Pattern Recognition Letters 167 (2023) 45–52

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Automated labeling of training data for improved object detection in traffic videos by fine-tuned deep convolutional neural networks

Iván García-Aguilar[a,b,*], Jorge García-González[a,b], Rafael Marcos Luque-Baena[a,b], Ezequiel López-Rubio[a,b]

[a] Department of Computer Languages and Computer Science, University of Málaga, Bulevar Louis Pasteur, 35, Málaga 29071, Spain
[b] Biomedical Research Institute of Málaga (IBIMA), C/ Doctor Miguel Díaz Recio, 28, Málaga 29010, Spain

A R T I C L E I N F O

Article history:
Received 24 May 2021
Revised 30 September 2022
Accepted 23 January 2023
Available online 24 January 2023

Edited by: Maria De Marsico

Keywords:
Object detection
Small scale
Super-resolution
Convolutional neural networks

A B S T R A C T

The exponential increase in the use of technology in road management systems has led to real-time visual information in thousands of locations on road networks. A previous step in preventing or detecting accidents involves identifying vehicles on the road. The application of convolutional neural networks in object detection has significantly improved this field, enhancing classical computer vision techniques. Although, there are deficiencies due to the low detection rate provided by the available pre-trained models, especially for small objects. The main drawback is that they require manual labeling of the vehicles that appear in the images from each IP camera located on the road network to retrain the model. This task is not feasible if we have thousands of cameras distributed across the extensive road network of each nation or state. Our proposal presented a new automatic procedure for detecting small-scale objects in traffic sequences. In the first stage, vehicle patterns detected from a set of frames are generated automatically through an offline process, using super-resolution techniques and pre-trained object detection networks. Subsequently, the object detection model is retrained with the previously obtained data, adapting it to the analyzed scene. Finally, already online and in real-time, the retrained model is used in the rest of the traffic sequence or the video stream generated by the camera. This framework has been successfully tested on the NGSIM and the GRAM datasets.

1. Introduction

In this new era where big data is not the future but the present, any essential area, such as agriculture, transportation, or industry, is driven by data. Therefore, analyzing it can help to improve and optimize each discipline's internal processes. In the field of transportation, the road network of each nation or state has installed thousands of IP cameras in many locations to monitor the condition of the roads and prevent traffic jams and accidents or count traffic density at different times of the day. This information provided by these video surveillance cameras can be helpful for detecting anomalous events or accidents on the roads. But for this, it is critical to detect correctly the vehicles that circulate in them, a task that has had a significant impact on the irruption of deep learning-based object detection algorithms [1]. Intelligent video surveillance applied to traffic analysis has been a recurring research topic in recent years, with a multitude of works related to the detection of vehicles [2,3], tracking of objects along with the scene [4] or the modeling of their behavior [5]. Deep learning techniques have rekindled the interest and development of new proposals in the area.

The change from recognizing which object an image shows (a problem known as image classification) [6–9] to detecting several objects within the image, their classes, and their positions (a problem known as object detection), is a qualitative leap that has led to a revolution in the Computer Vision field. Many proposals can benefit from identifying objects, including these methods as a key part [10,11]. Both problems (image classification and object detection) performances have dramatically improved as Deep Convolutional Neural Networks (DCNN) use has become widespread. Therefore, DCNN-based object detection methods have been a field of intense exploration in recent years [12,13]. Their main challenges are the classes to be detected, processing time, computational power, and reliability when applied under non-ideal conditions.

* Corresponding author at: Department of Computer Languages and Computer Science, University of Málaga, Bulevar Louis Pasteur, 35, Málaga 29071, Spain.
E-mail addresses: ivangarcia@lcc.uma.es (I. García-Aguilar), jorgegarcia@lcc.uma.es (J. García-González), rmluque@lcc.uma.es (R.M. Luque-Baena), ezeqlr@lcc.uma.es (E. López-Rubio).

https://doi.org/10.1016/j.patrec.2023.01.015
0167-8655/© 2023 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/)

The detectable classes set with reasonable performance are typically conditioned on the existence of a labeled dataset large enough to perform the supervised training requiring the previous labeling by human intervention. Training the network successfully demands time and computing power, so obtaining a network trained from scratch to a specific domain is not trivial. Those reasons motivate to work from a pre-trained network trained with some of the most common datasets (e.g. Common Objects in Context, also known as COCO [14]) and perform fine-tuning with a domain-specific smaller dataset to adapt the network to the domain of application.

Although the computational power required for inference is less than for training, using dedicated graphics processing units (GPUs) is often essential to apply the method reasonably. Some methods propose detection algorithms focused on minimizing computing time and thus speeding up detection [15,16]. However, their use in applications that require high response speed (at least 15 frames per second) still requires a considerable amount of computing power. Although there is an intense line of research in adapting this kind of method to work using lower power and cost hardware [17,18], there is currently no general solution to trivialize the computation time required to apply object detection to an image.

Creating and adapting DCNN-based object detection methods to deal with conditions far from ideal is also an intense research field. One of the most typical challenges is to use object detection with a long distance between the camera and objects. This problem is known as Small Object Detection. Its importance is demonstrated by the number of recent approaches applied to solve it: methods based on Generative Adversarial networks [19], methods that adapt the operation of existing techniques [20], or studies about how the loss functions affect this issue [21]. Most approaches to increasing effectiveness when detecting small objects involve increasing processing time. Fatih et al. propose a library known as SAHI (Slicing Aided Hyper Inference) [22]. This generic solution is based on slicing-aided inference, maintaining lower complexity and memory requirements improve.

Our proposal uses a fine network adjustment (it does not increase the execution time) that allows it to adapt automatically to the traffic scene without human intervention. Firstly, we propose to apply a super-resolution algorithm to detect objects in the scene that would otherwise go unnoticed by the DCNN object detection method. These detected objects or labeled data are used to generate a training dataset to tune the network. All this process is done offline and only once per scene. The contributions provided by the proposed methodology are based on the automatic adaptation and training of a sequence, avoiding a previously labeled dataset since it is generated automatically. Without modifying the network, it is possible to improve the average precision rate obtained by the model significantly.

The following sections are structured as follows: Section 2 shows the proposed methodology, Section 3 on page 4 explains the experiments supporting our proposal, and Section 4 on page 7 explains our conclusions.

2. Methodology

The proposed methodology for improving the performance of object detections by deep convolutional neural networks is detailed next. Our proposal is composed of two subsystems. First, the output of an original deep convolutional neural network for object detection is refined by super-resolution to obtain high-quality detections, thus generating a labeled dataset automatically. This subsystem is described in Section 2.1. Then, the high-quality detections are employed to build a new training set for the object detection deep network to fine-tune the model, which improves its performance, as explained in Section 2.2.

2.1. Super-resolution enhancement of object detection and re-inference

This subsection describes our procedure to improve the quality of the object detection of a deep convolutional neural network. Our proposal begins with an unannotated video sequence, converted into a set of frames that will be processed in the next step.

$$\mathbf{D} = \{(\mathbf{Y}_l) \mid l \in \{1, \ldots, N\}\} \tag{1}$$

D is the unlabeled dataset, and $\mathbf{Y}_l$ is each low-resolution frame extracted. N stands for the total number of frames that compose the sequence. Our starting point is an original deep convolutional network $\mathcal{G}$ for object detection whose input is an image $\mathbf{Y}$ and yields a set of detections W:

$$W = \mathcal{G}(\mathbf{Y}) \tag{2}$$

$$W = \{(\alpha_i, \beta_i, \gamma_i, \delta_i, \lambda_i, \rho_i) \mid i \in \{1, \ldots, N\}\} \tag{3}$$

where N stands for the number of detections, the coordinates of the top left corner of the ith detection within $\mathbf{Y}$ are noted $(\alpha_i, \beta_i) \in \mathbb{R}^2$, the coordinates of the bottom right corner of the ith detection within $\mathbf{Y}$ are noted $(\gamma_i, \delta_i) \in \mathbb{R}^2$, λ_i stands for the detected class label, and $\rho_i \in \mathbb{R}$ denotes the obtained class score. The larger ρ_i, the more confident that we are that there is actually an object of class λ_i at that detection. The origin of coordinates is assumed to be at the center of the image in all cases.

The second task to be performed in our procedure is to process the incoming low-resolution input image $\mathbf{Y}_{LR}$ with the object detection deep network to produce a set of tentative detections W_{LR}, as indicated in step two of workflow Fig. 1:

$$W_{LR} = \mathcal{G}(\mathbf{Y}_{LR}) \tag{4}$$

In step three, several tentative regions are defined to set the areas on which to re-infer subsequently. In the worst scenario, the object detection model would not detect any initial element. Therefore, five fixed zones denoted as $\mathbf{F_r}$ have been defined. These regions cover the top, bottom corners, and center.

$$\mathbf{F_r} = \{(\chi_i, \psi_i) \mid i \in \{1, \ldots, K\}\} \tag{5}$$

K stands for the number of fixed regions selected. χ and ψ are the x and y coordinates of the centroid. Thanks to these settled regions, we ensure that our proposal covers all areas of the input image.

In step four, a high-resolution version $\mathbf{Y}_{HR}$ of the low resolution original image $\mathbf{Y}_{LR}$ with upscaling factor Z is obtained by a suitable super-resolution deep network.

$$\mathbf{Y}_{HR} = \mathcal{F}(\mathbf{Y}_{LR}) \tag{6}$$

$\mathcal{F}$ is the applied super-resolution model, which receives as input $\mathbf{Y}_{LR}$, a low-resolution image and processes it to obtain $\mathbf{Y}_{LR}$, an image with a higher number of pixels, thus increasing its quality. Based on our proposal, we have selected the Fast Super-Resolution Convolutional Neural Network model (FSRCNN) [23] as one of the fastest models to apply Super-resolution. This model has several versions available for use. Each determines a particular Z upscaling factor ($X2$, $X3$, $X4$). Since the selected upscaling factor represented by the variable Z is $X2$, the low-resolution image $\mathbf{Y}_{LR}$ will be processed, obtaining a new one denoted as $\mathbf{Y}_{HR}$ with the double of pixels. The proposal submitted can be executed by modifying Z. However, it is necessary to identify the context of the scene. With very high Z upscaling factors, the object detection model could incorrectly infer the class due to the increased size of the element.

Next, the list of candidate regions $\mathbf{C}$ is set according to the centroid of each detected element initially W_{LR}, denoted as $\mathbf{C}_{LR}$, together with the previously determined fixed regions $\mathbf{F_r}$. Based on

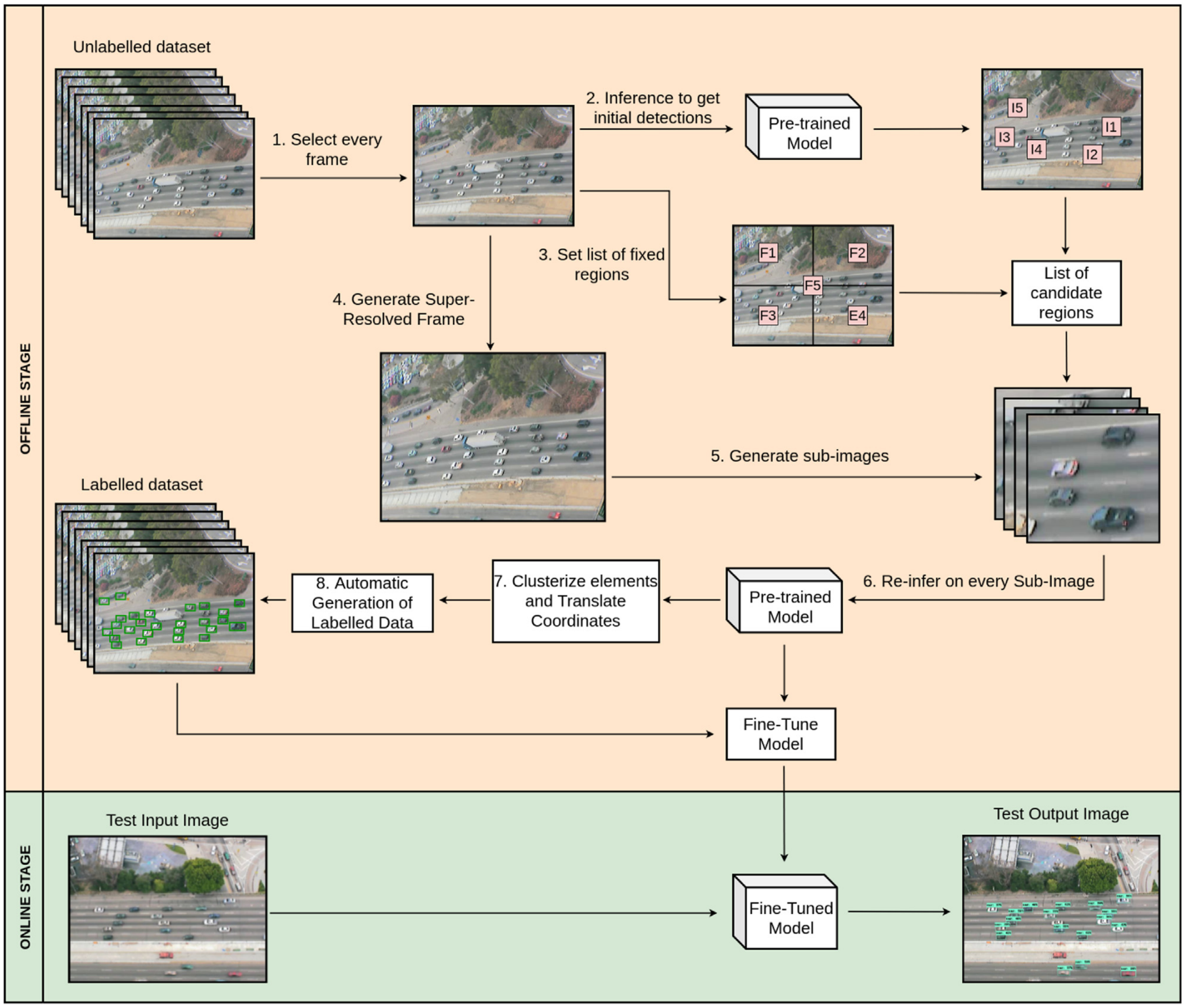
OFFLINE STAGE
Unlabelled dataset
1. Select every frame
2. Inference to get initial detections
Pre-trained Model
I5
I3
I4
I2
I1
3. Set list of fixed regions
F1
F2
F5
F3
E4
List of candidate regions
4. Generate Super-Resolved Frame
5. Generate sub-images
Labelled dataset
6. Re-infer on every Sub-Image
Pre-trained Model
7. Clusterize elements and Translate Coordinates
8. Automatic Generation of Labelled Data
Fine-Tune Model
ONLINE STAGE
Test Input Image
Fine-Tuned Model
Test Output Image

Fig. 1. Workflow of the proposed technique. The Offline part is composed of the SR application and the generation of the dataset for Fine-Tuning. The Online part is composed of the detection performed by the retrained model.

these regions, sub-images are generated to re-infer from the super-resolved image $\mathbf{Y}_{HR}$.

$$\mathbf{C}_{LR} = \{(\chi_i, \psi_i) \mid i \in \{1, \dots, N\}\} \tag{7}$$

$$(\chi_i, \psi_i) = \left(\frac{\alpha_i + \gamma_i}{2}, \frac{\beta_i + \delta_i}{2}\right) \tag{8}$$

where χ_i and ψ_i represent the x and y coordinates of the centroid obtained for each of the initially detected elements of W_{LR}.

$$\mathbf{C} = \mathbf{C}_{LR} \cup \mathbf{F_r} \tag{9}$$

The center of each generated sub-image $\mathbf{Y}_i$ matches each pair of translated coordinates denoted as $\hat{\mathbf{C}}_i$. The center of $\mathbf{Y}_i$ expressed in coordinates of $\mathbf{Y}_{LR}$ is noted as $\mathbf{C}_i$, while the center of $\mathbf{Y}_i$ expressed in coordinates of $\mathbf{Y}_{HR}$ is denoted $\hat{\mathbf{C}}_i$ according to the upscaling factor Z applied when generating the super-resolved image.

$$\hat{\mathbf{C}}_i = Z \cdot \mathbf{C}_i \tag{10}$$

In step five, we extract from $\mathbf{Y}_{HR}$ an image window $\mathbf{Y}_i$ of the same size as $\mathbf{Y}_{LR}$ for each pair denoted as $\hat{\mathbf{C}}_i$. The next step is to apply the object detection network to set a new list of detections, leading to step six of the workflow:

$$W_{HR} = \mathcal{G}(\mathbf{Y}_i) \tag{11}$$

The next step is to translate the locations of each element in W_{HR} from the coordinate system of $\mathbf{Y}_i$ to the coordinate system of $\mathbf{Y}_{LR}$. The equation that translates a point $\hat{\mathbf{C}}_i$ expressed in the coordinate system of $\mathbf{Y}_i$ to the same point $\mathbf{C}_i$ expressed in the coordinate system of $\mathbf{Y}_{LR}$ is the following:

$$\mathbf{C}_i = (\chi_i, \psi_i) + \frac{1}{Z} \cdot \hat{\mathbf{C}}_i \tag{12}$$

The list of elements with the translated coordinates is denoted as W_{HRT}.

$$W_{HRT} = \left\{\left(\alpha_{k,l}, \beta_{k,l}, \gamma_{k,l}, \delta_{k,l}, \lambda_{k,l}, \rho_{k,l}\right) \mid l \in \{1, \dots, N_k\}\right\} \tag{13}$$

$$\left(\alpha_{k,l}, \beta_{k,l}\right) = \mathbf{y}_k + \frac{1}{Z}\left(\hat{\alpha}_{k,l}, \hat{\beta}_{k,l}\right) \quad \left(\gamma_{k,l}, \delta_{k,l}\right) = \mathbf{y}_k + \frac{1}{Z}\left(\hat{\gamma}_{k,l}, \hat{\delta}_{k,l}\right) \tag{14}$$

$$\lambda_{k,l} = \hat{\lambda}_{k,l} \quad \rho_{k,l} = \hat{\rho}_{k,l} \tag{15}$$

Next, in step seven, the detections comprised in the lists of detections W_{HRT} are evaluated to determine whether they are associated with previously undetected objects or already correspond to a known detection in W_{LR}. This evaluation is done by computing the

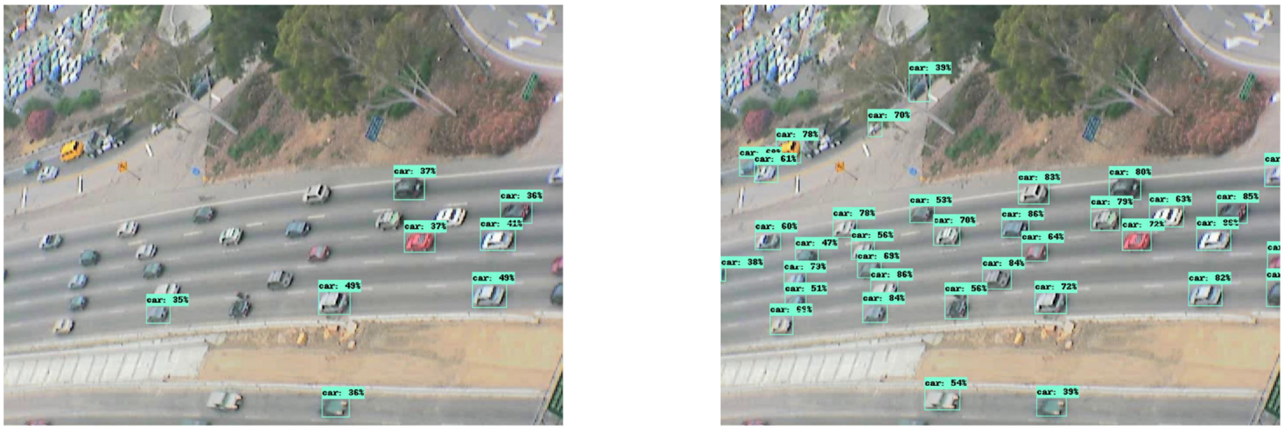

Fig. 2. An example applied to frame 91 of the first video, denoted as sb-camera1-0820am-0835am. The left side shows the results of the Fine Tune with the raw model, while the right side shows the detections after applying the Fine Tune with the outputs of the described technique using sequence one and the EfficientDet D4 model.

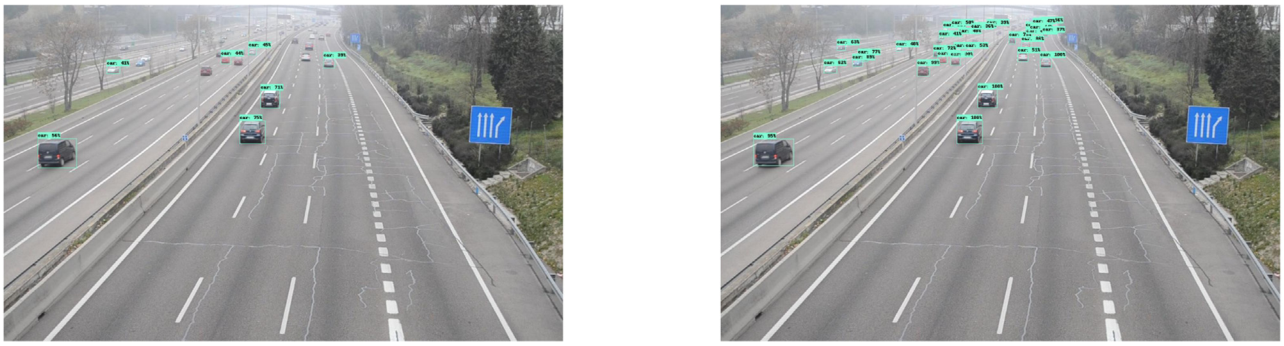

Fig. 3. An example applied to frame 401 of the M30-HD sequence [27]. The left side shows the results of the Fine Tune with the raw model, while the right side shows the detections after applying the Fine Tune with the outputs of the described technique using sequence three and the EfficientDet D4 model.

intersection over union metric for each pair of detections W_{HRT_j} and W_{HRT_k}:

$$IOU = \frac{Area\left(W_{HRT\_j} \cap W_{HRT\_k}\right)}{Area\left(W_{HRT\_j} \cup W_{HRT\_k}\right)} \qquad (16)$$

A pair of detections W_{HRT_j} and W_{HRT_k} are associated with the same object if and only if $IOU > \theta$, where the threshold θ must be adjusted. After that, the final set of detections W'_{HRT} is obtained by collecting the remaining after removing duplicates. The cluster κ forms a list with the detections obtained for each element. The detection with the highest score for each is selected.

2.2. Fine-tuning by automatically labeled training data

Subsequently, as detailed in step eight of the workflow, an automated procedure to generate a labeled training dataset for fine-tuning an object detection network is proposed. Once a set of high-quality detections W'_{HRT} is obtained from the subsystem described in Section 2.1, it is employed as the training set.

$$W'_{HRT} = \{(\alpha_i, \beta_i, \gamma_i, \delta_i, \lambda_i, \rho_i) \mid i \in \{1, \ldots, N\}\} \qquad (17)$$

Finally, the detections are converted to the required format to generate the files for Fine-Tuning the model. The motivation behind this is that the set of high-quality detections contains more detections with higher confidence than the set of detections obtained by the original object detection network W_{LR}. Therefore, fine-tuning the network with W'_{HRT} is expected to result in a higher accuracy of the fine-tuned object detection network, as seen in the image given as output by the model after being retrained; see Figs. 2 and 3 for more details, in which the qualitative results obtained with our proposal increases the number of elements detected and the average class score. A quantitative comparison of our approach has been performed; see Tables 1 and 2. The official implementation is publicly available in the repository.[1]

3. Experiments

Our objective is to determine the effectiveness based on the number of detections and the class score of the detected elements. For this purpose, sequences captured by video surveillance systems have been selected to apply our proposal. A comparison has been made between the following methods:

- Original Model: The direct application of the unmodified raw object detection model.
- Original Model + Fine-Tuning: Re-training the model with the outputs previously provided by the RAW model.
- SR Enhancement Only: Our previously proposed meta-method [24].
- Proposed: The proposed technique is based on re-training the model, using the automatically labeled dataset provided by the enhancement using Super-resolution, see Fig. 1 for more details.
- SAHI: Slicing Aided Hyper Inference [22]. This proposal is designed for the Yolo and FRCNN Convolutional Neural Network.

In this section, a comparative analysis is performed according to the accuracy obtained between the direct application of the model and our methodology presented. The main difference between the Original Model + Fine Tuning and our proposal is mainly

[1] https://github.com/IvanGarcia7/ALAF

Table 1
The total number of elements in the sequence, average detections, normalized average class score, population standard deviation, and average sub-images are processed per frame by the different techniques proposed using the EfficientDet D4 model. The best results are marked in **bold**.

		EfficientDet D4 - class car - score > 40%			
		Total number elements	Average detections	Normalized average class score	Average sub-images Processed per frame
Seq. 1 - NGSIM sb-camera1-0820am-0835am	Ground Truth	6292	–	–	–
	Original Model	398	2.503 ± 1.466	3.142 ± 1.975	1
	SR Enhancement Only	4087	23.090 ± 5.214	35.829 ± 6.855	35
	Original Model + Fine-Tuning	4947	27.949 ± 3.107	51.406 ± 4.498	1
	Proposed	**6011**	**33.960 ± 3.496**	**66.310 ± 3.321**	1
	YOLOv5 [28] + SAHI [29]	1453	8.256 ± 3.359	12.125 ± 5.207	9
	FRCNN [30] + SAHI [29]	2325	13.136 ± 4.625	23.232 ± 8.095	9
Seq. 2 - NGSIM sb-camera2-0820am-0835am	Ground Truth	4438	–	–	–
	Original Model	632	4.104 ± 2.184	7.075 ± 4.480	1
	SR Enhancement Only	2968	19.026 ± 4.891	40.826 ± 8.400	28
	Original Model + Fine-Tuning	2833	18.160 ± 4.209	39.517 ± 6.844	1
	Proposed	**4142**	**26.551 ± 4.799**	**71.051 ± 5.030**	1
	YOLOv5 [28] + SAHI [29]	887	5.836 ± 2.905	11.030 ± 6.075	9
	FRCNN [30] + SAHI [29]	1129	7.237 ± 2.867	16.624 ± 6.118	9
Seq. 3 - NGSIM sb-camera3-0750am-0805am	Ground Truth	3810	–	–	–
	Original Model	614	4.234 ± 2.216	7.613 ± 4.305	1
	SR Enhancement Only	1808	12.053 ± 4.161	25.817 ± 8.546	25
	Original Model + Fine-Tuning	1787	11.913 ± 3.192	25.697 ± 7.193	1
	Proposed	**2944**	**19.627 ± 3.509**	**48.510 ± 5.649**	1
	YOLOv5 [28] + SAHI [29]	610	4.357 ± 2.364	8.638 ± 4.688	9
	FRCNN [30] + SAHI [29]	2017	13.447 ± 4.647	32.454 ± 10.925	9
Seq. 4 - NGSIM sb-camera4-0820am-0835am	Ground Truth	4803	–	–	–
	Original Model	203	1.990 ± 1.080	2.886 ± 1.819	1
	SR Enhancement Only	1632	10.953 ± 4.007	19.426 ± 6.984	20
	Original Model + Fine-Tuning	1077	7.086 ± 2.409	11.443 ± 4.151	1
	Proposed	**2558**	**16.829 ± 3.572**	**30.940 ± 6.066**	1
	YOLOv5 [28] + SAHI [29]	147	1.564 ± 0.929	2.558 ± 1.383	9
	FRCNN [30] + SAHI [29]	1869	12.296 ± 4.256	25.042 ± 7.374	9
GRAM-M30HD	Ground Truth	3892	–	–	–
	Original Model	540	5.870 ± 2.983	9.640 ± 5.317	1
	SR Enhancement Only	1262	13.570 ± 4.519	22.662 ± 7.235	21
	Original Model + Fine-Tuning	1163	12.505 ± 3.996	25.744 ± 8.400	1
	Proposed	**2272**	**24.430 ± 5.260**	**43.243 ± 8.933**	1
	YOLOv5 [28] + SAHI [29]	1711	18.398 ± 4.192	31.228 ± 7.038	24
	FRCNN [30] + SAHI [29]	2592	27.871 ± 5.571	56.874 ± 9.600	24

based on the dataset used to perform the fine-tuning of the model. Since our proposal considers re-training, to be fair, when performing the comparison, we have exclusively applied the automatic re-training part described in Section 2.2 to the original model. Original Model + Fine Tuning uses the outputs provided by the direct application of the model (Raw) without implementing any additional technique. According to our proposal, it is first used in an offline phase to improve the number of elements detected and the quality of the detections. Therefore, a dataset will be generated fully automatically with a more significant number of elements that the model will learn about and that, at first, were not initially detected. See Fig. 1 for more details about the framework. In addition, a comparison is made between the first version, García-Aguilar et al. [24] and the present proposal. There are several differences, highlighting mainly the implementation of fixed regions to re-infer along the entire image, the inclusion and improvement of the clustering algorithm for coincident grouping elements, and the new phase of labeled data generation and re-training included in the methodology presented in Section 2. During experiments, an SGD optimizer with a learning rate between 0.002 and 0.005, a momentum of 0.9, and a linear warmup of 500 iterations are used. Additionally, the competitor known as SAHI is considered.

3.1. Pre-trained model

Even though our proposal can be applied using any neural DCNN-based object detection model to carry out our experiments, *EfficientDet D4* [25] was selected as a pre-trained model. It is the most suitable one according to the size of the images given as input and its accuracy in detecting initial elements of the road on which the vehicles circulate. This pre-trained model has been obtained from the Tensorflow Model Zoo repository.[2] These models have been trained with the COCO dataset (Common Objects in Context) [26]. The COCO dataset comprises a wide range of challenging realistic images, consisting of various disorganized scenes. These scenes highlight mainly for having a high diversity, showing objects of diverse scale and overlapping on certain occasions. These facts allow training object detection models in realistic environments. It is widely used in several machine learning projects, playing an essential task in Deep Learning applications, such as object detection.

Based on the selected video sequences (Section 3.2), when performing the fine-tuning of the pre-trained model, the number of classes has been restricted from the initial 90 classes in the COCO dataset [26] to 4 (car, truck, motorcycle, and bus) since they are the most probable classes in traffic scenes. The car class was selected for the experimentation phase since the number of vehicles belonging to this class was higher in the selected video sequences.

3.2. Video sequences

Four video sequences from *US Highway 101 (NGSIM) Dataset*[3] and an additional sequence named M30-HD from the GRAM

[2] https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md

[3] https://www.fhwa.dot.gov/publications/research/operations/07030/index.cfm

Table 2
Mean average precision (mAP) for the five tested sequences (higher is better). The best results are marked in **bold**. Fine-Tuning has been performed using the outputs provided by the EfficientDet D4 model.

		Mean average precision (mAP) - EfficientDet D4 - score > 35%				
		IoU = 0.50:0.95| area = all	IoU > 0.50| area = all	IoU > 0.75| area = all	IoU = 0.50:0.95| area = Small	IoU > 0.50| area = Medium
Seq. 1 - NGSIM sb-camera1-0820am-0835am	Original Model	0.145	0.188	0.187	0.137	0.218
	SR Enhancement Only	0.470	0.697	0.588	0.468	0.484
	Original Model + Fine-Tuning	0.555	0.821	0.732	0.562	0.490
	Proposed	**0.643**	**0.944**	**0.879**	**0.647**	**0.608**
	YOLOv5 [28] + SAHI [29]	0.181	0.247	0.232	0.180	0.229
	FRCNN [30] + SAHI [29]	0.155	0.237	0.188	0.158	0.176
Seq. 2 - NGSIM sb-camera2-0820am-0835am	Original Model	0.214	0.277	0.275	0.189	0.288
	SR Enhancement Only	0.519	0.721	0.669	0.510	0.541
	Original Model + Fine-Tuning	0.476	0.683	0.622	0.451	0.546
	Proposed	**0.647**	**0.926**	**0.875**	**0.637**	**0.682**
	YOLOv5 [28] + SAHI [29]	0.171	0.237	0.221	0.172	0.168
	FRCNN [30] + SAHI [29]	0.139	0.211	0.176	0.146	0.126
Seq. 3 - NGSIM sb-camera3-0750am-0805am	Original Model	0.196	0.305	0.239	0.199	0.139
	SR Enhancement Only	0.333	0.546	0.384	0.342	0.199
	Original Model + Fine-Tuning	0.326	0.554	0.360	0.335	0.204
	Proposed	**0.524**	**0.818**	**0.679**	**0.535**	**0.406**
	YOLOv5 [28] + SAHI [29]	0.109	0.177	0.124	0.111	0.050
	FRCNN [30] + SAHI [29]	0.125	0.219	0.134	0.131	0.032
Seq. 4 - NGSIM sb-camera4-0820am-0835am	Original Model	0.072	0.109	0.092	0.079	0.033
	SR Enhancement Only	0.251	0.395	0.301	0.264	0.149
	Original Model + Fine-Tuning	0.169	0.297	0.181	0.172	0.136
	Proposed	**0.382**	**0.592**	**0.501**	**0.401**	**0.241**
	YOLOv5 [28] + SAHI [29]	0.025	0.039	0.028	0.026	0.000
	FRCNN [30] + SAHI [29]	0.057	0.086	0.069	0.063	0.024
GRAM-M30HD	Original Model	0.125	0.168	0.146	0.049	0.724
	SR Enhancement Only	0.218	0.355	0.229	0.152	0.744
	Original Model + Fine-Tuning	0.186	0.305	0.197	0.112	**0.756**
	Proposed	**0.260**	**0.510**	**0.241**	**0.191**	0.746
	YOLOv5 [28] + SAHI [29]	0.230	0.416	0.228	0.169	0.685
	FRCNN [30] + SAHI [29]	0.240	0.503	0.208	0.185	0.664

dataset [27] have been selected to test our proposal. *NGSIM* videos have been captured by highway video surveillance systems from a high perspective and belong to the U.S. Department of Transportation. These systems collect a series of sequences with many small vehicles. It has a total duration of about 15 min. Images for validation have been annotated (632 manually labeled images with a total of 19,343 vehicles)[4] to evaluate our proposal by obtaining the Mean Average Precision (*mAP*) once the fine-tuning has been applied. An appropriate time margin has been allowed to ensure the same vehicle from the evaluation set does not appear in the training set. The training dataset is generated from the following frames by applying our proposal. In addition, 3892 elements have been manually labeled for the GRAM sequence M30-HD to perform the evaluation.

3.3. Evaluation

To compare the effectiveness of the presented proposal from a quantitative point of view, the number of elements that compose the selected sequences has been determined, as well as the average number of detections per frame and the normalized class score obtained for them. This process has been performed for the five chosen video sequences in Table 1.

The evaluation of COCO[5] has been considered. This evaluator is widely used to evaluate the performance obtained by the selected model. The metric studied in this section is the mean average precision (mAP). It is complex because it is calculated in several steps. First, the coincidence of the detections obtained is matched to the Ground Truth (GT) according to the degree of IoU. After calculating the true positives, as well as false positives and negatives, a precision-recall curve is generated. Finally, the accuracy of each class is calculated, and it's mean is computed. This evaluator performs a much more complex analysis since it calculates the mAP for 10 IoU thresholds, covering a range from 0.5 to 0.95 with a step size of 0.05. In evaluating a model, generating metrics for several thresholds is more accurate because calculating only one of them makes it possible to induce a bias in the evaluation metric, becoming indulgent with it. Table 2 on page 6 shows a comparison of the mean average precision (mAP) obtained for sequences 1, 2, 3, 4, and M30-HD. The results obtained by each proposal are shown. The tests have been performed using an Nvidia Geforce RTX 3080 Ti.

3.4. Results

According to Table 1, we can affirm that our proposal detects a more significant number of elements. As can be seen, the number of average detections identified for each frame is higher. For example, we have obtained an average of ≈ 34 vehicles for sequence one, compared to the ≈ 3 detected by the original model. The dataset used to apply our proposal contains elements that a priori were not identified. This is why, when fine-tuning the model, it learns from them and can identify them later. Using sequence number two as an example, we can determine that the number of elements detected by our proposal is close to the number of elements defined by the GT.

Under the established proposal, the normalized class score is much higher, obtaining average reliability of 66.31% in the de-

[4] https://github.com/IvanGarcia7/NGSIM-Dataset-Annotations

[5] https://github.com/cocodataset/cocoapi/blob/master/PythonAPI/pycocotools/cocoeval.py

tected elements for sequence one with our proposal compared to 3.142% obtained by the original model. This is also the case for the other sequences in which the experimentation phase has been performed.

Another aspect to consider is the inference time required by each compared technique. In Table 1, *SR enhancement only* based on the exclusive application of super-resolution is not valid in scenarios where real-time element detection is required because it needs to re-infer on the several sub-images generated. For this reason, this considerably increases the time required when processing a single frame. For example, sequence one, shown in Table 1, known as *SR Enhancement Only*, requires re-infer on 35 sub-images created from the input frame. Therefore, it is possible to affirm that applying our proposal once the offline process has been performed does not increase the time required to process a frame since higher accuracy precision rates are achieved by inferring only once on the input image.

Applying the COCO evaluator, Table 2 is obtained. We can determine that the metrics obtained by our proposal outperform the rest. Using sequence number two as an example, we go from an average mAP of 27.7% by the raw model to 92.6% obtained after applying our proposal. This happens in the rest of the sequences. It should be highlighted that the application of our proposal not only improves the detections in small objects since the rest of the fields also increase the accuracy obtained.

Qualitative examples of one of the frames of sequences 1, 3, and M30-HD are used to perform the experimental phase and are shown in Figs. 2 and 3. The number of elements detected in the image is higher. In addition, the normalized class inference established for each element also increases.

4. Conclusions

This work proposes a methodology to automatically adapt any object detection model, without modifying its architecture, to a specific scene with small objects with no human intervention. An offline super-resolution strategy is first applied to find detection examples from the scene, even if the object detection method cannot detect them from the original image. The new examples are then used as training data to perform a fine-tuning process from a general-purpose pre-trained object detection method. In the online phase, the model only needs to infer once on the input image compared to other techniques. It is recommended in sequences captured by video-surveillance systems.

To test the proposal, experiments with five video sequences have been done to test each method section's utility. The results support our proposal by showing an outstanding improvement in Mean Average Precision (mAP) values when applied with no associated processing time increase since, after being re-trained, it only requires inferring on the frame once.

Studies with other CNN-based object detection methods could be performed in future work. Since the initial data collection method based on SR enhancement is offline, an ensemble of object detection methods could be used to get a better fine-tuning training dataset.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work is partially supported by the Autonomous Government of Andalusia (Spain) under project UMA20-FEDERJA-108, project name Detection, characterization and prognosis value of the non-obstructive coronary disease with deep learning. It includes funds from the European Regional Development Fund (ERDF). It is also partially supported by the University of Málaga (Spain) under grants B1-2019_01, project name *Anomaly detection on roads by moving cameras;* B1-2019_02, project name *Self-Organizing Neural Systems for Non-Stationary Environments;* and B1-2021_20, project name *Detection of coronary stenosis using deep learning applied to coronary angiography.* Iván García-Aguilar is funded by a scholarship from the Autonomous Government of Andalusia (Spain) under the Young Employment operative program [grant number SNGJ5Y6-15]. The authors thankfully acknowledge the computer resources, technical expertise and assistance provided by the SCBI (Supercomputing and Bioinformatics) center of the University of Málaga. They also gratefully acknowledge the support of NVIDIA Corporation with the donation of two Titan X GPUs. The authors also thankfully acknowledge the grant of the Universidad de Málaga and the Instituto de Investigación Biomédica de Málaga y Plataforma en Nanomedicina-IBIMA Plataforma BIONAND. The authors have declared no conflict of interest. Funding for open access charge: Universidad de Málaga/CBUA.

References

[1] Z.-Q. Zhao, P. Zheng, S.-T. Xu, X. Wu, Object detection with deep learning: a review, IEEE Trans. Neural Netw. Learn. Syst. 30 (11) (2019) 3212–3232, doi:10.1109/TNNLS.2018.2876865.
[2] M. Molina-Cabello, R. Luque-Baena, E. López-Rubio, K. Thurnhofer-Hemsi, Vehicle type detection by ensembles of convolutional neural networks operating on super resolved images, Integr. Computer-Aided Eng. 25 (4) (2018) 321–333, doi:10.3233/ICA-180577.
[3] R. Luque, E. Domínguez, E. Palomo, J. Muñoz, A Neural Network Approach for Video Object Segmentation in Traffic Surveillance, in: Lecture Notes in Computer Science, vol. 5112, LNCS, 2008, pp. 151–158.
[4] S. Sivaraman, M. Trivedi, Looking at vehicles on the road: a survey of vision-based vehicle detection, tracking, and behavior analysis, IEEE Trans. Intell. Transp. Syst. 14 (4) (2013) 1773–1795, doi:10.1109/TITS.2013.2266661.
[5] W. Hu, T. Tan, L. Wang, S. Maybank, A survey on visual surveillance of object motion and behaviors, IEEE Trans. Syst., Man, Cybern. Part C 34 (3) (2004) 334–352.
[6] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, Commun. ACM 60 (6) (2017) 84–90, doi:10.1145/3065386.
[7] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, in: Y. Bengio, Y. LeCun (Eds.), 3rd International Conference on Learning Representations, ICLR 2015, 2015.
[8] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
[9] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 1–9, doi:10.1109/CVPR.2015.7298594.
[10] G. Lan, J. Benito-Picazo, D. Roijers, E. Domínguez, A. Eiben, Real-time robot vision on low-performance computing hardware, 2018. doi:10.1109/ICARCV.2018.8581288.
[11] J. García-González, J.M. Ortiz-de-Lazcano-Lobato, R.M. Luque-Baena, E. López-Rubio, Foreground detection by probabilistic mixture models using semantic information from deep networks, in: 24th European Conference on Artificial Intelligence, ECAI, vol. 325, 2020, pp. 2696–2703, doi:10.3233/FAIA200408.
[12] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014.
[13] R. Girshick, Fast R-CNN, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2015.
[14] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, C.L. Zitnick, Microsoft COCO: common objects in context, in: Computer Vision – ECCV 2014, Springer International Publishing, Cham, 2014, pp. 740–755.
[15] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, A.C. Berg, SSD: Single Shot Multibox Detector, in: Lecture Notes in Computer Science, 2016, pp. 21–37, doi:10.1007/978-3-319-46448-0_2.

[16] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, real-time object detection, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779–788, doi:10.1109/CVPR.2016.91.

[17] Y. Lee, J.-w. Hwang, S. Lee, Y. Bae, J. Park, An energy and GPU-computation efficient backbone network for real-time object detection, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2019, pp. 752–760.

[18] J. Benito-Picazo, E. Domínguez, E. Palomo, E. López-Rubio, Deep learning-based video surveillance system managed by low cost hardware and panoramic cameras, Integr Computer-Aided Eng. 27 (2020) 1–15, doi:10.3233/ICA-200632.

[19] J. Rabbi, N. Ray, M. Schubert, S. Chowdhury, D. Chao, Small-object detection in remote sensing images with end-to-end edge-enhanced GAN and object detector network, Remote Sens. 12 (9) (2020), doi:10.3390/rs12091432.

[20] G. Cao, X. Xie, W. Yang, Q. Liao, G. Shi, J. Wu, Feature-fused SSD: fast detection for small objects, in: Ninth International Conference on Graphic and Image Processing (ICGIP 2017), vol. 10615, 2018, pp. 381–388, doi:10.1117/12.2304811.

[21] X. Yang, J. Yan, Q. Ming, W. Wang, X. Zhang, Q. Tian, Rethinking rotated object detection with gaussian Wasserstein distance loss, 2021. arXiv:2101.11952.

[22] F.C. Akyon, S.O. Altinuc, A. Temizel, Slicing aided hyper inference and fine-tuning for small object detection, 2022. doi:10.48550/ARXIV.2202.06934.

[23] C. Dong, C.C. Loy, X. Tang, Accelerating the super-resolution convolutional neural network, in: B. Leibe, J. Matas, N. Sebe, M. Welling (Eds.), Computer Vision – ECCV 2016, Springer International Publishing, Cham, 2016, pp. 391–407.

[24] I. García-Aguilar, R.M. Luque-Baena, E. López-Rubio, Improved detection of small objects in road network sequences using CNN and super resolution, Expert Syst. 39 (2) (2021), doi:10.1111/exsy.12930.

[25] M. Tan, R. Pang, Q.V. Le, EfficientDet: scalable and efficient object detection, 2020. arXiv:1911.09070.

[26] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C.L. Zitnick, P. Dollár, Microsoft COCO: common objects in context, 2015, arXiv:1405.0312.

[27] R. Guerrero-Gomez-Olmedo, R.J. Lopez-Sastre, S. Maldonado-Bascon, A. Fernandez-Caballero, Vehicle tracking by simultaneous detection and viewpoint estimation, in: IWINAC 2013, Part II, LNCS 7931, 2013, pp. 306–316.

[28] G. Jocher, A. Chaurasia, A. Stoken, J. Borovec, NanoCode012, Y. Kwon, K. Michael, TaoXie, J. Fang, Imyhxy, Lorna, Z. Yifu, C. Wong, Abhiram V, D. Montes, Z. Wang, C. Fati, J. Nadar, Laughing, UnglvKitDe, V. Sonck, Tkianai, YxNONG, P. Skalski, A. Hogan, D. Nair, M. Strobel, M. Jain, ultralytics/yolov5: v6.1 - tensorrt, tensorflow edge TPU and openvino export and inference, 2022, doi:10.5281/ZENODO.3908559.

[29] F.C. Akyon, S.O. Altinuc, A. Temizel, Slicing aided hyper inference and fine-tuning for small object detection, arXiv preprint arXiv:2202.06934 (2022).

[30] S. Ren, K. He, R. Girshick, J. Sun, Faster R-CNN: towards real-time object detection with region proposal networks, 2015. doi:10.48550/ARXIV.1506.01497.