Journal of the Brazilian Computer Society, 2026, 32:1, doi: 10.5753/jbcs.2026.4937

An embedded vision-based system for cyclist detection and counting

Leandro Alves dos Santos [**Federal University of Technology — Paraná** | ***santos.2010@alunos.utfpr.edu.br***]
Roberto Cesar Betini [**Federal University of Technology — Paraná** | ***betini@utfpr.edu.br***]
Bogdan Tomoyuki Nassu [**Federal University of Technology — Paraná** | ***btnassu@utfpr.edu.br***]

PPGCA, Federal University of Technology — Paraná, Av. Sete de Setembro, 3165 Rebouças, Curitiba, PR, 80230-901, Brazil.

Received: 10 September 2024 • **Accepted:** 06 October 2025 • **Published:** 15 April 2026

Abstract Automatically detecting and counting cyclists in urban scenarios is a task in intelligent transportation systems and smart cities that enables the generation of important structured data. This data contributes to understanding the dynamics of cyclists' use of the urban space and guides the development of public policies for cycling mobility and traffic safety. In this study, we propose an embedded system for cyclist detection and counting, aiming to be a lightweight solution using computer vision and deep learning methods. It is characterized by low energy consumption and easy handling, based on the Raspberry Pi 4 platform and the Edge TPU Coral accelerator. The developed system achieved an $F1$-score of 0.9137 for processing prerecorded video. In experiments conducted in a real urban setting, we achieved counting accuracy between $78{,}3\%$ and $82{,}2\%$, a performance comparable to solutions with higher computational requirements and/or costs. Code is available at `https://github.com/leandroAS86/det-cicle`

Keywords: Cyclist Detection, Intelligent Transport Systems, Smart Cities, Computer Vision, Deep Learning.

1 Introduction

Cyclomobility is a mode of active mobility in which the bicycle is used as the primary means of transport, characterized by its reliance on human power. The practicality and efficiency for short and medium-distance trips, together with its low cost, make it a more accessible option for urban mobility, recreation and sports. Additionally, as it requires movement and energy expenditure, it also provides health benefits by directly combating sedentary lifestyles [Piatkowski and Bopp, 2021; Oja *et al*., 2011].

Like pedestrians, cyclists are vulnerable road users, as they are unprotected while cycling and an accident can result in serious injuries or even death [García-Venegas *et al*., 2021]. Therefore, it is important for public authorities to pay more attention to making cities safer for cyclists. Automatic cyclist counting monitors the use of this transport mode, providing crucial data for creating for creating public cycling policies, such as the construction and resizing of cycle lanes, and adjusting vehicle speed limits [Beitel *et al*., 2018].

Pneumatic tube and inductive loop counters are common methods for automatically counting cyclists. These systems have an accuracy of around 85% and provide supplementary information regarding the direction of travel [Ozan *et al*., 2021]. However, both methods only perform counting at specific points. Moreover, inductive loop counters are intrusive to the pavement and require specialized labor for installation.

The fields of computer vision and deep learning have been important for the development of intelligent transport systems and smart cities. Many studies have been proposed for vehicle detection and traffic surveillance [Guindel *et al*., 2018; Mhalla *et al*., 2018; Liu *et al*., 2021], as well as cyclist detection [Masalov *et al*., 2018; García-Venegas *et al*., 2021]. These studies focus on offline analysis of traffic scenarios. In contrast, other proposals are based on embedded platforms, which are low-cost, compact, and easy to install, aimed at local processing [Stahl *et al*., 2023; Jin *et al*., 2020].

This paper presents a system for detecting and counting cyclists in urban settings. Based on the YOLOv8n deep learning architecture, it performs on-site processing with video camera capture. To achieve this, the camera was positioned with an elevated view alongside the road, which allows for an expanded monitoring area. For improved counting, the system employs cyclist tracking, which also enables determining the direction of travel. The system runs on a Raspberry Pi 4 (RPI4) embedded platform with an Edge TPU Coral USB accelerator. An overview is shown in Figure 1 [1]. The equipment can be installed next to the road at a height of 4 to 5 meters, with the camera's field of view adjusted to a lateral or longitudinal perspective. Given its low power consumption, the system can be powered by solar energy.

Local processing is useful for operation in areas without internet connection for transmitting image data for remote processing, besides reducing costs by eliminating the need for storage infrastructure and data security for protection against unauthorized leaks and to ensure privacy. In locations with Internet connectivity, this solution enables more agile monitoring by sending counting data at regular intervals.

This approach aimed to meet technical recommendations for cyclist counting, allowing the selection of an appropriate measurement location on the road, whether on two-way streets or intersections. It also aimed to protect the count against the

[1] Adapted from **Bicycle infrastructure plan** — Curitiba Institute for Research and Urban Planning: `https://ippuc.org.br/storage/uploads/4a17bf54-5a50-419d-aee7-ce3b34eeade7/plano_de_estrutura_cicloviaria.pdf`

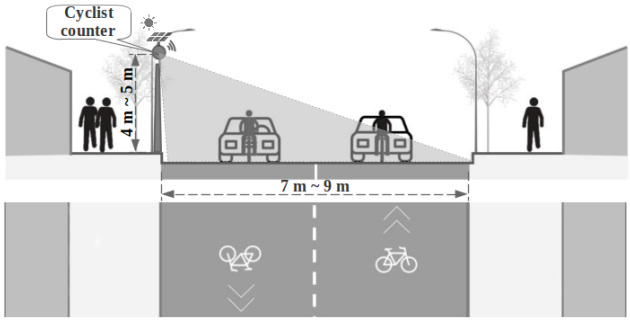


Figure 1. Cyclist counting system. The device is installed alongside the road at a height between 4 and 5 meters.

negative effects of dense crowds or severe occlusion, thereby making the automatic count closer to those carried out by humans, as shown in Figure 2 [2]. In on-site counting tests conducted at locations and times of intense movement in the city of Curitiba, Brazil, the proposed system showed a success rate of between 78,3% and 82,2% compared to visual counting. This result is comparable to solutions with higher costs [Ozan *et al*., 2021].

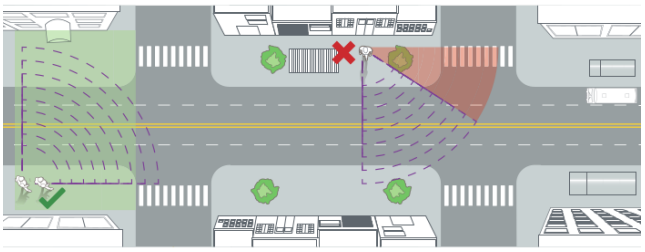


Figure 2. Suitable locations for counting when performed by humans.

This study contributes to the field by proposing and evaluating a lightweight, practical, autonomous system that leverages advances in computer vision and deep learning for urban scene analysis, achieving performance comparable to other available approaches [Stahl *et al*., 2023; Jin *et al*., 2020]. In addition, we propose a post-processing step that assists counting based on the quantity of detections and the distance between detections obtained from the frame sequence by cyclist tracking.

The rest of the paper is structured as follows. Section 2 describes the state of the art in this research area, while Section 3 outlines the development process of the proposed system. Section 4 presents the obtained results, and Section 5 outlines our conclusion and points to future work.

2 Related Work

Traditional computer vision systems for object detection employed methods such as SIFT, HOG and LBP for feature extraction and k-NN, SVM and AdaBoost for classification [Boukerche and Hou, 2021]. In recent years most approaches are based on deep learning architectures and convolutional neural networks. Deep learning approaches enable extracting features with greater representational power, which is important for the detection task, and have become the main mechanism currently used [Zou *et al*., 2023].

Deep learning methods are classified into two-stage — e.g. Faster R-CNN [Ren *et al*., 2017], and single-stage — e.g. SSD [Liu *et al*., 2016], YOLO [Redmon *et al*., 2016] and EfficienteDet [Tan *et al*., 2020]. Two-stage architectures first generate candidate regions for the presence of objects and then perform classification, while single-stage architectures perform both tasks in a single network pass. Each architecture varies in network size, accuracy, processing speed, among other characteristics. The choice of which one to use must take into account matters such as available hardware, training datasets, inference time and the desired accuracy. Studies have analyzed the application of deep learning based approaches in urban scenarios [Boukerche and Hou, 2021] and their use with embedded devices for image object detection and classification [Rodrigues Moreira *et al*., 2025]. Other research has evaluated the performance of various architectures for cyclist detection [Liu *et al*., 2021; García-Venegas *et al*., 2021].

Table 1 summarises the characteristics of the state-of-the-art, highlighting the similarities and differences with this study. The generation of data for application in public policies aimed at urban planning and improving bicycle mobility has been addressed in Stahl *et al*. [2023]; Jin *et al*. [2020] and Liu *et al*. [2021].

Embedded systems using lightweight detectors through classical artificial intelligence models are described by Stahl *et al*. [2023] and Jin *et al*. [2020]. These studies aimed to achieve characteristics such as low cost, power consumption, compactness, ease of installation and non-invasiveness to the traffic flow of cars and pedestrians.

The system for classifying and counting cyclists proposed by Stahl *et al*. [2023] was based on thermal images captured by a long-wave infrared sensor, with a pre-processing stage running on the RPI4 platform to extract features, and the NM500 processor being responsible for classification. This solution is susceptible to the environment temperature, which poses challenges in situations of occlusions and similarity in this type of image. In addition, low-resolution thermal sensors tend to be an alternative to the higher costs of better quality sensors, which limits the system's coverage area.

Employing a millimeter-wave (mmWave) sensor-based approach and the Nvidia Jetson Nano embedded platform, Jin *et al*. [2020] proposed a system for recognizing pedestrians and cars applied to scenarios involving intersections. By being able to process high-resolution images, it was possible to overcome problems related to adverse weather conditions and distance, being effective in recognizing objects at distances of up to 30 meters. A multivariate Gaussian mixture (GMM) model was developed and used as a classifier. Although the system was only tested for these two classes, the authors mention the possibility of including new classes in future work, allowing for the detection of cyclists as well as the use of deep learning models.

Automotive applications were the focus of studies by García-Venegas *et al*. [2021]; Allebosch *et al*. [2020]; Guindel *et al*. [2018] and Masalov *et al*. [2018]. These studies describe systems for object detection using moving cameras. Furthermore, they were only tested on pre-recorded videos

[2] Report **Cyclist Counts – Technical Recommendations and Monitoring** — Institute for Transportation and Development Policy (ITDP): `https://itdpbrasil.org/wp-content/uploads/2018/10/Contagens-de-ciclistas_ITDP_out2018_v04.pdf`

Table 1. Similarities and differences between related works.

Authors	Actions	Characteristics	Hardware
Stahl *et al*. [2023]	Counting Urban planning No travel direction	Embedded hardware Smaller coverage area Low-resolution thermal sensor (160×120)	RPI4 + NM500 LWIR
Jin *et al*. [2020]	Counting Urban planning	Embedded hardware Not tested for cycling class Traffic monitoring at intersections	Nvidia Jetson Nano xWR1843BOOST
Liu *et al*. [2021]	Counting Urban planning No travel direction	Prerecorded videos Based on fixed urban monitoring system	Nvidia Titan Xp
García-Venegas *et al*. [2021]	Automotive applications Detection and risk evaluation	Prerecorded videos Camera at cyclist's level Lightweight deep learning	Nvidia RTX 2070
Allebosch *et al*. [2020]	Cycling competitions application Detection and drafting violation	Camera at cyclist's level Lightweight deep learning	Nvidia GTX 1060
Masalov *et al*. [2018]	Automotive applications	Prerecorded videos Camera at cyclist's level Based on cycling jersey patterns	Intel Core i7
Guindel *et al*. [2018]	Automotive applications Viewpoint estimation	Prerecorded videos	Nvidia Titan Xp
Ours	Counting Travel direction Urban planning	Embedded hardware Lightweight deep learning Larger coverage area Local processing Adjustable site characteristics Higher image resolution (416×416)	RPI4 + Edge Tpu Coral 12 megapixel Sony IMX708 Camera

and did not address data generation.

The study by Masalov *et al*. [2018] proposed a lightweight system for detecting cyclists in videos, exploiting the characteristics of sports jerseys, for which a specialized dataset was created. Their algorithm relies on knowing a wide variety of patterns found in these outfits, which limits the system's applicability, as this type of clothing is not always part of the local culture, and patterns can vary over time and location.

The study by García-Venegas *et al*. [2021] aimed at improving cyclist safety, based on assessing the risk of accidents. The study emphasized that the SSD MobileNetv2 achieved the best average precision (AP) and inference time. Liu *et al*. [2021] used Faster R-CNN for detecting and counting cyclists and pedestrians, achieving good results in detecting small objects and in heavy traffic conditions. They employed YOLOv2 and SSD for detecting other vehicles due to the lower inference time. Guindel *et al*. [2018] also used Faster R-CNN for cyclist detection and viewpoint estimation, while Allebosch *et al*. [2020] used YOLOv3 to detect cyclists in cycling competitions and perform drafting detection, successfully detecting cyclists at distances greater than 20 meters. All of these studies rely on GPUs with superior processing capabilities to run the deep learning models on which they are based.

Studies involving deep learning models often focus on offline processing, whereas those based on embedded platforms tend to use classic computer vision models. This study is distinct because it focuses on running lightweight deep learning models on embedded platforms for local processing of images captured in real time. In addition, we apply tracking to enhance counting accuracy and determine travel direction. Our system can be easily adjusted to different types of locations, such as intersections, two-way roads, bicycle lanes, squares, and parks. To achieve this goal, we proposed a pattern of specialised datasets and created a database that, to our knowledge, has no counterpart with similar characteristics in the literature. Another point to be highlighted in this study is that we evaluated the feasibility of the system's autonomy through solar power and conducted practical on-site testing.

3 Materials and Methods

In this section, we describe the equipment and methods applied to develop the proposed system.

3.1 Devices and equipments

The Raspberry Pi 4B (RPI4) platform is equipped with a quad core Cortex A72 CPU running at 1.5 GHz. Although it offers a good cost versus processing power ratio, the RPI4 has high latency for processing deep learning architectures, so we decided to integrate it with the Edge TPU Coral coprocessor, which incorporates an Application Specific Integrated Circuit (ASIC) designed for models based on the TensorFlow library and performs up to 4 trillion operations per second (TOPS), with 0.5W per TOPS.

With maximum processing using the CPU's 4 cores, the RPI4 consumes 6.5mW, making it possible to use a portable solar panel to power the system. The used solar panel provides a voltage of 5V and an output current of 2A when operating at full power. This panel has $260 \times 160 \times 30$mm and weights 0.4 kg, preserving the equipment's easy handling requirements. Figure 3a shows the RPI4 connected to the Coral via USB and the solar panel fixed above the equipment. Figure 3b shows how the equipment was positioned alongside the road using a 4-meter tall tripod for conducting tests.

For training the deep learning models, we used a PC with an Intel Core i7 processor, 32 GB of RAM, and an NVIDIA

(a) All devices connected and the camera tilted at an angle of approximately $25°$.

(b) Equipment positioned at the sidewalk using a 4-meter tall tripod.

Figure 3. The cyclist counting hardware and equipments.

RTX 3070 TI GPU with 8 GB of RAM. The used software libraries were TensorFlow 2.11.0 and PyTorch 1.13.0 for Python 3.9 running on Linux Ubuntu 22.04 64-bit.

3.2 Datasets

Public image datasets are highly important in the field of computer vision. For studies involving urban scenes, the KITTI [Geiger *et al*., 2012] and Cityscapes [Cordts *et al*., 2016] datasets are among the most widely used, while for cyclist detection the CIMAT-Cyclist [García-Venegas *et al*., 2021] and TDCB [Li *et al*., 2016] datasets stand out. However, due to the great difference between the images in these sets and the real scenario in which the system is applied, it is necessary to supplement them with data that better represents the considered scenario, improving detection performance. In this study, the CIMAT-Cyclist and TDCB datasets were supplemented with images from the intended field of view.

To create our own dataset, 185 minutes of video were recorded at a resolution of 1280×1080 and 30 FPS, from which we extracted 18,330 images containing cyclists and 1,706 containing only the background, annotated with the RoboFlow software. These videos were split into 145 minutes for training, 20 minutes for validation, and 20 for testing. Splitting the setsbefore extracting the images guarantees the same instance of a cyclist does not appear in multiple sets. Data was collected between August and September 2023 at 5 points in the urban scenarios with heavy pedestrian, bus, and car traffic in varying weather conditions.

While the developed dataset shares an urban environment focus with the CIMAT-Cyclist and TDCB datasets, it differs in its field of view. The developed dataset contains images of cyclists of varying sizes and perspectives with multiple cyclists often visible from different angles within a single image. This allows for a more comprehensive representation of the proposed real-world application scenarios. To the best of our knowledge, no other dataset has these features. Figure 4 shows six representative images from the dataset that illustrate these features. The images were captured using a Wide NoIR camera, which has an infrared filter that alters the color palette during the day.

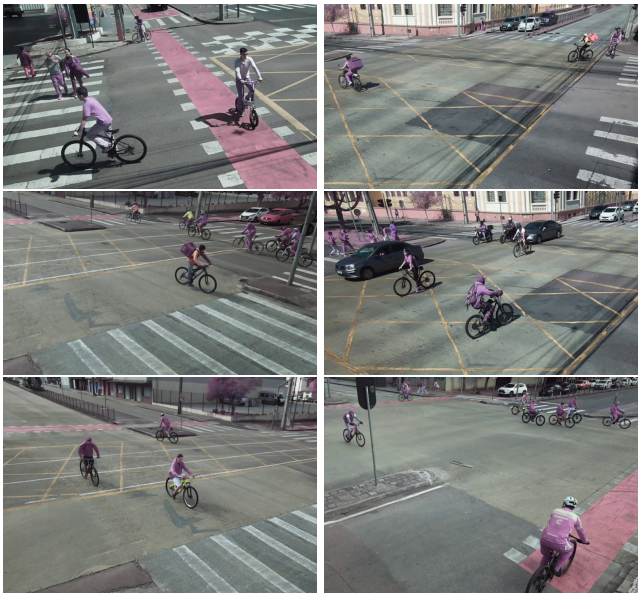


Figure 4. Example of images present in the developed datasets.

Table 2 describes the composition of the dataset, with a total of 44,801 images divided into training, validation, and test sets.

Table 2. Characteristics of datasets for cyclist detection.

Base	Train	Validation	Test	Total
TDCB	9,741	1,019	2,914	13,674
CIMAT	7,104	1,776	2,211	11,091
Ours	16,874	1,781	1,381	20,036
Total	33,719	4,576	6,506	44,801

3.3 Evaluation Metrics

The most commonly used metric for evaluating the performance of detection approaches whose output is given by rectangular regions is the Intersection over Union (IoU), calculated by the Equation 1, which measures the overlap between each expected bounding box (B_g) and its corresponding prediction by the system (B_p). The expected bounding boxes are those labeled in the test set. An object is considered to be successfully detected if the IoU measured for its labeled region exceeds a given threshold. In competitions like COCO [Lin *et al.*, 2014], a threshold of 0.5 is typically considered, the same value adopted by this work.

$$IoU = \frac{area(B_p \cap B_g)}{area(B_p \cup B_g)} \tag{1}$$

A detection will be considered:

- **True positive (TP):** if the object is detected with an IoU greater than or equal to the threshold;
- **False positive (FP):** if the detection does not contain a corresponding object;
- **False negative (FN):** if the IoU is below the threshold.

From these definitions, it is possible to obtain precision and recall metrics according to Equation 2 and 3, respectively. The $F1$-score is then obtained from these metrics, as per Equation 4. The Average Precision [Li *et al.*, 2014] (AP) is obtained as the area under the curve formed by recall and precision, calculated by varying the threshold, which controls the sensitivity of the detector.

$$P = \frac{TP}{TP + FP} \tag{2}$$

$$R = \frac{TP}{TP + FN} \tag{3}$$

$$F1\text{-score} = \frac{2 \times P \times R}{P + R} \tag{4}$$

For the complete system, including tracking across frames, we also consider a single rate, which represents the proportion of cyclist instances that were successfully counted.

3.4 Selection of deep learning architecture

Training a CNN from scratch is a computationally expensive task that requires a large amount of data to produce an effective model. To overcome these difficulties, the transfer learning technique is widely used. This involves applying a generic detector, trained on a large and varied dataset, to solve a problem in a specific domain, based on a fine-tuning stage on a smaller base, representative of this domain. This technique was used to train and evaluate three architectures in this study: EfficientDet-D0, SSD-MobileNetv2, and YOLOv8n. The models were selected for their suitability for edge computing, which requires lightweight architectures [Rodrigues Moreira *et al.*, 2025; Rodriguez-Conde *et al.*, 2021]. At the time of prototype implementation, YOLOv8n was the latest version available for our target platform.

The Edge TPU Coral device runs models with the network weights in 8-bit integer format, requiring a quantization procedure to convert the weights of the model trained in 32-bit floating point format. This makes the model smaller and faster without significantly affecting the network's inference accuracy [Ma *et al.*, 2023]. Figure 5 illustrates the training procedure to obtain the model in the compatible format.

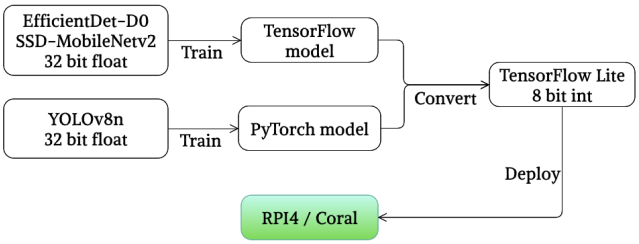


Figure 5. Activity diagram for training EfficientDet-D0, SSD-MobileNetv2 and YOLOv8n and obtaining the quantized model in the 8-bit format supported by the Edge TPU Coral device.

For an initial evaluation of the architectures, the models were initialized with pre-trained weights and fine-tuned for 10 epochs and with batch size 64, using only the public cyclist datasets. The $F1$-score was computed from a small set of 109 images collected from the Flickr social network and brazilian web page called *Cicloativismo*[3], containing 155 instances of cyclists. Processing speed was measured on the RPI4 + Edge TPU Coral. Table 3 shows the results.

Table 3. Performance of the tested models for 109 images from Flickr and the *Cicloativismo* page.

Model	Resolution	F1-score	FPS
EfficientDet-D0	320×320	0,9085	16
SSD-MobileNetv2	320×320	0,9459	10
YOLOv8n	320×320	0,9484	40
YOLOv8n	416×416	0,9579	31

The YOLOv8n model had significantly lower processing time compared to EfficientDet-D0 and SSD-MobileNetv2. This allowed for an increase in the input image resolution of the network (416×416), while still keeping the frame rate higher, contributing to improving network performance and, consequently, enhancing results, making it more suitable for the proposed objectives. Further increasing the resolution did not improve the results.

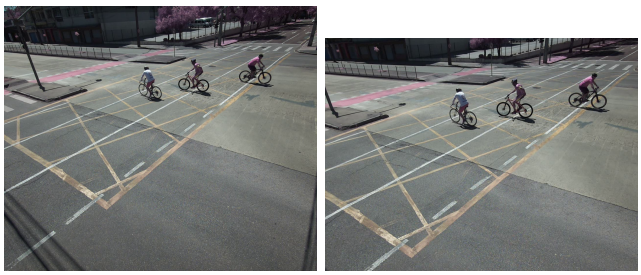
Given the variety and amount of images in the dataset and the fact that there is a single target class — the cyclist class — common data preprocessing techniques, like data augmentation and balancing were not employed, as they are recommended to address issues with limited data or class imbalance. As it is important to prevent the same cyclist from appearing in both the training and test sets in order to ensure

[3] `https://www.cicloativismo.com/`

training quality, no cross-validation was performed, which could lead to the results being overestimated.

3.5 Cyclist Counting System

For the final system, the inputs for the network are the frames captured by the camera, pre-processed so that only a central region of interest, where cyclysts may appear, is kept. This region is defined manually, according to the camera positioning, as shown in Figure 6.



(a) 1280 × 1080 pixel. **(b)** 1080 × 800 pixels.

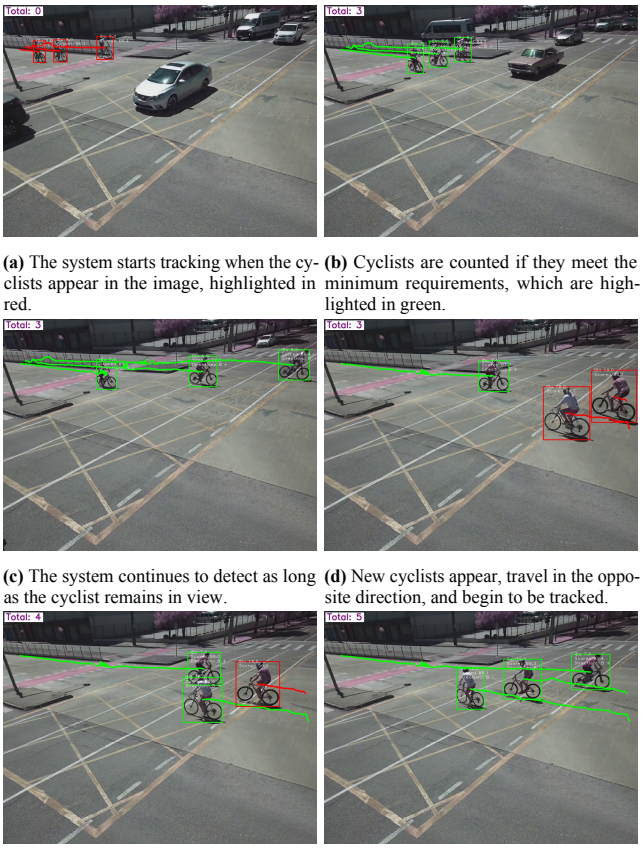
Figure 6. Pre-processing and selection of the region of interest. Original of the 1280 × 1080 frame reduced to 1080 × 800.

Tracking is an efficient way of improving the performance of a detection system against false negative and positive results [García-Venegas *et al*., 2021; Allebosch *et al*., 2020], as well as helping to count objects [Liu *et al*., 2021]. In this study, tracking is also used to obtain additional information about the cyclist's direction of travel. The BoT-SORT tracking algorithm [Aharon *et al*., 2022], which is based on the Kalman filter, was selected based on its satisfactory performance when combined with YOLOv8n on the Raspberry Pi 4. Other approaches in the literature include ByteTrack [Zhang *et al*., 2022] and StrongSORT [Du *et al*., 2023]. However, the primary objective of this study was to demonstrate the feasibility of integrating a tracking technique into the system to enhance its robustness and completeness. A detailed comparative analysis between different tracking algorithms is beyond the scope of this study.

The counting procedure takes into account the number of detections of the same cyclist and the Euclidean distance of travel, in pixels, since the first detection. These two measures are obtained by updating the history of detections of each cyclist through the frame sequence, thus allowing the count to be incremented for each cyclist who reaches 10 detections, with a minimum distance of 100 pixels. These values were defined empirically. Figure 7 shows a sequence of frames illustrating this process. Five cyclists are tracked and counted as long as they appear in the field of view.

4 Results and Discussion

This section presents the results of the study. First, the performance of the model on single images is shown, followed by the performance of the complete system on video, in line with the on-site results. Some qualitative observations and comparisons regarding state-of-the-art are also presented.



(a) The system starts tracking when the cyclists appear in the image, highlighted in red. **(b)** Cyclists are counted if they meet the minimum requirements, which are highlighted in green.

(c) The system continues to detect as long as the cyclist remains in view. **(d)** New cyclists appear, travel in the opposite direction, and begin to be tracked.

(e) The system increments the count. **(f)** All cyclists were accounted for.

Figure 7. Sequence of frames illustrating the detection of five cyclists.

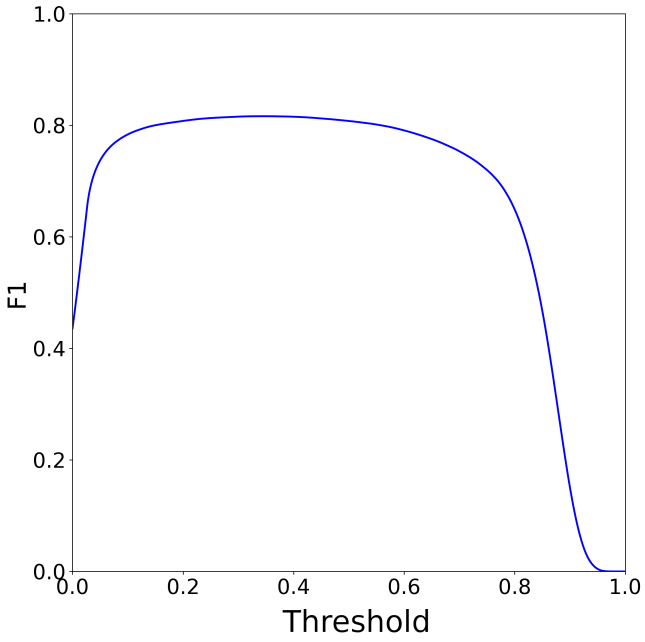
4.1 Model Performance on Single Images

The chosen architecture, YOLOv8n, was trained on our full dataset (Table 2) for 30 epochs. The performance on the test set for $F1$-score was 0.8201 with a threshold of 0.346, as shown in Figure 8a, and for AP was 0.8240, as shown in Figure 8b. The smaller $F1$-score, compared to those reported for a small public dataset in Table 3, is a consequence of including images from the real-world scenario, where the size of the cyclist is reduced in the field of view above the road, as shown in Figure 9a, compared to the images from the public dataset, as shown in Figure 9b.

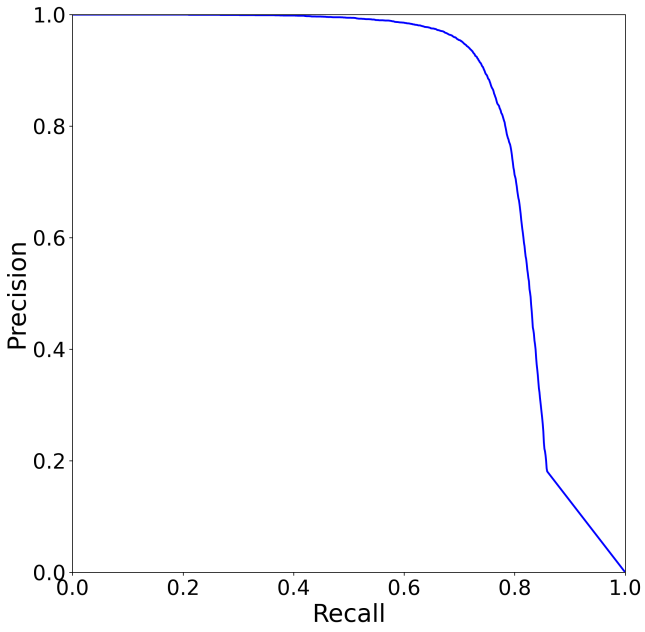
The better AP values reported by other studies focused on cyclist detection are presented in Table 4. Allebosch *et al*. [2020] tested their sports competition dataset only with YOLOv3 model. García-Venegas *et al*. [2021] and Li *et al*. [2016] evaluated several deep learning models using the CIMAT-Cyclist and TDCB datasets, respectively. With the dataset developed in this study, integrated with these two datasets, it was possible to achieve better AP performance, even with a lightweight deep learning model, compared to the two-stage architectures in those studies.

Table 4. AP performance reported in the literature.

Study	Better AP	Model
García-Venegas *et al*. [2021]	0.8190	Faster R-CNN
Allebosch *et al*. [2020]	0.7719	YOLOv3
Li *et al*. [2016]	0.7460	SP-Fast R-CNN
Ours	0.8240	YOLOv8n



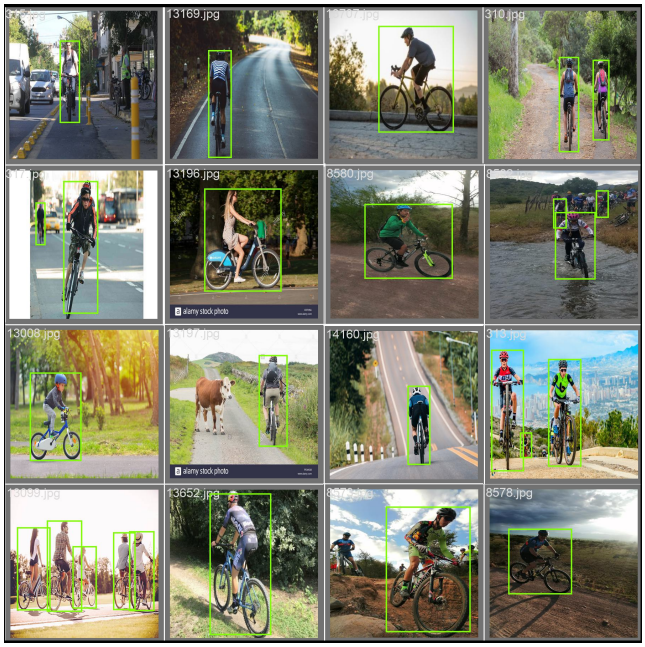
(a) $F1$-score



(b) AP

Figure 8. YOLOv8n performance on the test set.

(a) Detections in our own test dataset.



(b) Detections in the public test dataset.

Figure 9. Detections in the test set.

4.2 System Performance on Videos

The complete system was tested on the 20 minutes of the test videos we captured for our own test dataset, which contain 103 instances of cyclists. The system's performance is shown in Table 5, considering different detection criteria. Combining the pre and post-processing stages shows a gradual increase in the $F1$-score. Although more rigorous detection led to a small increase in false negatives, there was a more significant reduction in the number of false positives. Cropping the borders of the frame helps to improve the detector's performance, and consequently improves tracking.

To count moving objects in Liu *et al*. [2021], a virtual line was drawn as a reference in the image, and the object is counted whenever it crosses this line. The approach in the present study is better suited for the scenarios we consider, where the cyclist's movement is unpredictable and the choice of the locations may include intersections between avenues, making it difficult to define a region in the image to draw a reference line.

Table 5. I: Full frame + number of detections, II: full frame + distance, III: full frame + number of detections + distance, IV: cropped frame + number of detections + distance.

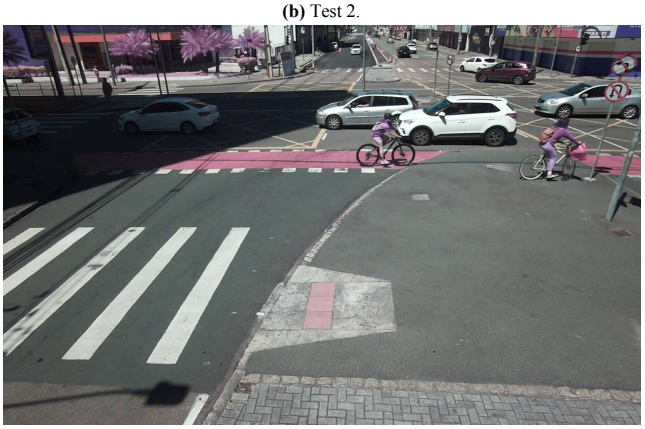
Criteria	TP	FP	FN	F1-score
I	92	19	11	0.8591
II	90	14	13	0.8695
III	89	9	14	0.8856
IV	90	4	13	0.9137

After the offline tests, we evaluated the system's performance on site. The solar panel was connected to a UPS (Uninterruptible Power Supply) charging module. The mod-

ule, equipped with two 3.7 V cylindrical rechargeable Li-Ion 18650 batteries, provided sufficient power for the system. Three tests were carried out in real scenarios, each lasting one hour. For these tests, we considered a single metric that represents the proportion of cyclists successfully counted. The number of cyclists was recorded both manually through human visual count and automatically by the system. The chosen locations connect the city center to populated neighborhoods, having a constant flow of cyclists, pedestrians, buses and automobiles. These locations are distinct from those used for the database's image collection. Tests 1 and 2 were conducted on weekdays, and Test 3 on a weekend, with all trials taking place under sunny conditions. Figure 10 shows the camera's view at these sites.

(a) Test 1.

(b) Test 2.



(c) Test 3.

Figure 10. On-site test fields of view.

The counting accuracy is presented in Table 6. The lower performance compared to calculations based on test videos, as shown in Table 5, was expected. This can be attributed to the fact that, in addition to the higher processing latency caused by the limited computational resources of low-cost embedded platforms [Rodrigues Moreira *et al*., 2025], real-time local processing is more challenging due to adverse weather conditions. For instance, gusts of wind caused the structure to sway. These challenges are addressed in recent literature [Nie and Wang, 2025].

Table 6. System performance with on-site processing.

Site	Human	System	Counting accuracy
1	69	54	78,3%
2	62	51	82,2%
3	94	77	81,9%

When analyzing the results, we noticed the possibility of reducing the occurrence of false negatives and improve counting performance by adjusting the positioning of the camera, providing a side view of the cyclist. This can be seen in tests 2 and 3, in which this positioning was adopted, resulting in a counting performance exceeding 80%. In test 1, the camera predominantly captured the cyclist from a front or rear perspective, a more challenging situation.

A difficulty in detecting cyclists was observed when they appear at the corner of the image. In some situations, cyclists ride on the sidewalk, which compromises the ability to identify them, because in these circumstances the time the cyclists remains in the camera's field of view is not enough to count them. This difficulty is demonstrated in Figure 11.

Figure 11. Unaccounted cyclist. In the left image, the cyclist appears at the upper right corner, identified by the red bounding box. This area has been enlarged in the image on the right.

Another possible improvement is in the dataset. Currently, it mainly has images in which cyclists appear approximately at the center of the frame. The inclusion of more images depicting cyclists on sidewalks can significantly contribute to improving the detector's performance in this scenario.

This proposal for the integration of embedded systems and the deployment of lightweight deep learning models presents significant technical innovation, achieving results comparable to those of state-of-the-art and traditional pneumatic tube counters.

While pneumatic counters identify only two directions of travel due to point counting, the superior spatial coverage and the tracking system of the proposed method enable the identification of various paths taken by cyclists, as shown in Figure 12, where the green line identifies the tracking.

Based on the Raspberry Pi 4, Coral Edge TPU, and a video camera, the system requires approximately 6.5 mW of power. This low power consumption allows for autonomy through

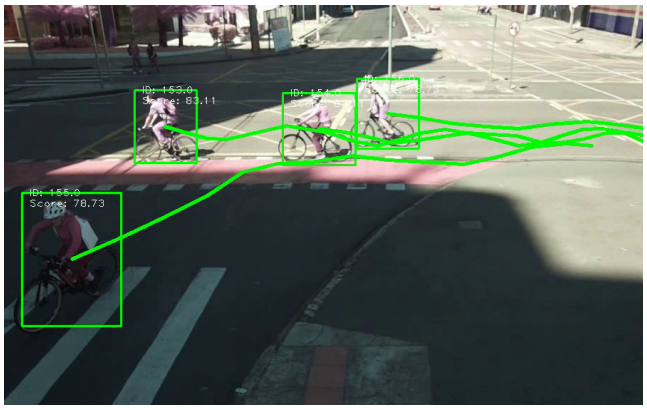


Figure 12. Directions of travel for cyclists. While three cyclists travel in one direction, one cyclist converts to another lane.

solar power and makes it compatible with numerous portable solar panel models currently available.

Regarding the first three studies described in Table 1, which are closely aligned with this research and share a similar focus on using embedded hardware for urban planning and cyclist counting, neither direction of travel nor a solar power supply is avalible or mentioned by the authors. Additionally, the proposed system can be integrated into the Artificial Intelligence as a Service (AIaaS) paradigm [Moreira *et al.*, 2024], which facilitates the deployment of modular and scalable artificial intelligence on heterogeneous edge infrastructures while maintaining local processing capabilities.

5 Conclusions and Future Work

This study aimed to present and evaluate an embedded system designed to automatically detect and count cyclists in an urban scenario. The system is based on the principles of practicality and low energy consumption (6.5mW). To achieve this goal, a lightweight deep learning architecture was selected and trained in accordance with the state-of-the-art for object detection.

This process involved creating a representative image dataset for the system's application context, which enabled us to surpass the state-of-the-art in Average Precision (AP) performance for cyclist detection using the YOLOv8n model. Future work will focus on the creation of a more comprehensive dataset. This will be accomplished by expanding its temporal and spatial scope, including images from various seasons and time periods, and extending coverage to new areas within the city and other cities.

Another important improvement, is the classification of cyclists, since bicycles, when used as a work tool, are often adapted to carry loads by installing luggage racks and bodies, differentiating them from ordinary bicycles. Improving the scope of this practice is crucial due to the diverse ways in which the population uses bicycles, requiring appropriate public measures for traffic safety and infrastructure. This includes the development of appropriate patterns for bicycle lanes and bicycle parking facilities, aiming to create a city that promotes high quality access to bicycle mobility.

Authors' Contributions

L. A. Santos: experiment design and execution, implementation, manuscript writing. R. C. Betini and B. T. Nassu: supervision, writing revision.

Competing interests

The authors declare that they have no conflicts of interest.

Availability of data and materials

The source code is available at `https://github.com/leandroAS86/det-cicle`.

References

Aharon, N., Orfaig, R., and Bobrovsky, B.-Z. (2022). BoT-SORT: Robust associations multi-pedestrian tracking. *Arxiv preprint arXiv:2206.14651*. DOI: 10.48550/arXiv.2206.14651.

Allebosch, G., Van den Bossche, S., Veelaert, P., and Philips, W. (2020). Camera-based system for drafting detection while cycling. *Sensors*, 20(5). DOI: 10.3390/s20051241.

Beitel, D., McNee, S., McLaughlin, F., and Miranda-Moreno, L. F. (2018). Automated validation and interpolation of long-duration bicycle counting data. *Transportation Research Record*, 2672(43):75–86. DOI: 10.1177/0361198118783123.

Boukerche, A. and Hou, Z. (2021). Object detection using deep learning methods in traffic scenarios. *ACM Comput. Surv.*, 54(2). DOI: 10.1145/3434398.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., and Schiele, B. (2016). The Cityscapes dataset for semantic urban scene understanding. *Arxiv preprint arXiv:1604.01685v2*, abs/1604.01685. Available at:`http://arxiv.org/abs/1604.01685`.

Du, Y., Zhao, Z., Song, Y., Zhao, Y., Su, F., Gong, T., and Meng, H. (2023). Strongsort: Make deepsort great again. *IEEE Transactions on Multimedia*, 25:8725–8737. DOI: 10.1109/TMM.2023.3240881.

García-Venegas, M., Mercado-Ravell, D. A., Pinedo-Sánchez, L. A., and Carballo-Monsivais, C. A. (2021). On the safety of vulnerable road users by cyclist detection and tracking. *Mach. Vision Appl.*, 32(5). DOI: 10.1007/s00138-021-01231-4.

Geiger, A., Lenz, P., and Urtasun, R. (2012). Are we ready for autonomous driving? The KITTI vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. DOI: 10.1109/CVPR.2012.6248074.

Guindel, C., Martin, D., and Armingol, J. M. (2018). Fast joint object detection and viewpoint estimation for traffic scene understanding. *IEEE Intelligent Transportation Systems Magazine*, 10(4):74–86. DOI: 10.1109/MITS.2018.2867526.

Jin, F., Sengupta, A., Cao, S., and Wu, Y.-J. (2020). MmWave radar point cloud segmentation using GMM in multimodal traffic monitoring. In *2020 IEEE International Radar Conference (RADAR)*, pages 732–737. DOI: 10.1109/RADAR42522.2020.9114662.

Li, K., Huang, Z., Cheng, Y.-C., and Lee, C.-H. (2014). A maximal figure-of-merit learning approach to maximizing mean average precision with deep neural network based classifiers. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4503–4507. DOI: 10.1109/ICASSP.2014.6854454.

Li, X., Flohr, F., Yang, Y., Xiong, H., Braun, M., Pan, S., Li, K., and Gavrila, D. M. (2016). A new benchmark for vision-based

cyclist detection. In *2016 IEEE Intelligent Vehicles Symposium (IV)*, pages 1028–1033. DOI: 10.1109/IVS.2016.7535515.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. In Fleet, D., Pajdla, T., Schiele, B., and Tuytelaars, T., editors, *Computer Vision – ECCV 2014*, pages 740–755, Cham. Springer International Publishing. DOI: 10.1007/978-3-319-10602-1_48.

Liu, C., Huynh, D. Q., Sun, Y., Reynolds, M., and Atkinson, S. (2021). A vision-based pipeline for vehicle counting, speed estimation, and classification. *IEEE Transactions on Intelligent Transportation Systems*, 22(12):7547–7560. DOI: 10.1109/TITS.2020.3004066.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. (2016). SSD: Single shot multibox detector. In Leibe, B., Matas, J., Sebe, N., and Welling, M., editors, *Computer Vision – ECCV 2016*, pages 21–37, Cham. Springer International Publishing. DOI: 10.1007/978-3-319-46448-0_2.

Ma, H., Qiu, H., Gao, Y., Zhang, Z., Abuadbba, A., Xue, M., Fu, A., Zhang, J., Al-Sarawi, S. F., and Abbott, D. (2023). Quantization backdoors to deep learning commercial frameworks. *IEEE Transactions on Dependable and Secure Computing*, pages 1–18. DOI: 10.1109/TDSC.2023.3271956.

Masalov, A., Ota, J., Corbet, H., Lee, E., and Pelley, A. (2018). CyDet: Improving camera-based cyclist recognition accuracy with known cycling jersey patterns. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 2143–2149. DOI: 10.1109/IVS.2018.8500668.

Mhalla, A., Chateau, T., Gazzah, S., and Amara, N. E. B. (2018). An embedded computer-vision system for multi-object detection in traffic surveillance. *IEEE Transactions on Intelligent Transportation Systems*, 20(11):4006–4018. DOI: 10.1109/TITS.2018.2876614.

Moreira, L. F. R., De F. B. Saar, L. N., Moreira, R., Rodrigues, L. G. F., Travençolo, B. A. N., and Backes, A. R. (2024). Enabling intelligence on edge through an artificial intelligence as a service architecture. In *2024 IEEE 13th International Conference on Cloud Networking (CloudNet)*, pages 1–8. DOI: 10.1109/CloudNet62863.2024.10815777.

Nie, X. and Wang, Y. (2025). Real-time object detection in adverse weather conditions using transformer-based architectures. *International Journal of Engineering and Computer Science*, 14(06):27376–27397. DOI: 10.18535/ijecs.v14i06.5171.

Oja, P., Titze, S., Bauman, A., de Geus, B., Krenn, P., Reger-Nash, B., and Kohlberger, T. (2011). Health benefits of cycling: a systematic review. *Scandinavian Journal of Medicine & Science in Sports*, 21(4):496–509. DOI: 10.1111/j.1600-0838.2011.01299.x.

Ozan, E., Searcy, S., Geiger, B. C., Vaughan, C., Carnes, C., Baird, C., and Hipp, A. (2021). *State of the Art Approaches to Bicycle and Pedestrian Counters*. NCDOT Research. Available at:`https://connect.ncdot.gov/projects/research/RNAProjDocs/RP2020-39%20Final%20Report.pdf`.

Piatkowski, D. and Bopp, M. (2021). Increasing bicycling for transportation: A systematic review of the literature. *Journal of Urban Planning and Development*, 147(2):04021019. DOI: 10.1061/(ASCE)UP.1943-5444.0000693.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788. DOI: 10.1109/CVPR.2016.91.

Ren, S., He, K., Girshick, R., and Sun, J. (2017). Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis amp; Machine Intelligence*, 39(06):1137–1149. DOI: 10.1109/TPAMI.2016.2577031.

Rodrigues Moreira, L. F., Moreira, R., Travençolo, B. A. N., and Backes, A. R. (2025). Deep learning based image classification for embedded devices: A systematic review. *Neurocomputing*, 623:129402. DOI: 10.1016/j.neucom.2025.129402.

Rodriguez-Conde, I., Campos, C., and Fdez-Riverola, F. (2021). On-device object detection for more efficient and privacy-compliant visual perception in context-aware systems. *Applied Sciences*, 11(19). DOI: 10.3390/app11199173.

Stahl, B., Apfelbeck, J., and Lange, R. (2023). Classification of micromobility vehicles in thermal-infrared images based on combined image and contour features using neuromorphic processing. *Applied Sciences*, 13(6). DOI: 10.3390/app13063795.

Tan, M., Pang, R., and Le, Q. V. (2020). EfficientDet: Scalable and efficient object detection. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10778–10787, Los Alamitos, CA, USA. IEEE Computer Society. DOI: 10.1109/CVPR42600.2020.01079.

Zhang, Y., Sun, P., Jiang, Y., Yu, D., Weng, F., Yuan, Z., Luo, P., Liu, W., and Wang, X. (2022). Bytetrack: Multi-object tracking by associating every detection box. In Avidan, S., Brostow, G., Cissé, M., Farinella, G. M., and Hassner, T., editors, *Computer Vision – ECCV 2022*, pages 1–21, Cham. Springer Nature Switzerland. DOI: $10.1007/978\text{-}3\text{-}031\text{-}20047\text{-}2_1$.

Zou, Z., Chen, K., Shi, Z., Guo, Y., and Ye, J. (2023). Object detection in 20 years: A survey. *Proceedings of the IEEE*, 111(3):257–276. DOI: 10.1109/JPROC.2023.3238524.