Counting Vehicles with Deep Learning in Onboard UAV Imagery

1st Giuseppe Amato
Institute of Information Science and Technologies
National Research Council
Pisa, Italy
luca.ciampi@isti.cnr.it

2nd Luca Ciampi
Institute of Information Science and Technologies
National Research Council
Pisa, Italy
claudio.gennaro@isti.cnr.it

3rd Fabrizio Falchi
Institute of Information Science and Technologies
National Research Council
Pisa, Italy
giuseppe.amato@isti.cnr.it

4th Claudio Gennaro
Institute of Information Science and Technologies
National Research Council
Pisa, Italy
fabrizio.falchi@isti.cnr.it

***Abstract*—The integration of mobile and ubiquitous computing with deep learning methods is a promising emerging trend that aims at moving the processing task closer to the data source rather than bringing the data to a central node. The advantages of this approach range from bandwidth reduction, high scalability, to high reliability, just to name a few. In this paper, we propose a real-time deep learning approach to automatically detect and count vehicles in videos taken from a UAV (Unmanned Aerial Vehicle). Our solution relies on a convolutional neural network-based model fine-tuned to the specific domain of applications that is able to precisely localize instances of the vehicles using a regression approach, straight from image pixels to bounding box coordinates, reasoning globally about the image when making predictions and implicitly encoding contextual information. A comprehensive experimental evaluation on real-world datasets shows that our approach results in state-of-the-art performances. Furthermore, our solution achieves real-time performances by running at a speed of 4 Frames Per Second on an NVIDIA Jetson TX2 board, showing the potentiality of this approach for real-time processing in UAVs.**

***Index Terms*—Object Counting; Deep Learning; Convolutional Neural Networks; Onboard Embedded Processing; Real-time Vehicle Detection; Drones; UAV**

I. Introduction

Visual tasks like object detection, classification or segmentation are essential for many practical applications of great significance for unmanned aerial vehicles (i.e., drones) and other embedded mobile platforms. The solutions to these tasks commonly involve the onboard real-time processing of the acquired images and range from traditional computer vision algorithms to the more recent application of deep learning-based strategies [1], especially Convolutional Neural Networks (CNNs) [2]. Whereas traditional algorithms are optimized for a specific goal and particular conditions, CNNs are trained in a massive way to undertake more general challenges. Employing CNNs for the onboard real-time applications requires to face with limited processing resources. Anyway, although the training phase is usually highly demanding in terms of computation, the inference phase can be exploited using less sophisticated hardware resources.

In this work, we address the counting problem for evaluating the number of vehicles present in drone-based videos of parking areas, proposing a real-time approach to be used on board by mobile platforms such as Unmanned Aerial Vehicles (UAVs) (Fig. 1). Objects counting is an inter-disciplinary field and has been tackled in computer vision by various techniques. Most of the current methods are CNNs-based and basically, we can broadly classify them into two categories [3]: regression-based and detection-based approaches. In the first category, we learn a regression model that maps the high-dimensional features space of the image into non-negative counting numbers or into density maps, skipping the hard task of detecting instances of the objects. These techniques work very well in extremely overlapped scenarios where the single object instances are not well defined due to inter-class and intra-class occlusions, for example in the estimation of the number of people present at a public event. In the detection-based approaches we instead localize the instances of the objects and then we count them.

The scenario that we consider in this paper consists of images taken from a drone view of parking areas, so vehicles are not heavily overlapped (there are not intra-class occlusions), although cars appear in various orientations, often within the same scene, and can be partially occluded by trees and bridges; furthermore there is also clutter from motorbikes, buildings, different light conditions, etc. In order to address these challenges, we propose a CNN-based detection approach that is able to count cars in images localizing precisely the instances. Furthermore, our solution is fast and computationally inexpensive, suited for working in low-power embedded systems. We evaluate the effectiveness and reliability of our solution testing it on two publicly available car counting datasets: the CARPK dataset and the PUCPR+ dataset [4].

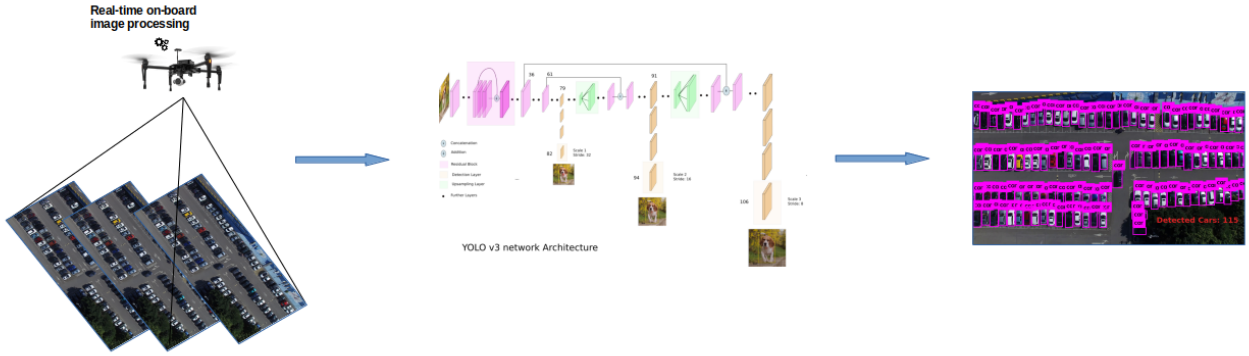


Fig. 1. System overview: we propose a real-time deep learning solution that is able to localize and count vehicles in videos taken by a drone. The image processing is performed directly on board the UAV equipped with a NVIDIA Jetson TX2.

In both cases, we achieve state-of-the-art accuracy by a large margin. For our tests we use the NVIDIA Jetson TX2, a power-efficient embedded AI computing device, running at 4 Frames Per Second (FPS).

II. Related Works

Objects counting has been addressed in computer vision by various techniques, especially for the estimation of the number of people in crowded scenes. Most of the state-of-the-art methods are CNNs-based since they are more robust to the typical challenges we must deal with, like variations in scales and perspectives, inter-object occlusions, non-uniform illumination of the scene, and many others. Following the taxonomy adopted in [3], we can broadly classify existing approaches into two categories: counting by regression and counting by detection.

Counting by *regression* is a supervised method that tries to establish a direct mapping (linear or not) from the image features to the number of objects present in the scene or to a corresponding density map (i.e. a continuous-valued function), skipping the hard task of detecting instances of the objects. The most influential work on counting via density map estimation is done by [5] where the authors employ a linear transformation between the pixel-level feature representations of the images and the associated density maps. Starting from this seminal work, many solutions are exploited in various application fields. In [6] authors count Antarctic penguins from images captured by smart cameras with the intention of monitoring the population of the continent. Authors in [7] propose a simple but effective solution for crowd counting in extremely overlapped scenarios exploiting a CNN with filters having receptive fields of different sizes, resulting to be robust to variations in people size due to perspective distortion of the scenes. The problem of counting cells in microscopy images is tackled for example by [8], where a convolutional neural network is used to regress a cell spatial density map across the image. Other interesting works are represented by [9], where authors develop a deep spatio-temporal neural networks to sequentially count vehicles from low-quality videos captured by city cameras in order to manage urban traffic, and by [10] that proposes a simple way to improve regression models for object counting by regulating activation maps from the final convolution layer of the network with coarse ground-truth activation maps generated from simple dot annotations (they call this strategy heatmap regulation (HR)).

Counting by *detection* is instead a supervised approach where we simply localize instances of the objects and then we count them. While regression-based techniques work very well in extremely overlapped scenarios where the single object instances are not well defined due to inter-class and intra-class occlusions, but they perform poorly in images having large perspective and oversized objects, the detection-based solutions are instead employed in scenarios not too crowded where challenges are instead represented by different light conditions, objects orientation and other clutters. An example of a detection-based solution is [4], where authors use a novel Layout Proposal Network (LPN) that counts and localizes vehicles in drone videos leveraging the spatial layout information (e.g., cars often park regularly).

Recently, authors of [11] propose a new approach that claims to be able to count objects belonging to a generic class not decided a priori - a class-agnostic counting network. To this end, the network must be adapted using very few annotated data. They achieved this recasting the counting problem as an image matching problem, where counting instances is performed by matching (self-similar patches) within the same image. This solution is tested counting bacterial cells, people and vehicles.

Most of the previously mentioned solutions are not suitable to work on embedded real-time devices, due to the time constraints and the limited available computing resources. In this work, we propose a real-time detection-based solution able to count vehicles of parking areas using a drone equipped with a low-computing embedded device. Prior methods able to perform onboard computations, like [12], assume that the locations of the monitored objects of a scene are already known in advance, and cast car counting as a classification problem, which makes conventional car counting methods not

directly applicable in unconstrained drone videos. The authors of [13] propose a more flexible approach, counting cars using semantic segmentation, but this technique is not suitable for a real-time task. Another detection-based solution presented in [14] proposes a computationally inexpensive network for vehicle detection in UAV imagery but it performs poorly.

III. Datasets

We evaluate our cars counting solution with two datasets widely used in our application context: CARPK and PUCPR+ [4]. The CARPK dataset is the first large-scale aerial dataset for counting cars in parking lots. It includes 989 and 459 training and test samples, respectively, each of resolution $720 \times 1,280$. The training images are taken from three different parking lot scenes, while the test set is taken from a fourth scene. The total number of car instances is 42,274 in the range [1,87] (i.e, from a minimum of one car up to a maximum of 87 cars in the whole lot) and in the test dataset is 47,500 in the range [2,188]. The PUCPR+ dataset is published in the same paper as the CARPK dataset. It is a subset of PUCPR dataset [15], adapted by the authors for the counting task. It contains images captured using a fixed camera from a height of the 10^{th} floor of a building, which provides a slanted view of the parking lot. This dataset has in total 100 and 25 training and test samples, respectively. Images are taken under three different weather conditions (sunny, rainy and cloudy), resulting in different illuminations of the scene. The total number of car instances in the training dataset is 1,299 in the range [0,331] and in the test dataset is 3,920 in the range [1,328]. Some sample images from the dataset are shown in Fig. 2.

IV. Methodology

Our solution is based on *YOLOv3* (You Only Look Once) [16], a popular deep convolutional neural network, employed in many detection systems and implemented using *darknet*, an open source neural network framework written in C and CUDA. Unlike previous detection methods based on classification using a sliding-window classifier or on region proposals, YOLO addresses the detection problem using a regression approach, straight from image pixels to bounding box coordinates and class probabilities, reasoning globally about the image when making predictions and implicitly encoding contextual information about classes.

YOLOv3 uses a custom deep architecture as the backbone for the features extraction, called *darknet-53*, having 53 layers. For the detection task, 53 more layers are stacked onto it, giving us a 106 layer fully convolutional underlying architecture. In particular, YOLOv3 makes detections at three scales, which are precisely given by down-sampling the dimensions of the input image by 32, 16 and 8. The three detections are made by the 82^{nd}, the 94^{th} and the 106^{th} layer, respectively, by applying 1x1 detection kernels on feature maps. Therefore, YOLOv3 is a fully convolutional neural network since there are not fully connected layers.

As a starting point, we considered a model of YOLO pre-trained on the COCO dataset [17], a large dataset composed of images describing complex everyday scenes of common objects in their natural context, categorized in 80 different categories. Since this network is a generic objects detector, we specialize it to recognize and localize object instances belonging to a specific category - i.e. the car category in our case.

In particular, we first extract the weights of the firsts 81 layers of the pre-trained model, since these layers capture universal features (like curves and edges) that are also relevant to our new problem. Then, we fine-tune YOLO initialing the firsts 81 layers with the previously extracted weights, and the weights associated with the remaining layers at random. In this way, we get the network to focus on learning the dataset-specific features in the last layers.

For the fine-tuning, we make use of the training subsets of the CARPK and PUCPR+ datasets. We define a single epoch as 10 passes over the training images and train for 100 epochs. We prevent over-fitting exploiting the test subsets, and accordingly to this, we select a specific model for the evaluation after a certain number of epochs.

V. Experiments

In this section, we discuss experiments and quantitative results. We first introduce the evaluation metrics and the experimental setup, then we compare and discuss the proposed method with state-of-the-art approaches on the two datasets described above.

A. Evaluation Metrics

Following other counting benchmarks, we use Mean Absolute Error (*MAE*) and Root Mean Square Error (*RMSE*) as the metrics for comparing the performance of our solution against other counting approaches present in literature. MAE is defined as follows:

$$MAE = \frac{1}{N}\sum_{n=1}^{N}|c_n^{gt} - c_n^{pred}| \tag{1}$$

while RMSE is defined as:

$$RMSE = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(c_n^{gt} - c_n^{pred})^2} \tag{2}$$

where N is the total number of test images, c_{gt} is the actual count, and c_{pred} is the predicted count of the n-th image. Note that as a result of the squaring of each difference, RMSE effectively penalizes large errors more heavily than small ones. Then RMSE should be more useful when large errors are particularly undesirable.

Since these metrics do not take into account the quality of the detections, we also use Precision and Recall as performance evaluators, defined as:

$$Precision = \frac{TPs}{TPs + FPs} \tag{3}$$

(a) (b) (c) (d)

Fig. 2. Examples from the CARPK (top row) and PUCPR+ (bottom row) datasets. The CARPK dataset is an aerial dataset taken from a drone, where the resolution of the cars, the viewpoint and the location vary. Furthermore, there is also clutter from motorbikes and buildings. Differently, the PUCPR+ dataset records a parking lot under three different weather conditions with a fixed camera, and there are no other dynamic objects present apart from cars.

$$Recall = \frac{TPs}{TPs + FNs} \tag{4}$$

where TPs are the True Positives, FPs the False Positives and FN the False Negatives of the individual car detections.

B. Experimental Setup

We use the darknet implementation of YOLO and the pre-trained model on COCO dataset. The network is fine-tuned using a NVIDIA GeForce RTX 2080 Ti GPU and evaluated on NVIDIA Jetson TX2 as an embedded edge device. For the optimization, we use stochastic gradient descent with a base learning rate of 0.001, momentum 0.9 and a weight decay of 0.0005. Moreover, the images are resized to 608x608 pixels along with their annotations. and a batch size of 64 is used.

C. Evaluation and Discussion

We test our solution on the CARPK and PUCPR+ datasets previously described, following the original experimental setups proposed by the authors in [4], and we compare our results with the ones obtained using state-of-the-art approaches. Results in terms of MAE, RMSE, Precision and Recall for the two datasets are reported in Table I and Table II, respectively, while Fig. 3 shows some detection results.

We achieve state-of-the-art results by a large margin in both the datasets, considering all the performance evaluators. The Generic Matching Network (GMN) presented in [11] and tested on the CARPK dataset has the advantage to be class agnostic. However, the error obtained using only three training images is about four times larger than the one obtained with our approach. The error using the full training dataset is instead about two times larger than the one we obtained, and in this case the benefits of using a class agnostic solution are less self-evident. The VGG-GAP-HR solution [10] has instead the advantage of needing a dot-annotated training dataset, while our solution needs bounding box labels. Dot-annotation is a less time-consuming operation compared with the bounding box one, but the error of our solution is about the half of the one obtained in [10] for the CARPK dataset, and about three times smaller for the PUCPR+ dataset. Finally, comparing our approach with the one proposed by [14] in terms of images processed per second, we can say that the algorithm in [14] is faster than ours (14 FPS instead of 4 FPS), but the error in [14] is about 7 times larger for the CARPK dataset and about 23 times larger for the PUCPR+ dataset.

VI. Conclusions

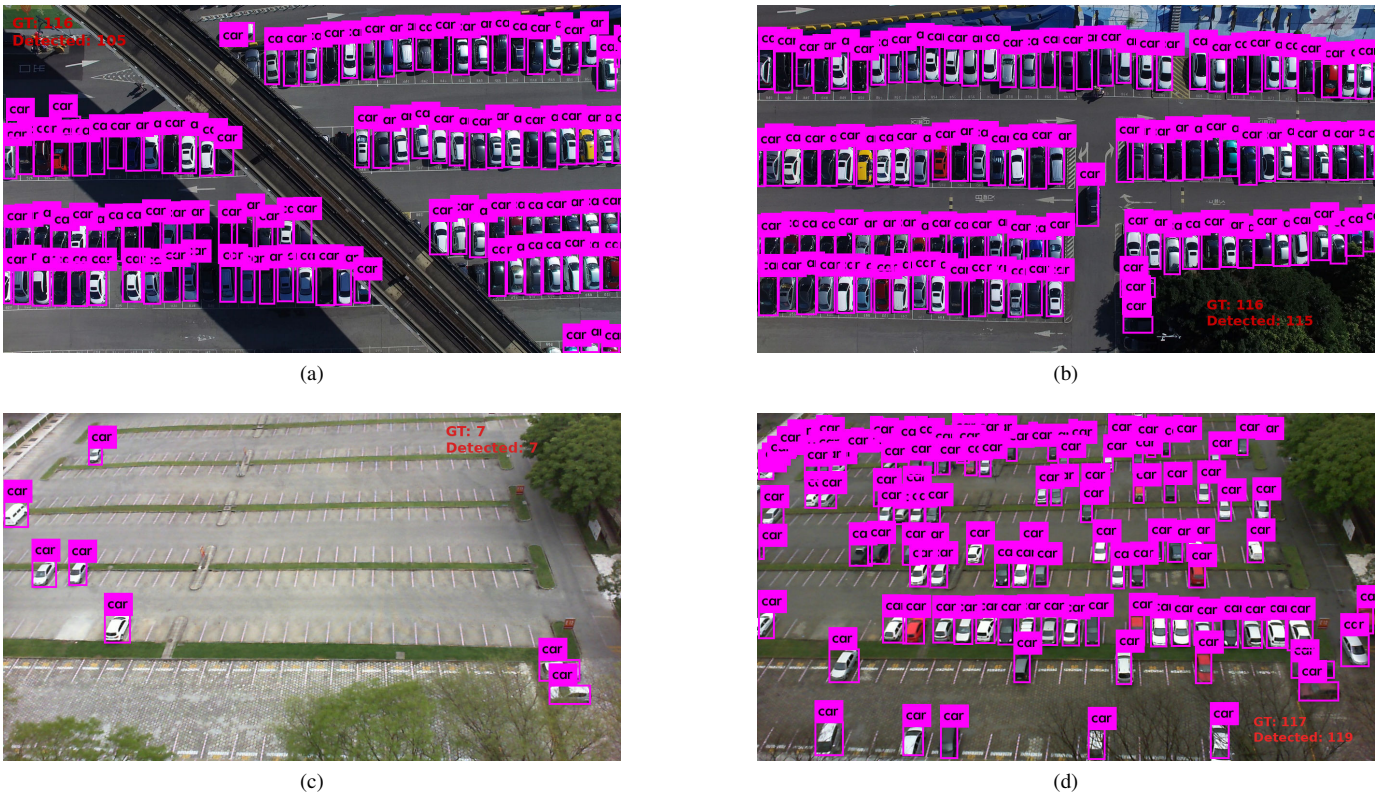
In this article, we presented a useful tool for counting objects as vehicles based on a state-of-the-art network for performing object detection. Our approach has proven to be competitive in terms of accuracy compared to other approaches in the literature. It should be noted, however, that although our approach outperforms the Generic Matching

TABLE I
RESULTS ON THE CARPK TEST SET (459 IMAGES AND 47500 TOTAL COUNTS)

Method	MAE	RMSE	Recall	Precision
Faster R-CNN [4], [18]	47.45	57.55	-	-
Faster R-CNN (RPN-small) [4], [18]	24.32	37.62	-	-
One-Look Regression [4], [19]	59.46	66.84	-	-
Spatially Regularized RPN [4]	23.80	36.79	57.5%	-
ShuffleDet [14]	26.75	38.46	-	-
VGG-GAP-HR [10]	7.88	9.30	-	-
GMN (3 images) [11]	13.38	18.03	76.1%	85.1%
GMN (full dataset) [11]	7.48	9.90	88.4%	91.8%
Our solution	**3.73**	**5.11**	**95%**	**97%**

TABLE II
RESULTS ON THE PUCPR+ TEST SET (25 IMAGES AND 3920 TOTAL COUNTS)

Method	MAE	RMSE	Recall	Precision
Faster R-CNN [4], [18]	156.76	200.59	-	-
Faster R-CNN (RPN-small) [4], [18]	39.88	47.67	-	-
One-Look Regression [4], [19]	21.88	36.73	-	-
Spatially Regularized RPN [4]	22.76	34.46	62.5%	-
ShuffleDet [14]	41.58	49.68	-	-
VGG-GAP-HR [10]	5.24	6.67	-	-
Our solution	**1.80**	**2.74**	**86%**	**95%**



(a) (b)

(c) (d)

Fig. 3. Sample results on the CARPK dataset (top row) and the PUCPR+ dataset (bottom row). Detections are marked with a bounding box and a label indicating the object class. In addition, we report the total number of instances found together with the Ground Truth value.

Network presented by Lu et al. [11], even with a comparable number of training images, their approach still remains valid in other scenarios where probably our solution could do not be suitable, such as cell and crowd counting, and in general in extremely crowded scenarios. Regarding the possibility of realizing an autonomous system located on board of a UAV using a mobile Jetson TX2 board, it is possible to guarantee an operating autonomy of six hours if, for example, a 3S 6000 mAh LiPo battery is used as proposed by [20].

ACKNOWLEDGMENT

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Jetson TX2 board used for this research.

REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[2] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[3] V. A. Sindagi and V. M. Patel, "A survey of recent advances in cnn-based single image crowd counting and density estimation," *Pattern Recognition Letters*, vol. 107, pp. 3–16, 2018.

[4] M.-R. Hsieh, Y.-L. Lin, and W. H. Hsu, "Drone-based object counting by spatially regularized regional proposal network," in *The IEEE International Conference on Computer Vision (ICCV)*, vol. 1, 2017.

[5] V. Lempitsky and A. Zisserman, "Learning to count objects in images," in *Advances in neural information processing systems*, 2010, pp. 1324–1332.

[6] C. Arteta, V. Lempitsky, and A. Zisserman, "Counting in the wild," in *European conference on computer vision*. Springer, 2016, pp. 483–498.

[7] Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma, "Single-image crowd counting via multi-column convolutional neural network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 589–597.

[8] W. Xie, J. A. Noble, and A. Zisserman, "Microscopy cell counting and detection with fully convolutional regression networks," *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization*, vol. 6, no. 3, pp. 283–292, 2016.

[9] S. Zhang, G. Wu, J. P. Costeira, and J. M. Moura, "Fcn-rlstm: Deep spatio-temporal neural networks for vehicle counting in city cameras," in *Computer Vision (ICCV), 2017 IEEE International Conference on*. IEEE, 2017, pp. 3687–3696.

[10] S. Aich and I. Stavness, "Improving object counting with heatmap regulation," *arXiv preprint arXiv:1803.05494*, 2018.

[11] E. Lu, W. Xie, and A. Zisserman, "Class-agnostic counting," *arXiv preprint arXiv:1811.00472*, 2018.

[12] G. Amato, F. Carrara, F. Falchi, C. Gennaro, C. Meghini, and C. Vairo, "Deep learning for decentralized parking lot occupancy detection," *Expert Systems with Applications*, vol. 72, pp. 327–334, 2017.

[13] L. Ciampi, G. Amato, F. Falchi, C. Gennaro, and F. Rabitti, "Counting vehicles with cameras," in *Proceedings of the 26th Italian Symposium on Advanced Database Systems, Castellaneta Marina (Taranto), Italy, June 24-27, 2018.*, 2018.

[14] S. M. Azimi, "Shuffledet: Real-time vehicle detection network in on-board embedded uav imagery," in *European Conference on Computer Vision*. Springer, 2018, pp. 88–99.

[15] P. R. de Almeida, L. S. Oliveira, A. S. Britto, E. J. Silva, and A. L. Koerich, "Pklot–a robust dataset for parking lot classification," *Expert Systems with Applications*, vol. 42, no. 11, pp. 4937–4949, 2015.

[16] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[17] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[18] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *CoRR*, vol. abs/1506.01497, 2015.

[19] 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," *CoRR*, vol. abs/1609.04453, 2016.

[20] B. Blanco-Filgueira, D. García-Lesta, M. Fernández-Sanjurjo, V. M. Brea, and P. López, "Deep learning-based multiple object visual tracking on embedded system for iot and mobile edge computing applications," *arXiv preprint arXiv:1808.01356*, 2018.