and laneless traffic," *IEEE Trans. Instrum. Meas.*, vol. 61, no. 5, pp. 1353–1360, May 2012, doi: 10.1109/TIM.2011.2175037.

[66] M. Hickman and P. Mirchandani, "Airborne traffic flow data and traffic management," in *Proc. Greenshields Symp.*, 2008, pp. 121–132.

[67] J. Leitloff, D. Rosenbaum, F. Kurz, O. Meynberg, and P. Reinartz, "An operational system for estimating road traffic information from aerial images," *Remote Sens.*, vol. 6, no. 11, pp. 11315–11341, Nov. 2014, doi: 10.3390/rs61111315.

[68] K. V. Najiya and M. Archana, "UAV video processing for traffic surveillence with enhanced vehicle detection," in *Proc. 2nd Int. Conf. Inventive Commun. Comput. Technol. (ICICCT)*, Apr. 2018, pp. 662–668, doi: 10.1109/ICICCT.2018.8473204.

[69] M. Elloumi, R. Dhaou, B. Escrig, H. Idoudi, and L. A. Saidane, "Monitoring road traffic with a UAV-based system," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2018, pp. 1–6, doi: 10.1109/WCNC.2018.8377077.

[70] H. Zhang, M. Liptrott, N. Bessis, and J. Cheng, "Real-time traffic analysis using deep learning techniques and UAV based video," in *Proc. 16th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS)*, Sep. 2019, pp. 1–5, doi: 10.1109/AVSS.2019.8909879.

[71] X. Zhao, F. Pu, Z. Wang, H. Chen, and Z. Xu, "Detection, tracking, and geolocation of moving vehicle from UAV using monocular camera," *IEEE Access*, vol. 7, pp. 101160–101170, 2019, doi: 10.1109/access.2019.2929760.

[72] S. Li, W. Zhang, G. Li, L. Su, and Q. Huang, "Vehicle detection in UAV traffic video based on convolution neural network," in *Proc. IEEE 1st Conf. Multimedia Inf. Process. Retr. (MIPR)*, Apr. 2018, pp. 1–6, doi: 10.1109/MIPR.2018.00009.

[73] B. Coifman, M. McCord, R. G. Mishalani, and K. Redmill, "Surface transportation surveillance from unmanned aerial vehicles," in *Proc. 83rd Annu. Meeting Transp. Res. Board*, 2004, p. 28.

[74] B. Coifman, M. McCord, R. G. Mishalani, M. Iswalt, and Y. Ji, "Roadway traffic monitoring from an unmanned aerial vehicle," *IEE Proc. Intell. Transp. Syst.*, vol. 153, no. 1, pp. 11–20, Mar. 2006.

[75] X. Xi, Z. Yu, Z. Zhan, Y. Yin, and C. Tian, "Multi-task cost-sensitive-convolutional neural network for car detection," *IEEE Access*, vol. 7, pp. 98061–98068, 2019, doi: 10.1109/ACCESS.2019.2927866.

[76] R. Ke, Z. Li, S. Kim, J. Ash, Z. Cui, and Y. Wang, "Real-time bidirectional traffic flow parameter estimation from aerial videos," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 4, pp. 890–901, Apr. 2017, doi: 10.1109/TITS.2016.2595526.

[77] Q. Pan, X. Wen, Z. Lu, L. Li, and W. Jing, "Dynamic speed control of unmanned aerial vehicles for data collection under Internet of Things," *Sensors*, vol. 18, no. 11, pp. 1–18, 2018, doi: 10.3390/s18113951.

[78] B. Benjdira, T. Khursheed, A. Koubaa, A. Ammar, and K. Ouni, "Car detection using unmanned aerial vehicles: Comparison between faster R-CNN and YOLOv3," in *Proc. 1st Int. Conf. Unmanned Vehicle Systems-Oman (UVS)*, Feb. 2019, pp. 1–6, doi: 10.1109/UVS.2019.8658300.

[79] A. Ayalew, "A review on object detection from unmanned aerial vehicle using CNN," *Int. J. Advance Res., Ideas Innov. Technol.*, vol. 5, no. 4, pp. 241–243, 2019.

[80] Y. Koga, H. Miyazaki, and R. Shibasaki, "A CNN-based method of vehicle detection from aerial images using hard example mining," *Remote Sens.*, vol. 10, no. 1, p. 124, Jan. 2018, doi: 10.3390/rs10010124.

[81] Y. Xu, G. Yu, Y. Wang, X. Wu, and Y. Ma, "Car detection from low-altitude UAV imagery with the faster R-CNN," *J. Adv. Transp.*, vol. 2017, pp. 1–10, Aug. 2017, doi: 10.1155/2017/2823617.

[82] H. Nguyen, "Improving faster R-CNN framework for fast vehicle detection," *Math. Problems Eng.*, vol. 2019, pp. 1–11, Nov. 2019, doi: 10.1155/2019/3808064.

[83] K. Shi, H. Bao, and N. Ma, "Forward vehicle detection based on incremental learning and fast R-CNN," in *Proc. 13th Int. Conf. Comput. Intell. Secur. (CIS)*, Dec. 2017, pp. 73–76.

[84] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 5693–5703.

[85] Y. Li, Y. Chen, N. Wang, and Z.-X. Zhang, "Scale-aware trident networks for object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6054–6063.

[86] Z. Cheng, Y. Wu, Z. Xu, T. Lukasiewicz, and W. Wang, "Segmentation is all you need," 2019, *arXiv:1904.13300*. [Online]. Available: http://arxiv.org/abs/1904.13300

[87] Z. Li, C. Peng, G. Yu, X. Zhang, Y. Deng, and J. Sun, "DetNet: Design backbone for object detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 334–350.

[88] H. Law and J. Deng, "CornerNet: Detecting objects as paired keypoints," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 734–750.

[89] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587, doi: 10.1109/CVPR.2014.81.

[90] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, Sep. 2015, doi: 10.1109/TPAMI.2015.2389824.

[91] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448, doi: 10.1109/ICCV.2015.169.

[92] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017, doi: 10.1109/TPAMI.2016.2577031.

[93] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object detection via region-based fully convolutional networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 379–387.

[94] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2961–2969, doi: 10.1109/ICCV.2017.322.

[95] L. Sommer, T. Schuchert, and J. Beyerer, "Comprehensive analysis of deep learning-based vehicle detection in aerial images," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 9, pp. 2733–2747, Sep. 2019, doi: 10.1109/TCSVT.2018.2874396.

[96] L. W. Sommer, T. Schuchert, and J. Beyerer, "Fast deep vehicle detection in aerial images," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2017, pp. 311–319, doi: 10.1109/WACV.2017.41.

[97] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *Int. J. Comput. Vis.*, vol. 104, no. 2, pp. 154–171, Sep. 2013, doi: 10.1007/s11263-013-0620-5.

[98] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788, doi: 10.1109/CVPR.2016.91.

[99] W. Liu *et al.*, "SSD: Single shot MultiBox detector," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 21–37, doi: 10.1007/978-3-319-46448-0_2.

[100] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6517–6525, doi: 10.1109/CVPR.2017.690.

[101] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*. [Online]. Available: http://arxiv.org/abs/1804.02767

[102] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2980–2988, doi: 10.1109/ICCV.2017.324.

[103] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," 2020, *arXiv:2004.10934*. [Online]. Available: http://arxiv.org/abs/2004.10934

[104] A. Ammar, A. Koubaa, M. Ahmed, and A. Saad, "Aerial images processing for car detection using convolutional neural networks: Comparison between faster R-CNN and YoloV3," 2019, *arXiv:1910.07234*. [Online]. Available: http://arxiv.org/abs/1910.07234

[105] M. Radovic, O. Adarkwa, and Q. Wang, "Object recognition in aerial images using convolutional neural networks," *J. Imag.*, vol. 3, no. 2, p. 21, Jun. 2017, doi: 10.3390/jimaging3020021.

[106] J. Lu *et al.*, "A vehicle detection method for aerial image based on YOLO," *J. Comput. Commun.*, vol. 6, no. 11, pp. 98–107, 2018, doi: 10.4236/jcc.2018.611009.

[107] T. Tang, Z. Deng, S. Zhou, L. Lei, and H. Zou, "Fast vehicle detection in UAV images," in *Proc. Int. Workshop Remote Sens. With Intell. Process. (RSIP)*, May 2017, pp. 1–5, doi: 10.1109/RSIP.2017.7958795.

[108] G. Amato, L. Ciampi, F. Falchi, and C. Gennaro, "Counting vehicles with deep learning in onboard UAV imagery," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, Jun. 2019, pp. 1–5, doi: 10.1109/ISCC47284.2019.8969620.

[109] M.-R. Hsieh, Y.-L. Lin, and W. H. Hsu, "Drone-based object counting by spatially regularized regional proposal network," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 4165–4173, doi: 10.1109/ICCV.2017.446.

[110] T. Tang, S. Zhou, Z. Deng, H. Zou, and L. Lei, "Vehicle detection in aerial images based on region convolutional neural networks and hard negative example mining," *Sensors*, vol. 17, no. 2, p. 336, Feb. 2017, doi: 10.3390/s17020336.

[111] J. Shen, N. Liu, H. Sun, X. Tao, and Q. Li, "Vehicle detection in aerial images based on hyper feature map in deep convolutional network," *IEEE Access*, vol. 13, no. 4, pp. 1989–2011, 2019, doi: 10.3837/tiis.2019.04.014.

[112] T. Tang, S. Zhou, Z. Deng, L. Lei, and H. Zou, "Arbitrary-oriented vehicle detection in aerial imagery with single convolutional neural networks," *Remote Sens.*, vol. 9, no. 11, p. 1170, Nov. 2017, doi: 10.3390/rs9111170.

[113] Y. Guo, Y. Xu, and S. Li, "Dense construction vehicle detection based on orientation-aware feature fusion convolutional neural network," *Autom. Construct.*, vol. 112, Apr. 2020, Art. no. 103124, doi: 10.1016/j.autcon.2020.103124.

[114] Q. Li, L. Mou, Q. Xu, Y. Zhang, and X. X. Zhu, "R3-Net: A deep network for multioriented vehicle detection in aerial images and videos," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 7, pp. 5028–5042, Jul. 2019, doi: 10.1109/TGRS.2019.2895362.

[115] Y. Gu, B. Wang, and B. Xu, "A FPN-based framework for vehicle detection in aerial images," in *Proc. 2nd Int. Conf. Video Image Process.*, Dec. 2018, pp. 60–64, doi: 10.1145/3301506.3301531.

[116] S. Vaddi, C. Kumar, and A. Jannesari, "Efficient object detection model for real-time UAV applications," 2019, *arXiv:1906.00786*. [Online]. Available: http://arxiv.org/abs/1906.00786

[117] H. Tayara and K. Chong, "Object detection in very high-resolution aerial images using one-stage densely connected feature pyramid network," *Sensors*, vol. 18, no. 10, p. 3341, Oct. 2018, doi: 10.3390/s18103341.

[118] S. Maiti, P. Gidde, S. Saurav, S. Singh, Dhiraj, and S. Chaudhury, "Real-time vehicle detection in aerial images using skip-connected convolution network with region proposal networks," in *Pattern Recognition and Machine Intelligence* (Lecture Notes in Computer Science), vol. 11941. Cham, Switzerland: Springer, 2019.

[119] J. Yang, X. Xie, and W. Yang, "Effective contexts for UAV vehicle detection," *IEEE Access*, vol. 7, pp. 85042–85054, 2019, doi: 10.1109/ACCESS.2019.2923407.

[120] X. Xie *et al.*, "Real-time vehicle detection from UAV imagery," in *Proc. IEEE 4th Int. Conf. Multimedia Big Data (BigMM)*, Sep. 2018, pp. 1–5, doi: 10.1109/BigMM.2018.8499466.

[121] A. Ghodrati, A. Diba, M. Pedersoli, T. Tuytelaars, and L. Van Gool, "DeepProposals: Hunting objects and actions by cascading deep convolutional layers," *Int. J. Comput. Vis.*, vol. 124, no. 2, pp. 115–131, Sep. 2017, doi: 10.1007/s11263-017-1006-x.

[122] L. Wang, J. Liao, and C. Xu, "Vehicle detection based on drone images with the improved faster R-CNN," in *Proc. 11th Int. Conf. Mach. Learn. Comput. (ICMLC)*, 2019, pp. 466–471, doi: 10.1145/3318299.3318383.

[123] L. W. Sommer, T. Schuchert, and J. Beyerer, "Deep learning based multi-category object detection in aerial images," *Proc. SPIE*, vol. 10202, May 2017, Art. no. 1020209, doi: 10.1117/12.2262083.

[124] C. Herrmann, D. Willersinn, and J. Beyerer, "Low-resolution convolutional neural networks for video face recognition," in *Proc. 13th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS)*, Aug. 2016, pp. 221–227, doi: 10.1109/AVSS.2016.7738017.

[125] L. Sommer, A. Schumann, T. Schuchert, and J. Beyerer, "Multi feature deconvolutional faster R-CNN for precise vehicle detection in aerial imagery," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2018, pp. 635–642, doi: 10.1109/WACV.2018.00075.

[126] O. Acatay, L. Sommer, A. Schumann, and J. Beyerer, "Comprehensive evaluation of deep learning based detection methods for vehicle detection in aerial imagery," in *Proc. 15th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS)*, Nov. 2018, pp. 1–6, doi: 10.1109/AVSS.2018.8639127.

[127] H. Tayara, K. Gil Soo, and K. T. Chong, "Vehicle detection and counting in high-resolution aerial images using convolutional regression neural network," *IEEE Access*, vol. 6, pp. 2220–2230, 2018, doi: 10.1109/ACCESS.2017.2782260.

[128] T. Tang, S. Zhou, Z. Deng, L. Lei, and H. Zou, "Fast multidirectional vehicle detection on aerial images using region based convolutional neural networks," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Jul. 2017, pp. 1844–1847, doi: 10.1109/IGARSS.2017.8127335.

[129] Z. Deng, H. Sun, S. Zhou, J. Zhao, and H. Zou, "Toward fast and accurate vehicle detection in aerial images using coupled region-based convolutional neural networks," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 10, no. 8, pp. 3652–3664, Aug. 2017, doi: 10.1109/JSTARS.2017.2694890.

[130] S. Zhang, L. Wen, X. Bian, Z. Lei, and S. Z. Li, "Single-shot refinement neural network for object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4203–4212, doi: 10.1109/CVPR.2018.00442.

[131] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2117–2125, doi: 10.1109/CVPR.2017.106.

[132] J. Zhou, C.-M. Vong, Q. Liu, and Z. Wang, "Scale adaptive image cropping for UAV object detection," *Neurocomputing*, vol. 366, pp. 305–313, Nov. 2019, doi: 10.1016/j.neucom.2019.07.073.

[133] Z. Wu, K. Suresh, P. Narayanan, H. Xu, H. Kwon, and Z. Wang, "Delving into robust object detection from unmanned aerial vehicles: A deep nuisance disentanglement approach," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1201–1210, doi: 10.1109/ICCV.2019.00129.

[134] H. Eriş and U. Çevik, "Implementation of target tracking methods on images taken from unmanned aerial vehicles," in *Proc. IEEE 17th World Symp. Appl. Mach. Intell. Inform. (SAMI)*, Jan. 2019, pp. 311–316, doi: 10.1109/SAMI.2019.8782768.

[135] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255, doi: 10.1109/cvprw.2009.5206848.

[136] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, Jun. 2010, doi: 10.1007/s11263-009-0275-4.

[137] T. Y. Lin *et al.*, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 740–755, doi: 10.1007/978-3-319-10602-1_48.

[138] F. Kamran, M. Shahzad, and F. Shafait, "Automated military vehicle detection from low-altitude aerial images," in *Proc. Digit. Image Comput., Techn. Appl. (DICTA)*, Dec. 2018, pp. 1–8, doi: 10.1109/DICTA.2018.8615865.

[139] Z. Xu, H. Shi, N. Li, C. Xiang, and H. Zhou, "Vehicle detection under UAV based on optimal dense YOLO method," in *Proc. 5th Int. Conf. Syst. Informat. (ICSAI)*, Nov. 2018, pp. 407–411, doi: 10.1109/ICSAI.2018.8599403.

[140] Y. Zhou, T. Rui, Y. Li, and X. Zuo, "A UAV patrol system using panoramic stitching and object detection," *Comput. Electr. Eng.*, vol. 80, Dec. 2019, Art. no. 106473, doi: 10.1016/j.compeleceng.2019.106473.

[141] L. Sommer, K. Nie, A. Schumann, T. Schuchert, and J. Beyerer, "Semantic labeling for improved vehicle detection in aerial imagery," in *Proc. 14th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS)*, Aug. 2017, pp. 1–6, doi: 10.1109/AVSS.2017.8078510.

[142] A. Kouris, C. Kyrkou, and C.-S. Bouganis, "Informed region selection for efficient UAV-based object detectors: Altitude-aware vehicle detection with CyCAR dataset," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Nov. 2019, pp. 51–58, doi: 10.1109/IROS40897.2019.8967722.

[143] J. Shen, N. Liu, H. Sun, and H. Zhou, "Vehicle detection in aerial images based on lightweight deep convolutional network and generative adversarial network," *IEEE Access*, vol. 7, pp. 148119–148130, 2019, doi: 10.1109/ACCESS.2019.2947143.

[144] A. Kompella and R. V. Kulkarni, "A semi-supervised recurrent neural network for video salient object detection," *Neural Comput. Appl.*, vol. 33, pp. 1–19, Jun. 2020, doi: 10.1007/s00521-020-05081-5.

[145] X. Feng, Q. M. Jonathan Wu, Y. Yang, and L. Cao, "An autuencoder-based data augmentation strategy for generalization improvement of DCNNs," *Neurocomputing*, vol. 402, pp. 283–297, Aug. 2020, doi: 10.1016/j.neucom.2020.03.062.

[146] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *J. Big Data*, vol. 6, no. 1, Dec. 2019, Art. no. 60, doi: 10.1186/s40537-019-0197-0.

[147] Y. Chen, J. Li, Y. Niu, and J. He, "Small object detection networks based on classification-oriented super-resolution GAN for UAV aerial imagery," in *Proc. Chin. Control Decis. Conf. (CCDC)*, Jun. 2019, pp. 4610–4615, doi: 10.1109/CCDC.2019.8832735.

[148] K. Zheng, M. Wei, G. Sun, B. Anas, and Y. Li, "Using vehicle synthesis generative adversarial networks to improve vehicle detection in remote sensing images," *ISPRS Int. J. Geo-Inf.*, vol. 8, no. 9, p. 390, Sep. 2019, doi: 10.3390/ijgi8090390.

[149] M. I. Lakhal, S. Escalera, and H. Cevikalp, "CRN: End-to-end convolutional recurrent network structure applied to vehicle classification," in *Proc. 13th Int. Joint Conf. Comput. Vis., Imag. Comput. Graph. Theory Appl.*, 2018, pp. 137–144, doi: 10.5220/0006533601370144.

[150] G. Ning *et al.*, "Spatially supervised recurrent convolutional neural networks for visual object tracking," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2017, pp. 1–4, doi: 10.1109/ISCAS.2017.8050867.

[151] S. Zhang, G. Wu, J. P. Costeira, and J. M. F. Moura, "FCN-rLSTM: Deep spatio-temporal neural networks for vehicle counting in city cameras," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 3687–3696, doi: 10.1109/ICCV.2017.396.

[152] T. Ringwald, L. Sommer, A. Schumann, J. Beyerer, and R. Stiefelhagen, "UAV-Net: A fast aerial vehicle detector for mobile platforms," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2019, pp. 544–552, doi: 10.1109/CVPRW.2019.00080.

[153] D. Gschwend, "ZynqNet: An FPGA-accelerated embedded convolutional neural network," Aug. 2020, *arXiv:2005.06892*. [Online]. Available: http://arxiv.org/abs/2005.06892

[154] C. Kyrkou, G. Plastiras, T. Theocharides, S. I. Venieris, and C.-S. Bouganis, "DroNet: Efficient convolutional neural network detector for real-time UAV applications," in *Proc. Design, Automat. Test Eur. Conf. Exhib. (DATE)*, Mar. 2018, pp. 967–972, doi: 10.23919/DATE.2018.8342149.

[155] S. M. Azimi, "ShuffleDet: Real-time vehicle detection network in on-board embedded UAV imagery," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 88–99, doi: 10.1007/978-3-030-11012-3_7.

[156] S. Razakarivony and F. Jurie, "Vehicle detection in aerial imagery: A small target detection benchmark," *J. Vis. Commun. Image Represent.*, vol. 34, pp. 187–203, Jan. 2016, doi: 10.1016/j.jvcir.2015.11.002.

[157] *Utah Mapping Portal*. Accessed: Jun. 12, 2020. [Online]. Available: https://gis.utah.gov/

[158] G.-S. Xia *et al.*, "DOTA: A large-scale dataset for object detection in aerial images," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3974–3983, doi: 10.1109/CVPR.2018.00418.

[159] T. N. Mundhenk, G. Konjevod, W. A. Sakla, and K. Boakye, "A large contextual dataset for classification, detection and counting of cars with deep learning," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 785–800, doi: 10.1007/978-3-319-46487-9_48.

[160] X. Liu, W. Liu, H. Ma, and H. Fu, "Large-scale vehicle re-identification in urban surveillance videos," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jul. 2016, pp. 1–6, doi: 10.1109/ICME.2016.7553002.

[161] G. Cheng, J. Han, P. Zhou, and L. Guo, "Multi-class geospatial object detection and geographic image classification based on collection of part detectors," *ISPRS J. Photogramm. Remote Sens.*, vol. 98, pp. 119–132, Dec. 2014, doi: 10.1016/j.isprsjprs.2014.10.002.

[162] F. Tanner *et al.*, "Overhead imagery research data set—An annotated data library & tools to aid in the development of computer vision algorithms," in *Proc. Appl. Imag. Pattern Recognit. Workshop*, 2009, pp. 1–8, doi: 10.1109/AIPR.2009.5466304.

[163] W. Chen, Z. Baojun, T. Linbo, and Z. Boya, "Small vehicles detection based on UAV," *J. Eng.*, vol. 2019, no. 21, pp. 7894–7897, Nov. 2019, doi: 10.1049/joe.2019.0710.

[164] W. Sakla, G. Konjevod, and T. N. Mundhenk, "Deep multi-modal vehicle detection in aerial ISR imagery," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2017, pp. 916–923, doi: 10.1109/WACV.2017.107.

[165] W. Li, H. Li, Q. Wu, X. Chen, and K. N. Ngan, "Simultaneously detecting and counting dense vehicles from drone images," *IEEE Trans. Ind. Electron.*, vol. 66, no. 12, pp. 9651–9662, Dec. 2019, doi: 10.1109/TIE.2019.2899548.

[166] J. Wang, S. Simeonova, and M. Shahbazi, "Orientation- and scale-invariant multi-vehicle detection and tracking from unmanned aerial videos," *Remote Sens.*, vol. 11, no. 18, p. 2155, Sep. 2019, doi: 10.3390/rs11182155.

[167] Y. Cai *et al.*, "Guided attention network for object detection and counting on drones," 2019, *arXiv:1909.11307*. [Online]. Available: http://arxiv.org/abs/1909.11307

[168] L. Sommer, N. Schmidt, A. Schumann, and J. Beyerer, "Search area reduction fast-RCNN for fast vehicle detection in large aerial imagery," in *Proc. 25th IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2018, pp. 3054–3058, doi: 10.1109/ICIP.2018.8451189.

[169] M. Zhang, H. Li, G. Xia, W. Zhao, S. Ren, and C. Wang, "Research on the application of deep learning target detection of engineering vehicles in the patrol and inspection for military optical cable lines by UAV," in *Proc. 11th Int. Symp. Comput. Intell. Design (ISCID)*, Dec. 2018, pp. 97–101, doi: 10.1109/ISCID.2018.00029.

[170] Z. Gao, H. Ji, T. Mei, B. Ramesh, and X. Liu, "EOVNet: Earth-observation image-based vehicle detection network," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 12, no. 9, pp. 3552–3561, Sep. 2019, doi: 10.1109/JSTARS.2019.2933501.

[171] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2818–2826, doi: 10.1109/CVPR.2016.308.

[172] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-ResNet and the impact of residual connections on learning," 2016, *arXiv:1602.07261*. [Online]. Available: http://arxiv.org/abs/1602.07261

[173] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4700–4708.

[174] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proc. CVPR*, Jul. 2017, pp. 1492–1500.

[175] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. CVPR*, Jun. 2018, pp. 7132–7141.


**Abdelmalek Bouguettaya** received the master's and Ph.D. degrees in telecommunications from Badji Mokhtar University, Annaba, Algeria, in 2011 and 2017, respectively.

He was a member of the "Embedded and Detection" Advanced Systems Division, Laboratoire d'Etude et de Recherche en Instrumentation et en Communication d'Annaba (LERICA Laboratory), Annaba, from 2009 to 2018. He is currently a Senior Researcher with the Research Center in Industrial Technologies (CRTI), Chéraga, Algeria. His current research interests include deep learning, computer vision, intelligent embedded systems, and unmanned aerial vehicles (UAVs).

**Hafed Zarzour** received the Ph.D. degree in computer science from Annaba University, Annaba, Algeria, in 2013.

He is currently an Associate Professor of computer science with the University of Souk Ahras, Souk-Ahras, Algeria. He has published several research articles in international journals and conferences of high repute, including IEEE, Springer, Elsevier, Wiley, ACM, Taylor and Francis, IGI Global, and Inderscience. His research focuses on deep learning, artificial intelligence, and educational technology.

**Ahmed Kechida** received the M.S. degree in electronics from the Polytechnic Military School, Algiers, Algeria, in 2003 and the Ph.D. degree in electronics from Saad Dahleb Blida 1 University, Blida, Algeria, in 2016.

He is currently a Senior Researcher with Research Center in Industrial Technologies (CRTI), Algiers. His research interests include signal processing, image analysis, embedded systems for drones, and nondestructive evaluation by ultrasound.

**Amine Mohammed Taberkit** was born in Tlemcen, Algeria, in 1988. He received the master's degree in electronic instrumentation and the Ph.D. degree from the University of Tlemcen, Tlemcen, in 2012 and 2018, respectively.

In 2010, he became an Electronic Engineer at the University of Tlemcen. He was a member of the Research Unit of Materials and Renewable Energy Research (URMER) and a part-time Teacher with the University of Tlemcen from 2013 to 2019. He is currently a Senior Researcher with the Research Center in Industrial Technologies (CRTI), Chéraga, Algiers. His research interests included strain silicon technology, MOSFET transistors, and nanotechnology. His actual research interests include artificial intelligence, intelligent embedded systems, unmanned aerial vehicle, and computer vision.