

Real-time on-board pedestrian detection using generic single-stage algorithms and on-road databases

*International Journal of Advanced
Robotic Systems*
September-October 2020: 1–10
© The Author(s) 2020
Article reuse guidelines:
sagepub.com/journals-permissions
DOI: 10.1177/1729881420929175
journals.sagepub.com/home/arx



Vicent Ortiz Castelló¹, Omar del Tejo Catalá¹,
Ismael Salvador Igual¹  and Juan-Carlos Perez-Cortes^{1,2}

Abstract

Pedestrian detection is a particular case of object detection that helps to reduce accidents in advanced driver-assistance systems and autonomous vehicles. It is not an easy task because of the variability of the objects and the time constraints. A performance comparison of object detection methods, including both GPU and non-GPU implementations over a variety of on-road specific databases, is provided. Computer vision multi-class object detection can be integrated on sensor fusion modules where recall is preferred over precision. For this reason, ad hoc training with a single class for pedestrians has been performed and we achieved a significant increase in recall. Experiments have been carried out on several architectures and a special effort has been devoted to achieve a feasible computational time for a real-time system. Finally, an analysis of the input image size allows to fine-tune the model and get better results with practical costs.

Keywords

Object detection, artificial intelligence, machine learning, convolutional neural networks, resource-constrained hardware, one-stage detectors, advanced driver-assistance systems, vulnerable road users

Date received: 28 November 2019; accepted: 1 May 2020

Topic: Vision Systems

Topic Editor: Antonio Fernandez-Caballero

Associate Editor: Ke-Cai Cao

Introduction

Object detection is a central problem in Computer Vision. Its goal is to detect the location and class of each object in images or image sequences. Applications include person identification, video surveillance or autonomous car driving, among others. Regarding the last one, pedestrian detection constitutes one of the most challenging tasks to perform in terms of on-road object detection for two main reasons. First, pedestrians are the most vulnerable users of the road, with any accident potentially causing them major injuries, and even death. Second, their intrinsic variability (people of different shapes in different clothes, poses and light conditions) makes it especially difficult for vision-based recognition systems to detect them precisely and confidently. Therefore, despite recent large improvements

in accuracy, the pedestrian detection task still has several difficulties that require more dedication on design, optimisation and evaluation.¹

Over the last 15 years, many tries to reach a strong pedestrian detector have been carried out. First attempts

¹Instituto Tecnológico de Informática (ITI), Universitat Politècnica de València, Valencia, Spain

²Departamento de Informática de Sistemas y Computadores (DISCA), Universitat Politècnica de València, Valencia, Spain

Corresponding author:

Ismael Salvador Igual, Instituto Tecnológico de Informática (ITI), Universitat Politècnica de València, Camino de Vera, s/n, 46022 Valencia, Spain.

Email: issalig@iti.es



Creative Commons CC BY: This article is distributed under the terms of the Creative Commons Attribution 4.0 License (<https://creativecommons.org/licenses/by/4.0/>) which permits any use, reproduction and distribution of the work without

further permission provided the original work is attributed as specified on the SAGE and Open Access pages (<https://us.sagepub.com/en-us/nam/open-access-at-sage>).

included the use of traditional algorithms for this purpose, such as AdaBoost and Cascade based detection structures, known as Viola–Jones² or Histogram of Oriented Gradients plus Support Vector Machine structures,³ with very poor results, especially when considering small, occluded or cropped pedestrians. Later, more refined, ad hoc algorithms were proposed: Deformable Part Model,⁴ Integral Channel Features,⁵ Locally Decorrelated Channel Features⁶ or Fast Feature Pyramids,⁷ with better results. Nevertheless, they still rely on carefully designed features. Finally, over the last few years, major improvements have been achieved in terms of quality and confidence in pedestrian detection,⁸ many of them based on Machine Learning (ML) techniques, mainly Neural Networks, such as Rich Feature Hierarchies (R-CNN),⁹ Deep Parts,¹⁰ Faster R-CNN,¹¹ Single Shot Multibox Detector (SSD)¹² or You Only Look Once (YOLO).^{13–15}

However, not only there are variability constraints that make it hard to overcome human-like performance at detecting pedestrians in an automotive task, but also high computational power demands that Convolutional Neural Networks (CNN) pose. Because of that, some methods put special efforts into getting fast detectors (throughput), while others are focused on achieving state-of-the-art results in terms of detection quality (precision and recall, among other metrics).

These detectors can be classified as two-stage and one-stage methods. Two-stage algorithms predict detections in two phases: they use spatial features at pixel level to extract some Regions of Interest, and then use a second phase to classify all the proposals to decide if each region is a pedestrian or not. These methods usually produce better detection results, but they are more computationally expensive,¹⁶ thus being less used in real-time (RT) detection tasks because of the limited computational power of most of the resource-constrained devices usually installed on-board. For these reasons, one-stage methods are very common: they use a single phase to detect relevant objects of many aspect ratios at multiple scales in the image. One-stage detectors involve lighter algorithms that are much more suitable for the available on-board hardware, but traditionally offer slightly worse detection accuracies. However, thanks to the special effort put into them, their detection quality tends to get closer to their two-stage counterparts.¹⁵

Regarding studies on one-stage detectors for pedestrian detection, some articles can be found on literature but they tend to focus on one single database for INRIA^{17,18} and KITTI.¹⁹ In this article, we train and fine-tune YOLOv3 algorithm and apply it to multiple recent, on-road image databases, and consider a set of indicators to assess performance.

The article is organised as follows. First, the most commonly used databases for on-road tasks are presented. Then, a selection of the best one-stage detectors are described and a comparative analysis is performed. Later,

performance and throughput of the analysed detectors is reported for a variety of architectures also including resource-constrained hardware. Also, an ad hoc training with reduction of the number of categories has been carried out for YOLOv3, achieving a significant increase in recall. Also, an analysis of the input image size is performed allowing to fine-tune the model. Finally, conclusions and future work on the topic are described.

Databases

Although generic-class databases, such as COCO,²⁰ are the starting point to develop general detection algorithms, we focused our effort on specific on-road and human image databases. After extensive research on the literature of the last 10 years and according to the number of images, number of categories, dissemination and usefulness, we have selected a subset of specific on-road databases that comprises Caltech-USA,²¹ Daimler,^{22,23} EuroCity Persons (ECP)²⁴ and nuScenes.²⁵

First, Caltech is chosen for its generality and size, including a variety of environmental conditions, such as rain, fog or variations of lighting, and the fact that its images are continuous sequences with more than 1200 unique pedestrians.

Besides, Daimler is a well-known, established dataset focused on pedestrian detection consisting of black and white (B/W) equalised images from a long video sequence.

ECP dataset is one of the most diverse and large automotive person dataset, including data from the 4 seasons, 12 countries, 31 cities, with high pedestrian density. The dataset comprises day and night images, with different weather and adverse lighting conditions, and its focus is on vulnerable road users (VRUs).

Finally, nuScenes is a very novel, public large-scale dataset for autonomous driving. It includes data from the full sensor suite of a self-driving car (RADAR, LiDAR, cameras, IMU and GPS), with more than 1.4 million camera images, and it provides manually labelled annotations for 23 classes, including VRUs.

Therefore, the databases used are joined to obtain a complete dataset that tries to represent as much variability as possible including different image sizes and aspect ratios, weather conditions, cities and roads, and a wide range of light conditions (see Figure 1).

Algorithms

To test the databases selected for the experiments, we used some of the state-of-the-art generic detectors suitable for RT environments. These include SSD,¹² Mobilenets,²⁶ Spatial Pyramid Pooling YOLOv3,²⁷ YOLOv3 416 and YOLOv3 608.¹⁵ All these detectors have been trained on generic datasets including objects such as ‘person’. Therefore, we were able to use the pre-trained models directly to perform our experiments.

Redmon and Farhadi proposed YOLOv3¹⁵: a one-stage detector which leverages its fully convolutional structure to create a predictor for images of any shape. It divides an image into grids at three different scales creating three grids of sizes 13×13 , 26×26 and 52×52 (see Figure 4). Each cell of the grid is responsible for predicting up to three bounding boxes for objects whose centre pixel is within the cell. The bounding box format is

$$(x, y, w, h, c, C_1, C_2, \dots, C_n)$$

in which x and y are the shifts relative to the top-left corner of the cell; w and h are width and height of the bounding box relative to some preselected anchor boxes; c is the confidence YOLOv3 estimates for the detection and C_1, C_2, \dots, C_n are the confidences for each class. As YOLOv3 can handle any image shape, we chose the most widespread versions (YOLOv3 416 and YOLOv3 608) for our experiments, in which the main difference is the image sizes used, that is, $416 \times 416 \times 3$ and $608 \times 608 \times 3$. Although the network architecture is the same, the computational cost for YOLOv3 608 is higher than for YOLOv3 416 due to the need to apply convolutions to bigger images. One of the reasons these two versions are included is to compare how different image sizes affect the results. We also include SSD,¹² which is prior to YOLOv2 and introduced the prediction of bounding boxes without fully connected layers and the concept of multiscale prediction. These new techniques were later included as an improvement to YOLO in YOLOv2.

SPP YOLOv3²⁷ is a modification of the YOLO-like architecture in which the last pooling layer is replaced by a Spatial Pyramid Pooling layer. YOLO's pooling layer output size relies on the input image size, thus preventing the usage of a fully connected network at the end of the pipeline. Conversely, the Spatial Pyramid Pooling layer divides the feature map into a $N \times M$ grid and performs a max-pooling for each cell of the grid, giving an output size independent to the dimensions of the input image.

Mobilenets²⁶ is a lightweight CNN designed for resource-constrained platforms, such as mobile systems. It includes the concept of depthwise separable convolutions into one-stage detectors. These sort of convolutions represent a variant of the classical convolution with reduced computational costs. This allows the architecture to run in RT even on non-GPU resource-constrained systems.

Experiments

In terms of pedestrian detection, not only should we take into account detection performance, but also throughput, since speed is crucial to confidently implement a detection on-board system. For this purpose, first we go through the most widely used metrics to assess performance and throughput of a detector. Then, the behaviour of some detectors when applied to some important datasets is

analysed. After that, results on training one of the best algorithms (YOLOv3) are presented in order to refine its detections. Finally, we modify an important CNN parameter (grid size) from this detector to evaluate how it affects performance.

Metrics

Common metrics usually used to assess object detection performance are the following

$$\begin{aligned} \text{FPPI} &= \frac{\text{FP}}{n_{\text{images}}} & \text{MR} &= \frac{\text{FN}}{\text{TP} + \text{FN}} \\ \text{Precision} &= \frac{\text{TP}}{\text{TP} + \text{FP}} & \text{Recall} &= \frac{\text{TP}}{\text{TP} + \text{FN}} \end{aligned}$$

where TP, FP and FN stand for True Positives, False Positives and False Negatives, FPPI stands for False Positives per Image and MR for Miss Rate. Therefore, scanning the confidence threshold, the MR versus FPPI and Precision versus Recall curves are obtained.

Finally, summarising particular parts of the curves, three performance measures can be defined

$$\text{LAMR} = \frac{1}{9} \sum_{i=0}^8 \text{MR}(\text{FPPI} = 10^{-2+0.25i})$$

$$\text{AP} = \frac{1}{11} \sum_{i=0}^{10} \text{Precision} \left(\text{Recall} = \frac{i}{10} \right)$$

$$\text{F1}_{\text{max_score}} = 2 \cdot \max \left(\frac{\text{Precision}(i) \cdot \text{Recall}(i)}{\text{Precision}(i) + \text{Recall}(i)} \right)$$

with LAMR standing for *Log-Average Miss Rate* and AP for *Average Precision*, with lower values being better for the first one and higher for the other two. Besides, $\text{F1}_{\text{max_score}}$ represents the maximum value for F1-score at all recall values.

Detection results

Pedestrian is the most difficult instance to detect on a driving task and represents the most vulnerable users on the road. The so-called *reasonable subset* was established as unoccluded or partially occluded (up to 35%) pedestrians with a height of 50 pixels or more, intrinsically derived from the annotation design of the original Caltech introductory article²⁸ while the so-called *difficult subset* contains all the annotations. Thus, bearing this in mind, we applied the transformation required by the *reasonable subset* proportionally to the image sizes of the other three datasets to establish a common *reasonable* scenario in order to compare their complexity and the adaptation of generic algorithms analysed to each on-road dataset.

Figure 2 shows detection performance results of SSD, Mobilenets and the three YOLOv3 versions for Caltech and



Figure 1. Samples of the databases used.

Daimler datasets. It can be noticed that YOLOv3 and its variants present a good performance for both databases. Besides, SSD shows poor precision with high recall, while Mobilenets has poor general performance due to its focus on low computational complexity.

Figure 3 shows detection performance (LAMR) versus speed frames per second (FPS) on the Jetson TX2 mobile platform. It can be shown that YOLOv3 416 presents higher FPS while obtaining a similar miss rate. In our setup, it is important to get a RT response and for this reason YOLOv3 416 will be selected for the next experiments.

Hardware performance

As stated previously, even having constraints mainly due to limited power and space available on board, there is hardware available with enough computing power to deal with algorithms like those presented in the third section. Consequently, we selected an NVIDIA Jetson

TX2 mobile board with multiple GPU and CPU cores that operates at a maximum power consumption of 15 W. Additionally, we also have included some powerful GPUs and general-purpose processors that allowed us to establish a comparison among three platforms: a desktop CPU (Intel Core i7-7700 CPU x 8), a GPU (NVIDIA GTX 1080 Ti) and the mobile GPU platform presented above (NVIDIA Jetson TX2).

We performed a comparative throughput analysis for YOLOv3 416 algorithm running on our three platforms with CUDA backend for GPUs, without any further code optimisation. For this reason, they represent a lower bound (baseline) from which a single forward-pass time of the net will decrease at prediction time in case any optimisation technique is applied.

An analysis of the forward-pass mean time is computed for two of the presented datasets, both containing fixed-size images. We present the results for the three computing platforms mentioned in Table 1.

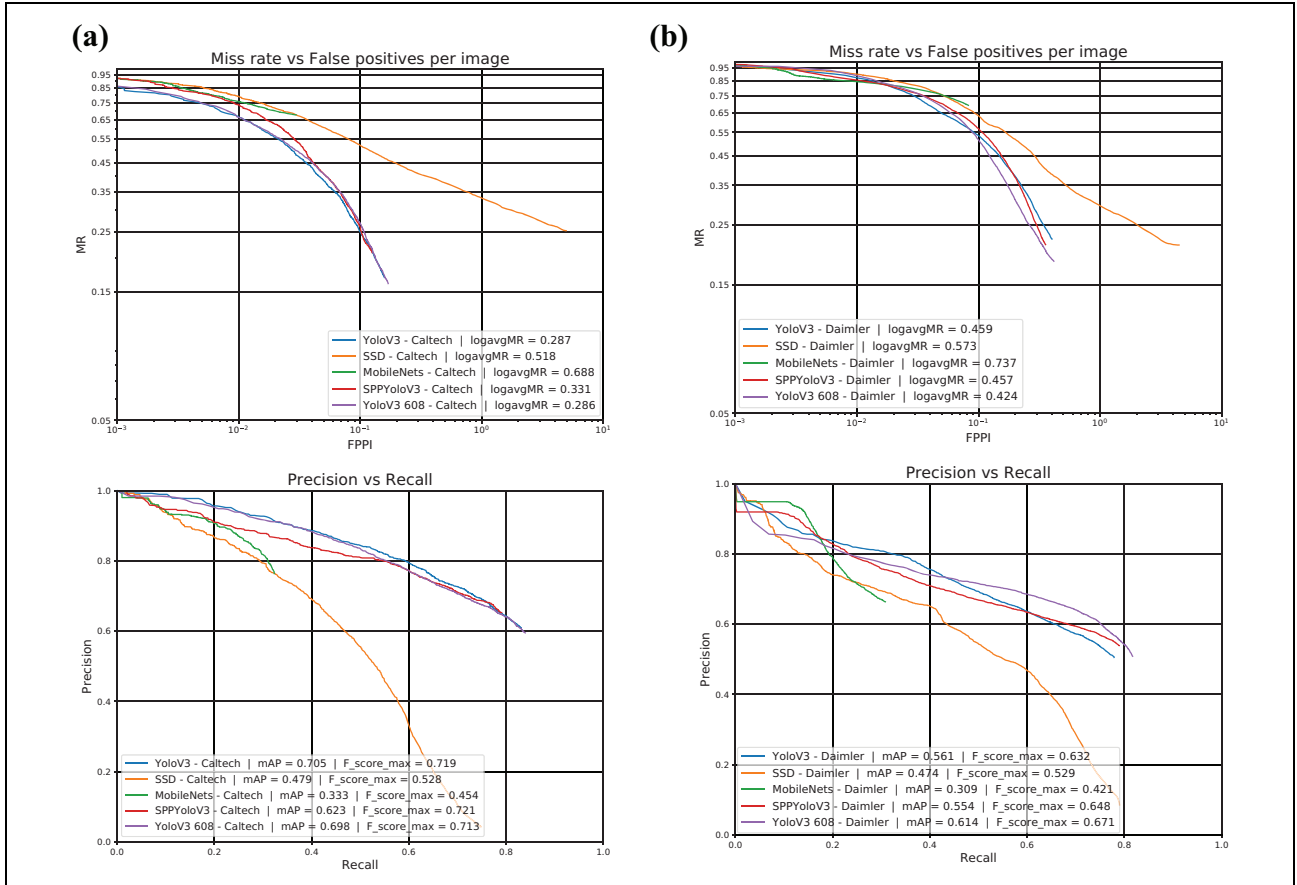


Figure 2. Detection performance of the five analysed generic algorithms applied to (a) Caltech and (b) Daimler datasets in the reasonable scenario.

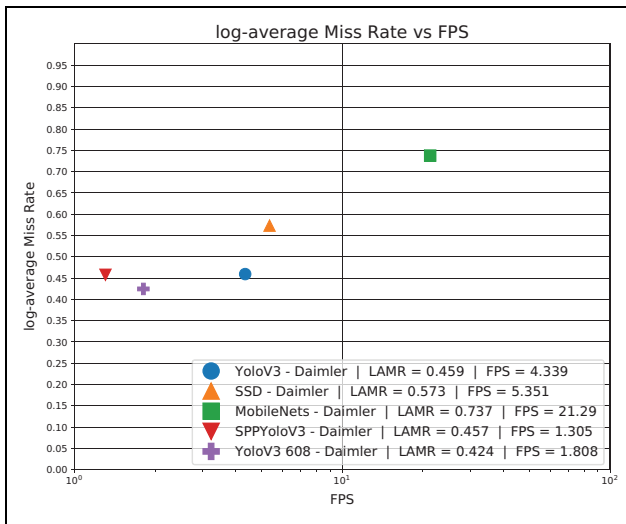


Figure 3. Detection performance versus throughput of the five analysed generic algorithms applied to the four aforementioned on-road datasets.

Driving reaction time in humans is known to range from 500 ms to 2 s,²⁹ so we must guarantee that our system works at a lower reaction time. Jetson TX2 fulfils it.

Table 1. Mean computing time and throughput of a single forward-pass of Daimler images on different platforms for YOLOv3 416.

	Daimler	
	Time (ms)	FPS
Intel Core i7 × 8	707	1.4
GTX 1080 Ti	35	28.6
TX2 Max-N mode	240	4.2

YOLO: You Only Look Once.

The final goal is to achieve high-quality, RT detection of VRUs. Therefore, as our experiments were developed in Python, some optimisation techniques may eventually be applied in order to increase throughput (see the fifth section).

Specific training

In order to improve the network performance as well as to increase the recall levels while keeping precision, we modified the architecture of YOLOv3 detector by reducing the number of classes of the task.

The original YOLOv3 was trained on the generic dataset COCO,²⁰ which includes up to 80 different classes. However,

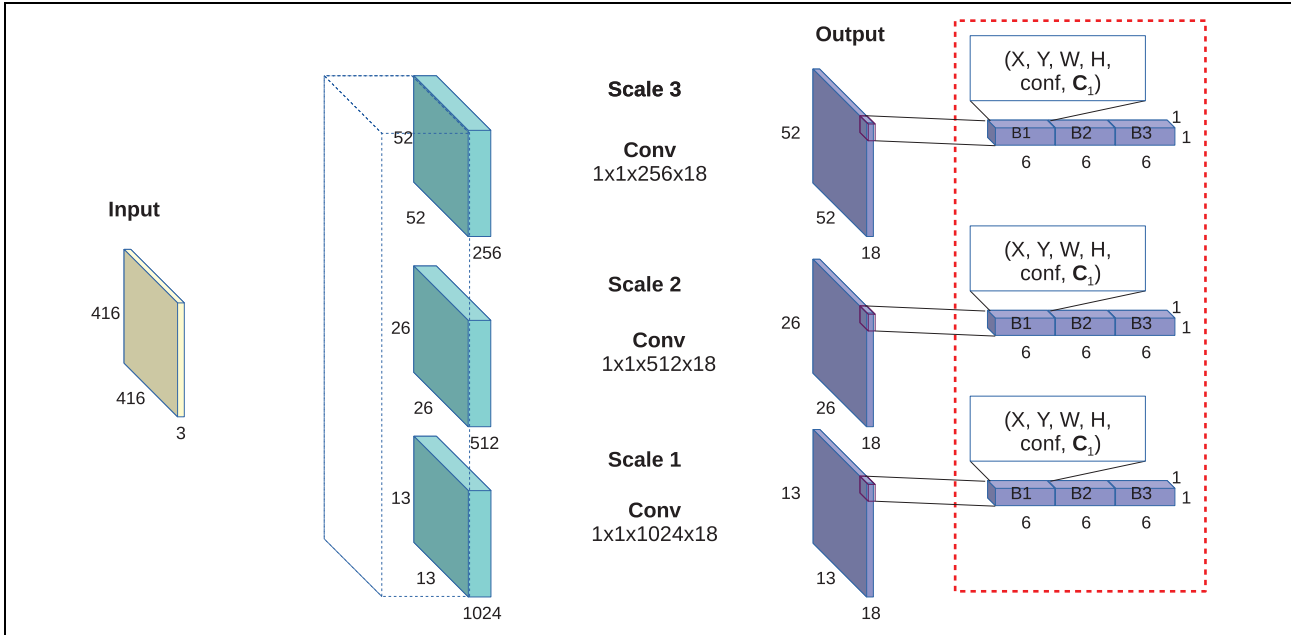


Figure 4. YOLOv3 structure after modifying the last layers with 18 channels instead of the original 255. YOLO: You Only Look Once.

in our task, we are only interested in pedestrians. Therefore, we trimmed the last layer of the network so that it predicts bounding boxes of only 6 floating point values (2 for position, 2 for shape, 1 for bounding box confidence and 1 for the class confidence) in contrast with the original 85 values (see Figure 4). YOLOv3 predicts 3 boxes for each grid cell at 3 different scales of size 13×13 , 26×26 and 52×52 . Thus, for each grid cell, the 3 boxes with 6 parameters are encoded in a $1 \times 1 \times 18$ vector instead of the original $1 \times 1 \times 255$ vector.

This new network has been trained on the same COCO dataset as the original one and tested using the 5000 images from the COCO 2014 validation dataset. Figure 5 shows the results for the COCO 2014 validation dataset. It can be noticed that the single class model outperforms the generic YOLOv3 results, and for a given precision, a higher recall is obtained for this model. Table 2 shows the increase in mean average precision (mAP), higher TP and lower FP, FN for the single class model.

As stated below, the trained model with one class gets better results under the COCO dataset, but it is also interesting to see how it performs for modern, specific, on-road databases such as ECP and nuScenes. For this reason, we have tested the model on ECP and nuScenes databases. Figure 6 also confirms the improvement of the single class model and Figure 7 shows that the single class model has a better recall without getting too much FP.

Grid size analysis

As shown below, we achieved a better mAP by training a model with only pedestrians. From the other side, the original YOLOv3 algorithm uses a 13×13 grid that

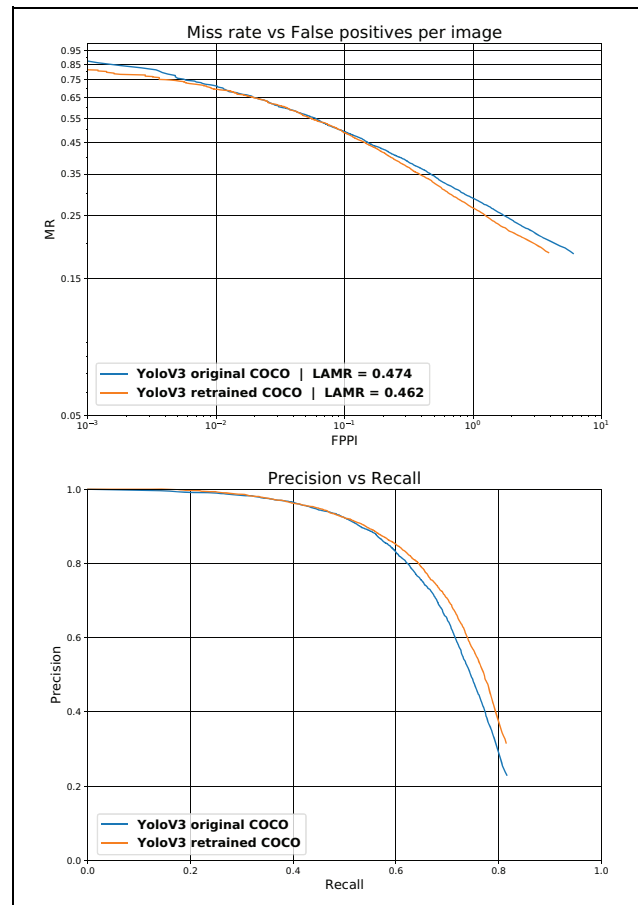


Figure 5. Detection performance of the original YOLOv3 and the YOLOv3 single class for the COCO validation dataset. YOLO: You Only Look Once.

corresponds to 416×416 pixels (grids are regions of 32×32 pixels) and YOLOv3 608 corresponds to a 19×19 grid. Each grid cell predicts 3 objects at 3 different scales, having a total of 10,647 objects. Figure 8 shows the default 13×13 grid.

It can happen that a 13×13 grid it is not the best setup for pedestrian class. Therefore, an extensive analysis has been carried out for different grid (image) sizes. One of the

Table 2. Results for the COCO validation dataset with the original 80 classes and for the single class model.^a

	YOLOv3 80	YOLOv3 1
mAP	72.04%	72.78%
F-score	0.68	0.71
Precision	0.67	0.71
Recall	0.69	0.70
TP	7505	7664
FP	3638	3101
FN	3383	3224

YOLO: You Only Look Once; TP: true positives; FP: false positives; FN: false negatives.

^amAP is computed at IOU = 0.5 and the confidence threshold is set to 0.25.

advantages of the YOLOv3 architecture is that image input (grid) size for inference can be different from the input size used to train the network, thus allowing a fine-tuning step to get the most from the model.

Figure 9 shows the detector performance depending on the grid size. Each line shows the performance for a grid of size $width \times height$, with $height$ being fixed (i.e. $w: 13$) and $width$ being variable. Besides, the x -axis shows the grid area obtained by multiplying $width$ and $height$, which is correlated with the computational cost.

It can be seen that a higher or equal resolution for $width$ instead of $height$ works better than the opposite. Values for mAP for a given grid area increase with higher $width$ and lower $height$ values.

Moreover, it can be seen that some grid configurations get better mAP with similar grid sizes (computational cost). Taking into account that YOLOv3 416 uses a grid size of 13×13 resulting in an area of 169 regions, and YOLOv3 608 corresponds to 19×19 , some intermediate configurations can be chosen in order to keep the computing time low while improving the mAP. Some of the selected points are near a grid area of 225, where the slopes of the curves begin to decrease having smaller mAP/grid ratio. Thus,

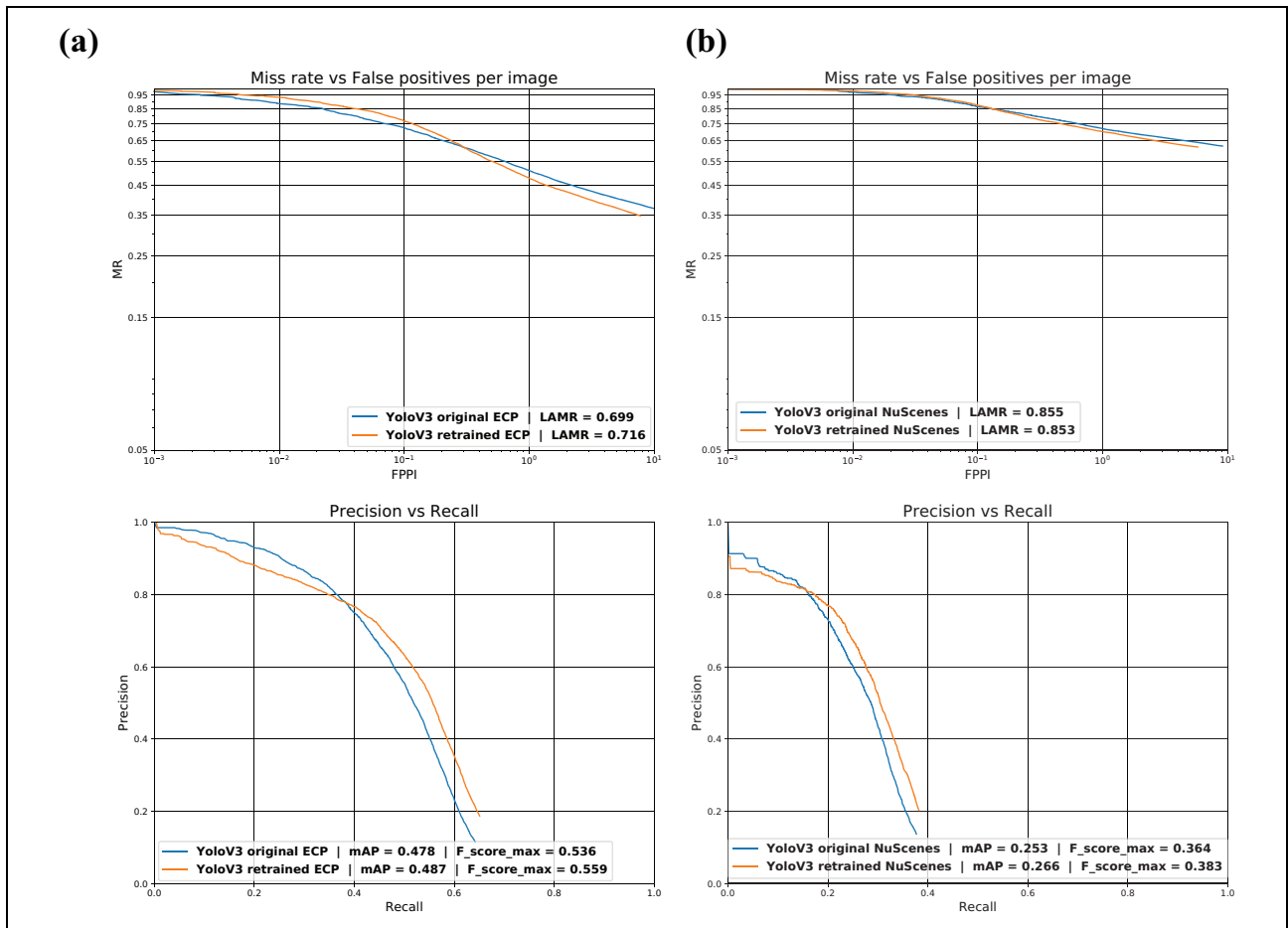


Figure 6. Detection performance of single class trained YOLOv3 for (a) ECP and (b) nuScenes on the *reasonable* scenario. YOLO: You Only Look Once; ECP: EuroCity Persons.



Figure 7. Pedestrian detection for the original YOLOv3 and the YOLOv3 for pedestrians on the ECP dataset: (a) berlin_00414, (b) berlin_00436, (c) budapest_00859, (d) barcelona_01293, (e) amsterdam_01078, (f) ljubljana_01051 and (g) hamburg_00681. From left to right: ground truth, original YOLOv3 and single class YOLOv3. YOLO: You Only Look Once; ECP: EuroCity Persons.

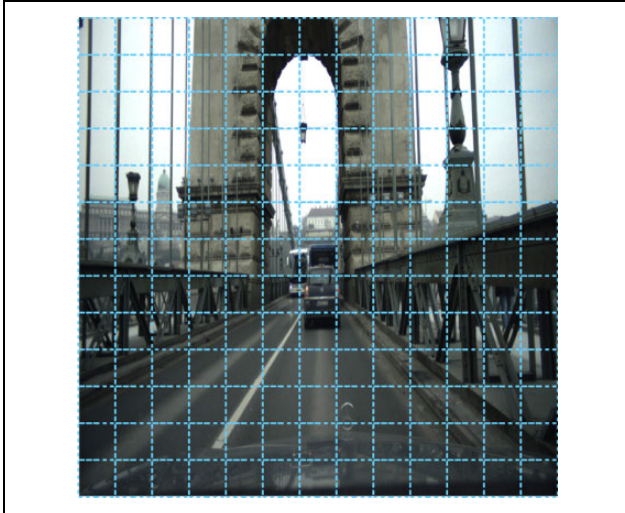


Figure 8. YOLOv3 default 13×13 grid. YOLO: You Only Look Once.

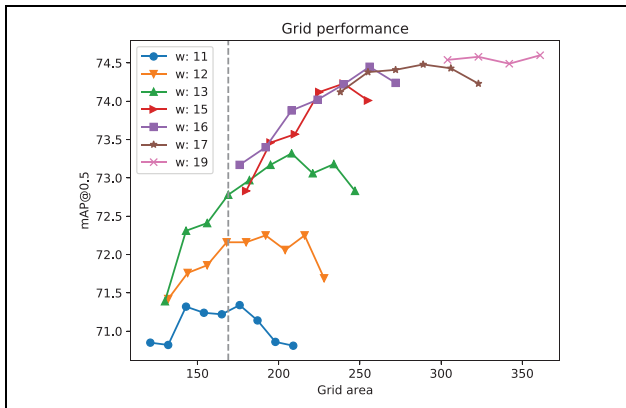


Figure 9. mAP versus grid area (dotted line for YOLOv3 416). YOLO: You Only Look Once.

we have selected 16×11 size, which has almost the same grid elements, and 15×15 , which gets a considerably better mAP (74.12 vs. 72.78 at 13×13) with a slightly larger grid size. It is also worth to mention that 15×15 obtains a recall of 0.72 while 13×13 gets 0.70.

Finally, Table 3 shows the detection results achieved with our training against the COCO validation dataset baseline and the results with our single class model for different grid sizes.

Conclusions and future work

In this article, we have presented five generic-purpose object detection algorithms and their performance on pedestrian detection task for the on-road datasets Caltech-USA, Daimler, ECP and nuScenes. This set of databases includes images in very different conditions, sizes and locations. Then, an experimentation with the YOLOv3 algorithm was carried out on three different hardware platforms, including the mobile platform

Table 3. Results for COCO validation dataset with original 80 classes and single class trained models.^a

	YOLOv3 80	YOLOv3 16 × 11	YOLOv3 16 × 11	YOLOv3 15 × 15
mAP	72.04%	72.78%	73.17%	74.12%
F-score	0.68	0.71	0.71	0.71
Precision	0.67	0.71	0.72	0.70
Recall	0.69	0.70	0.70	0.72
TP	7505	7664	7676	7890
FP	3638	3101	3043	3434
FN	3383	3224	3212	2998

YOLO: You Only Look Once; TP: true positives; FP: false positives; FN: false negatives.

^amAP is computed at IOU = 0.5 and confidence threshold is set to 0.25.

NVIDIA Jetson TX2, in order to establish a throughput baseline for further developments. Moreover, an ad hoc training experiment only with the pedestrian class was performed on the COCO generic dataset and we have been able to significantly increase recall, as we wish a decrease in type II errors in spite of an increase in type I errors which may be mitigated in subsequent stages (such as tracking or sensor fusion with other signals). Finally, as computation time is a critical parameter, different grid sizes have been evaluated, with some of them performing better at slightly higher computation times, getting an improvement in mAP from 72.04% to 74.12%.

As future work, further training of these generic CNNs on some of the presented on-road specific datasets to improve detection results for a number of interest classes (car, bus, pedestrian, bicycle, motorcycle, etc.) is likely to provide a detection performance that, when combined with RADAR and LiDAR inference and other post-processing techniques, delivers superior detection results. In addition, optimisations at code level and libraries, such as using TensorRT and porting Python functions to C, will be also considered.


Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported by PRYSTINE project which had received funding within the Electronic Components and Systems for European Leadership Joint Undertaking (ECSEL JU) in collaboration with the European Union's H2020 Framework Programme and National Authorities, under grant agreement no. 783190. It was also funded by Generalitat Valenciana through the Instituto Valenciano de Competitividad Empresarial (IVACE).

ORCID iD

Ismael Salvador Igual  <https://orcid.org/0000-0001-9269-3737>

References

1. Zhang S, Benenson R, Omran M, et al. Towards reaching human performance in pedestrian detection. *IEEE Trans Pattern Anal Mach Intell* 2017; 40(4): 973–986.
2. Viola P, Jones MJ and Snow D. Detecting pedestrians using patterns of motion and appearance. *Int J Comput Vision* 2005; 63(2): 153–161.
3. Dalal N and Triggs B. Histograms of oriented gradients for human detection. In: *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR '05)*, San Diego, CA, USA, 20–25 June 2005, Vol. 1, pp. 886–893. IEEE.
4. Felzenszwalb P, McAllester D and Ramanan D. Adiscriminatively trained, multiscale, deformable part model. In: *2008 IEEE conference on computer vision and pattern recognition*, Anchorage, AK, USA, 23–28 June 2008, pp. 1–8. IEEE.
5. Dollár P, Tu Z, Perona P, et al. Integral channel features. In: *Proceedings of the British machine vision conference*, London, UK, 7–10 September 2009, pp. 91.1–91.11.
6. Nam W, Dollár P and Han JH. Local decorrelation for improved pedestrian detection. In: *Advances in neural information processing systems*, Montreal, Quebec, Canada, 8–13 December 2014, pp. 424–432.
7. Dollár P, Appel R, Belongie S, et al. Fast feature pyramids for object detection. *IEEE Trans Pattern Anal Mach Intell* 2014; 36(8): 1532–1545.
8. Benenson R, Omran M, Hosang J, et al. Ten years of pedestrian detection, what have we learned? In: *European conference on computer vision*, Zurich, Switzerland, 2014, pp. 613–627, Cham: Springer.
9. Girshick R, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, Columbus, OH, USA, 23–28 June 2014, pp. 580–587.
10. Tian Y, Luo P, Wang X, et al. Deep learning strong parts for pedestrian detection. In: *Proceedings of the IEEE international conference on computer vision*, Santiago, Chile, 7–13 December 2015, pp. 1904–1912.
11. Ren S, He K, Girshick R, et al. Faster R-CNN: towards real-time object detection with region proposal networks. In: *Advances in neural information processing systems*, Cambridge, MA, USA, December 2015, pp. 91–99.
12. Liu W, Anguelov D, Erhan D, et al. SSD: single shot multibox detector. In: *European conference on computer vision*, Amsterdam, The Netherlands, 2016, pp. 21–37. Cham: Springer.
13. Redmon J, Divvala S, Girshick R, et al. You only look once: unified, real-time object detection. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, Las Vegas, NV, USA, 27–30 June 2016, pp. 779–788.
14. Redmon J and Farhadi A. YOLO9000: better, faster, stronger. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, Honolulu, HI, USA, 21–26 July 2017, pp. 7263–7271, http://openaccess.thecvf.com/content_cvpr_2017/papers/Redmon_YOLO9000_Better_Faster_CVPR_2017_paper.pdf (2018, accessed May 2020).
15. Redmon J and Farhadi A. YOLOv3: an incremental improvement. *CoRR* 2018, <http://arxiv.org/abs/1804.02767> (2015, accessed May 2020).
16. Ren S, He K, Girshick RB, et al. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR* 2015, <http://arxiv.org/abs/1506.01497>.
17. Lan W, Dang J, Wang Y, et al. Pedestrian detection based on YOLO network model. In: *2018 IEEE international conference on mechatronics and automation (ICMA)*, Changchun, China, 5–8 August 2018, pp. 1547–1551. IEEE.
18. Hong Z, Zhang L and Wang P. Pedestrian detection based on YOLO-D network. In: *2018 IEEE 9th international conference on software engineering and service science (ICSESS)*, Beijing, China, 23–25 November 2018, pp. 802–806. IEEE.
19. Liu Z, Shi Y and Sun M. A pedestrian detection algorithm based on improved YOLOv2. In: *2018 5th IEEE international conference on cloud computing and intelligence systems (CCIS)*, Nanjing, China, 23–25 November 2018, pp. 488–492. IEEE.
20. Lin T, Maire M, Belongie SJ, et al. Microsoft COCO: common objects in context. *CoRR* 2014, <http://arxiv.org/abs/1405.0312> (2014, accessed May 2020).
21. Dollár P, Wojek C, Schiele B, et al. Pedestrian detection: an evaluation of the state of the art. *IEEE Trans Pattern Anal Mach Intell* 2012; 34: 743–761.
22. Gavrila D and Munder S. An experimental study on pedestrian classification. *IEEE Trans Pattern Anal Mach Intell* 2006; 28(11): 1863–1868.
23. Enzweiler M and Gavrila DM. Monocular pedestrian detection: survey and experiments. *IEEE Trans Pattern Anal Mach Intell* 2008; 31(12): 2179–2195.
24. Braun M, Krebs S, Flohr F, et al. The EuroCity Persons dataset: a novel benchmark for object detection. *CoRR* 2018, <http://arxiv.org/abs/1805.07193> (2018, accessed May 2020).
25. Caesar H, Bankiti V, Lang AH, et al. nuScenes: a multimodal dataset for autonomous driving. *CoRR* 2019, <http://arxiv.org/abs/1903.11027> (2019, accessed May 2020).
26. Howard AG, Zhu M, Chen B, et al. Mobilenets: efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861, 2017.
27. He K, Zhang X, Ren S, et al. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Trans Pattern Anal Mach Intell* 2015; 37(9): 1904–1916.
28. Dollár P, Wojek C, Schiele B, et al. Pedestrian detection: a benchmark. In: *2009 IEEE conference on computer vision and pattern recognition*, Miami, FL, USA, 20–25 June 2009, pp. 304–311. IEEE. DOI: 10.1109/CVPRW.2009.5206631.
29. McGehee DV, Mazzae EN and Baldwin GS. Driver reaction time in crash avoidance research: validation of a driving simulator study on a test track. *Proc Hum Factors Ergon Soc Annu Meet* 2000; 44(20): 3-320–3-323.