Journal of Real-Time Image Processing (2023) 20:111
https://doi.org/10.1007/s11554-023-01368-7

RESEARCH

Low-cost system for real-time verification of personal protective equipment in industrial facilities using edge computing devices

Darío G. Lema[1] · Rubén Usamentiaga[1] · Daniel F. García[1]

Received: 18 April 2023 / Accepted: 13 September 2023 / Published online: 9 October 2023

Abstract

Ensure worker safety in the industry is crucial. Despite efforts to improve safety, statistics show a plateau in the reduction of these accidents in recent years. To decrease the number of accidents, compliance with established industrial safety standards and regulations by competent authorities must be ensured, including the use of Personal Protective Equipment (PPE). PPE usage is of paramount importance, as it is essential to prevent accidents from occurring. This work aims to improve worker safety by verifying PPE usage. Technology plays a key role here. A cost-effective solution is proposed to monitor PPE usage in real time. Most existing safety control systems are costly and require considerable maintenance. A low-cost computer vision system is proposed to supervise safety in industrial facilities. This system uses object detection and tracking technology in low-cost embedded devices and can generate alarms in real time if PPE is not used. Unlike other works, temporal information is used to generate the alarms. Safety managers receive this information to take necessary actions. Emphasis has been placed on cost, scalability, and ease of use to facilitate system implementation in industrial plants. The result is an effective system that improves worker safety by verifying established safety measures at a reduced cost. The methodology used improves the Average Precision of PPE detection by 6%. In addition, unlike other studies, the problem of application deployment is addressed, which has an impact on its cost.

Keywords Real-time applications · Low-cost devices · Safety in industry · Safety systems

1 Introduction

Safety standards in industrial environments must be established through industry safety regulations, protocols, procedures, and techniques. For example, international standard IEC 61,508 is one of the most common standards for the design of safety systems. This includes the definition of general requirements for safety, system design, installation, and commissioning. This standard also dictates the need for a risk analysis prior to the implementation of safety systems, to prevent, as far as possible, accidents from occurring. Safety organizations in industrial environments, such as the American National Standard Institute (ANSI) for industrial safety, the International Organization for Standardization (ISO) for safety in industrial control systems, and the Institute of Electrical and Electronics Engineers (IEEE) for safety in industrial automation, have proposed their own standards. These entities create standards that establish requirements for safety in industrial environments, which must be enforced through preventive and follow-up measures, such as equipment maintenance, risk analysis, process monitoring, and adequate personnel training.

One of the objectives of these standards is to ensure the safety of workers in industrial environments. Despite this, in 2019, there were 3408 fatal accidents in the European Union [1], and 5333 in the United States [2]. Although the data show that in recent years, there has been a reduction in fatal workplace accidents [3, 4], there appears to have been a stagnation in their reduction.

Traditionally, in each industrial facility or work area, there is at least one person responsible for ensuring that safety measures are complied with, including the use of PPE [5]. However, due to the large size of industrial facilities, it is difficult to check at all times whether PPE are being used correctly. This is where technology plays a vital role. Analysis of visual information is the key to detect these

✉ Darío G. Lema
gonzalezdario@uniovi.es

[1] Department of Computer Science and Engineering, University of Oviedo, Campus de Viesques, 33204 Gijón, Asturias, Spain

safety failures and, if necessary, generating the corresponding alarms. In the past, it was necessary to implement algorithms to process this information. However, in recent years, there has been a revolution in the field of image processing thanks to deep learning. Deep learning is a machine learning technique based on the use of deep neural networks to process large amounts of data. These deep neural networks are trained to recognize hidden patterns in the data, so they can process images much more accurately and quickly than traditional algorithms. Their main drawback is that the hardware needed is expensive. For this reason, it is important for companies to find affordable computer vision solutions, which can be easily implemented, reducing costs and improving the safety of their workers.

The cloud can be used to process large amounts of data. However, the cost of having a system running continuously is high, and requires constant Internet connectivity. It should also be noted that, due to network latency, it is not possible to process a large number of images in real time (30 FPS). This is why, computer vision solutions that can be run locally, without the need for Internet connectivity, must be sought.

The problem is that recent advances in deep learning-based object detection models, such as residual blocks [6], which greatly increase the accuracy of the models, increase the computational cost. To reduce this cost, modeling and optimization techniques can be used to create more efficient computer vision models, capable of running on low-cost devices. In addition, computational distribution techniques can be used to improve model performance. This enables companies to implement computer vision models efficiently.

Traditionally, the information captured by various video surveillance cameras is sent to and processed by a single central server. This approach clashes with the new trend of creating scalable systems. If new sources of information (cameras) are to be incorporated, it is necessary to acquire a more powerful server.

Fortunately, a new generation of devices capable of performing very fast convolution operations has recently emerged, allowing the application of real-time deep learning-based alarm generation systems at low cost. These new devices are called edge computing devices. The union of these devices with low-cost cameras allows the creation of highly distributed systems where each of these embedded devices contains the vision and detection process. The proposed alternative uses this approach to create a highly scalable autonomous system, increasing the number of points in an industrial facility controlled by the system, at a low cost.

In addition, this solution has a major advantage in terms of information security, as the data generated by the camera are processed in the embedded device, rather than in a centralized system, which means that there is no single point where all the data are stored. This ensures greater data security and privacy, which is a major advantage for many industries.

By integrating computer vision systems based on deep learning and edge computing devices, the novelty of this work lies in the creation of an end-to-end model for alarm generation, contrasting with the previous efforts that merely focused on detecting Personal Protective Equipment (PPE) and individuals. This groundbreaking approach presents a scalable and secure solution for triggering alarms when workers fail to utilize their PPE adequately, thereby significantly enhancing workplace safety.

The utilization of edge computing devices not only ensures scalability but also guarantees privacy. This is achieved by processing camera data directly within the edge computing devices, obviating the need to transmit sensitive information outside the company. Furthermore, the implementation of deep learning systems enables generalized threat detection, eliminating the necessity of fine-tuning parameters for each specific scenario. Consequently, the proposed approach overcomes prior limitations by merging diverse technologies into a cohesive and efficacious system that advocates comprehensive occupational safety.

2 Related works

Numerous studies have explored the real-time processing of video to fulfill the demands of diverse applications [7, 8]. In parallel, research endeavors have aimed to enhance worker safety through the integration of computer vision techniques. However, these efforts predominantly revolve around the detection of specific security equipment, whereas the approach taken in this study transcends these limitations by enabling alarm generation through the use of an end-to-end model in which no post-processing is required.

A pivotal contribution in this area is the construction of an architecture based on YOLOv4 and the Siamese network for personnel tracking in [9] construction environments. The method employs CMOS image sensors to power YOLOv4, but only identifies individuals without discerning whether they are using PPE. Similarly, a fusion of YOLOv5, Openpose, and a one-dimensional convolutional neural network has been utilized in [10] to detect whether workers are wearing their helmets according to the established standards. In [11], another initiative aimed at curbing fall-related accidents from scaffolding collapses combines instance segmentation for scaffolding detection with an object correlation module for hazardous worker behavior identification. Regrettably, their focus does not encompass PPE non-usage detection. These works seek to improve the PPE detection process; however, they do not generate alarms in the event that a worker does not use the established PPE.

An exploration of post-processing techniques for associating detected PPE items with workers has been proposed in [12]. However, the compatibility of such post-processing techniques with real-time applications or their suitability for deployment on edge computing devices remained unexplored. Throughout this paper, a comparative analysis will be conducted between these PPE–worker matching models and the novel end-to-end approach proposed, which directly generates alarms in case a worker does not use the corresponding PPE.

In alternative methodologies, each worker is equipped with a microcontroller-based device for PPE verification [13]. Such devices signal the control room upon detecting non-compliance. Nevertheless, this approach necessitates individualized devices for each worker, escalating system costs. Notably, industrial facilities often house pre-existing video surveillance cameras, which could be repurposed for a computer vision-based system.

To implement a system to help increase worker safety, it is necessary to deploy these systems in some type of device. In [14], a lightweight version of YOLOv5 is developed to detect helmets in construction. With this modified version of YOLOv5, real-time applications are achieved on a NVIDIA Jetson Nano. Unfortunately, the Average Precision decreased by 4.2%. In [15], YOLOv5 is also modified to achieve the same goal: detecting helmets in construction environments. The backbone used is ShuffleNetV2. Also, an optimization is carried out using quantization and layer merging techniques. The results show how these modifications make the model faster than the original. To demonstrate this, they use a NVIDIA Jetson Nano. In [16], traditional techniques, such as LBP classifiers, histogram of oriented gradients, and sequential classifiers, are compared with models based on deep learning. The different solutions are deployed on an Nvidia Jetson TX2 and Jetson Nano. The conclusion is clear: deep learning-based solutions offer better results. While these studies evaluate edge computing device deployment, they fall short of examining the comprehensive real-time monitoring system. Moreover, factors such as image size impact and the potential integration of object trackers to improve system reliability remain unexplored, aspects that this study comprehensively addresses.

3 Materials and methods

The computer vision-based surveillance system uses low-cost cameras to monitor the space under surveillance. Each camera is connected to an edge computing device, where the processing of the captured visual information takes place in real time. Instead of using object detection to determine the presence or absence of personal protective equipment (PPE), object tracking has been implemented in this system.

The main difference between object detection and object tracking is that object detection is limited to identifying the presence or absence of objects in an image, while object tracking goes beyond object detection to provide additional information about the location, movement, and changes in size of each object. While object detection is useful for detecting the presence of PPE, object tracking provides more information about the location and movement of PPE. This makes it possible to check whether PPE is being used in real time. In this way, greater accuracy and efficiency can be obtained in detecting potentially dangerous situations in the monitored space.

In the experiments carried out, it was found that on many occasions, an object is not detected, but it is actually present. For this reason, when using object tracking, the decision to generate an alarm is not made with a single image (frame), but with the information obtained from several frames. In this way, alarm generation is much more reliable. If, for example, in one frame, an occlusion occurs due to two workers crossing paths, so a PPE cannot be detected, the alarm will not be generated, since in the previous N frames, the PPE will have been detected. This provides greater reliability to the algorithm, since false alarms are avoided. In addition, object tracking also makes it possible to track objects entering the security area, keeping track of the workers in each sector, as well as the PPE they are wearing.

The proposal consists of the creation of a low-cost system to verify security measures in real time. The system is composed of the following components: a dataset with a large number of images to allow the creation of a robust model, a dataset of the facility where the system is to be implemented to check its performance, a camera (or several) to monitor the working environment, and a computing device (or several) to process the images captured by the camera. It is also necessary to define the methods to generate alarms and to evaluate the quality of the system. These aspects will be discussed in this section.

3.1 Steps to implement the low-cost system for real-time verification of PPE

Figure 1 shows a diagram of the six steps necessary:

1. Create safety corridors (optional): In some industrial facilities, there may be objects in the middle of the work environment, causing occlusions which makes it difficult to see the workers. This hinders the verification of PPE, not only to computer vision-based systems, but would also hinder verification by human safety officers. For this reason, it is recommended to create these safety corridors, thus favoring the vision process.
2. Locate surveillance spots: The choice of surveillance spots is key. If the correct locations are not selected, the

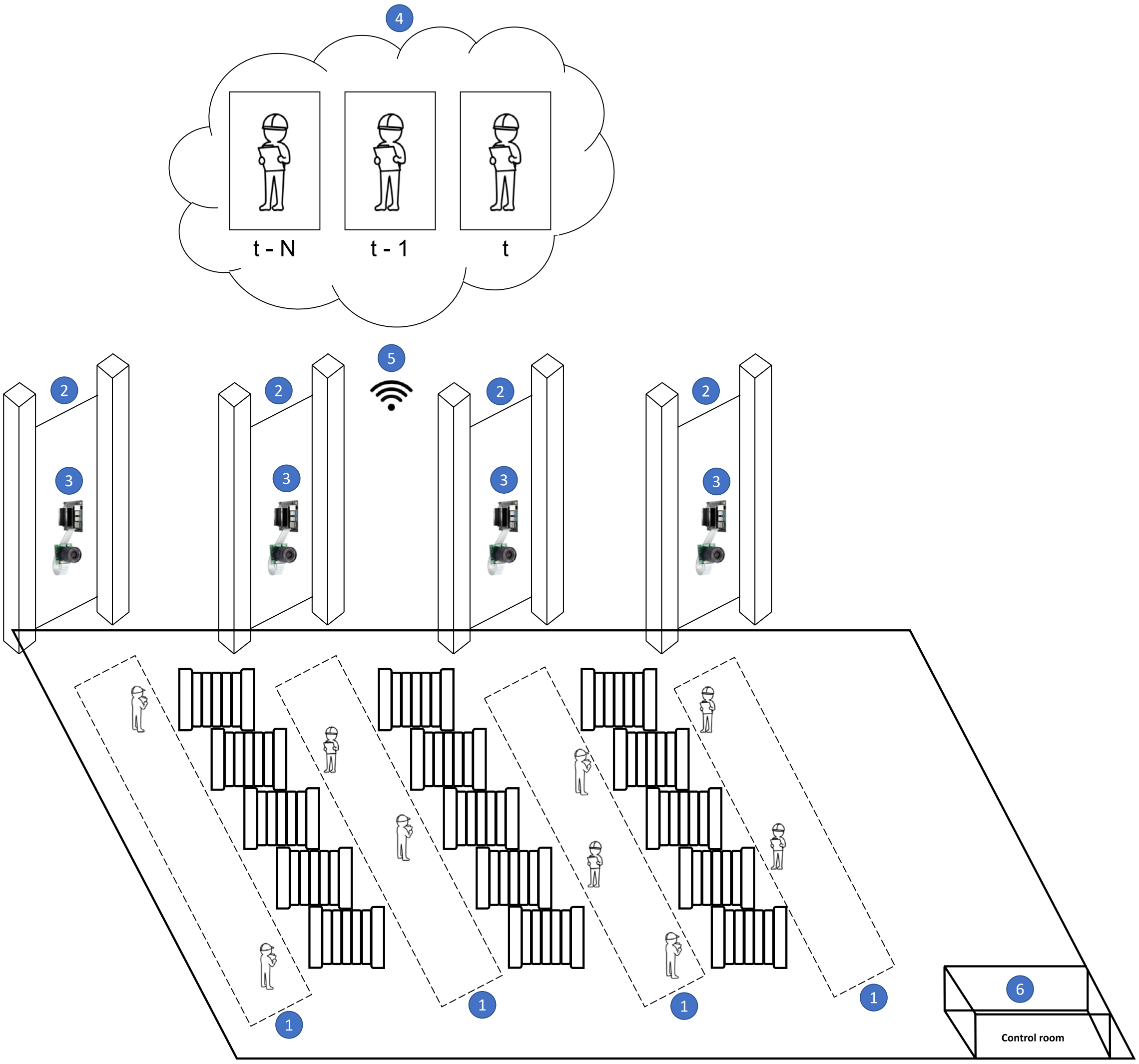


Fig. 1 Diagram that summarize the proposal: Low-cost system for real-time verification of PPE in industrial facilities using edge computing devices

detection will fail and, therefore, the alarms will not be generated correctly. In the facility tested, the existing columns on either side of the work area are used. In Fig. 1, only one of the sides is shown, for the sake of simplicity.

3. Installation of cameras and computing devices: The choice of these devices is key to the correct functioning of the system. They must be low cost and functional. They will be discussed in more detail in this section.
4. Image processing: The cameras placed in the facility will send video images video to the processing device. In these devices, the necessary algorithms must be executed to determine whether an alarm should be generated. Since video is available, it is interesting to study the possibility of using object trackers instead of object detectors, since they provide more information. For example, what happens if in one frame a worker with PPE is detected, but in the previous N frames, it is not. Object detection algorithms cannot make a decision based on this information, but object trackers can. This section will discuss object detection and object tracking algorithms in more detail.
5. Communication of the decision made: In the event that the algorithm used establishes that an alarm should be generated, it is necessary to transmit the information to those responsible for safety. In this project, it has been decided to use the wireless network of the facility where the system is tested. This way, the cost of the system is not affected, since no additional installation is required.
6. Decision-making: Once the recommendation for an alarm has been received, the security manager will decide the appropriate measures.

3.2 Materials

3.2.1 Dataset used

To create the PPE verification system, it is first necessary to generate a model capable of detecting workers and PPE. To train the model, it is necessary to use a suitable dataset. The most appropriate would be to use a dataset with images of the facility in which the system is to be implemented. However, two problems arise: companies do not usually have datasets large enough to generate robust models, and the model generated could not be generalized to other facilities.

For these reasons, it was decided to use a public dataset as a starting point. The selected dataset is Color Helmet and Vest (CHV) [17]. This dataset consists of people, helmets, and vests. The helmets are divided by color (blue, red, white, and yellow) to establish the category of the identified worker. Table 1 shows the number of objects in each class. In this work, it was decided to unify all the helmets in a single class, since the main objective is to detect whether workers are wearing helmets or not.

In Sect. 4.2.1, the results obtained with this dataset are analyzed. Nonetheless, it will be necessary to verify whether the use of these data serves to generalize a model capable of detecting the objects of interest in a particular industrial facility. In Sect. 4.2.3, experiments are carried out to analyze this.

3.2.2 Camera

There are various cameras that could be used to monitor the working environment. Three aspects are key: price, resolution, and viewing angle. To cover a large area of the industrial facility, it is necessary to place several cameras in strategic points of the facility. The larger the facility, the more cameras are needed, so with large installations, the price of the system can skyrocket. Another aspect to take into account is the resolution. Throughout this work, it will be shown how there are alarms that cannot be generated, because images have very low resolution and, therefore, workers cannot even be seen by the human eye. For this reason, it is necessary that the system has the highest resolution cameras possible. Finally, the greater the angle of vision of the selected camera, the wider the area to be covered, thus reducing the number of cameras to be used, and therefore the cost.

Table 1 CHV dataset data

	# of images	% of images	# of objects	% of objects
Person	1323	35.24	3887	42.20
Vest	696	18.54	1784	19.37
Blue Helmet	275	7.32	508	5.51
Red Helmet	269	7.16	536	5.82
White Helmet	536	14.27	1195	12.97
Yellow Helmet	655	17.44	1299	14.10
Total	1330	100	9209	100

Taking these aspects into account, the IMX219-160 camera was selected. This camera costs approximately $20, and is compatible with multiple devices. It has a resolution of 3280 × 2464, and a viewing angle of 160°. In Fig. 2, the camera used is shown.

3.2.3 Computing devices

To process the images captured by the camera, it is necessary to have a device with the appropriate computational capacity. Since one of the objectives is to achieve real-time image processing (30 FPS), one option would be to use traditional GPUs. The main advantage of these GPUs is that they are able to perform the convolution operations, necessary to apply the object detection algorithms, very efficiently. However, their price does not make them suitable for systems where scalability is sought through a distributed and independent system. Fortunately, in recent years, a series of devices, known as edge computing devices, have emerged, which offer high performance at a low cost.

Within this range of products, there are devices with high performance, such as the NVIDIA AGX Xavier, whose price is approximately $600. However, again, due to its high cost, this type of device has been discarded in favor of the Raspberry Pi v4, perhaps the most widely used device, and the NVIDIA Jetson Nano. Both devices have an approximate

Fig. 2 IMX219-160 camera

price of $50. These devices have been used in other works with excellent results [18–20]. In Fig. 3, both devices are shown. Sections 4.4.1 and 4.4.2 analyze the inference times of the two devices using the selected object detection algorithm and varying the size of the images used.

3.3 Methods

3.3.1 Alarm generation

The decision to generate an alarm can be made based on information from a specific moment, or using temporal information about the same event. In both cases, it is necessary to detect the objects of interest, and then make a decision. For this reason, object detection and object tracking algorithms are analyzed.

Object detection algorithms: Non-compliance with safety measures can be detected by applying an object detection algorithm. These algorithms have existed for decades, but in recent years, there has been a revolution in this field. This is mainly due to the popularization of GPUs, which allows convolution operations to be performed efficiently. Thanks to these advances, a wide variety of object detection algorithms based on deep learning have been developed. These algorithms are divided into two types: one-stage and two-stage detectors. The two-stage detectors first propose a set of regions of interest (ROIs) and then classify these regions into categories. RCNN [21] is an example of two-stage detector. One-stage detectors process the entire image at once. YOLO or SSD [22] are examples of one-stage detectors.

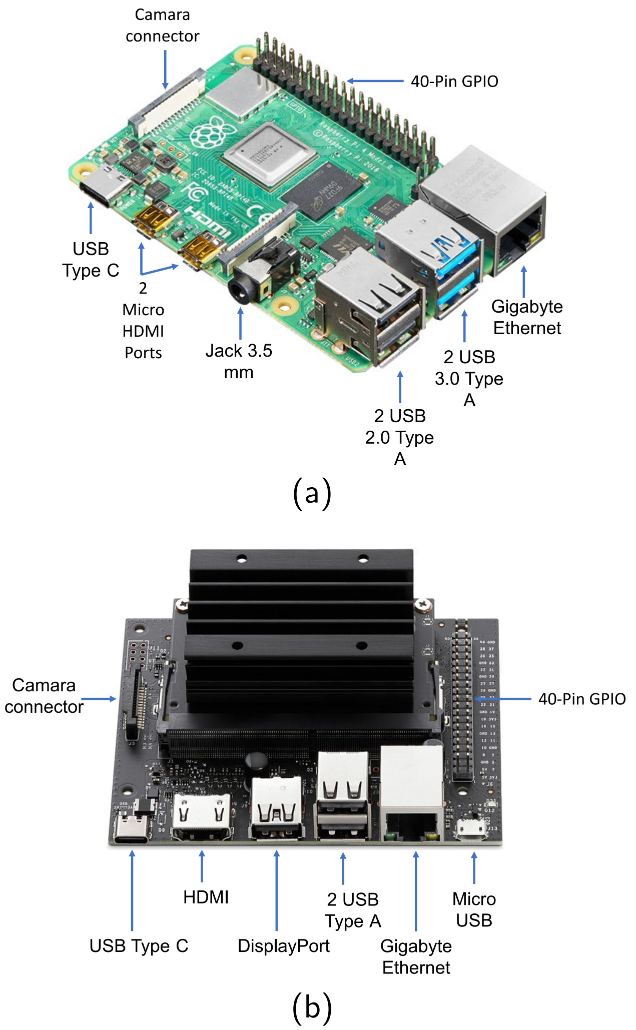


Fig. 3 Edge computing devices: (a) Raspberry Pi v4. (b) NVIDIA Jetson Nano

Because real-time processing is key to the proposed system, one-stage detectors are chosen. After analyzing several works where these detectors are used, and due to its evolution, YOLOv5 was chosen to detect objects of interest.

Table 2 shows a comparison between the different versions of YOLO. The first version of YOLO [23] divides the image to be processed into an $S \times S$ grid. Each of the cells to be processed is responsible for detecting the objects whose center falls in it. YOLOv2 [24] incorporates improvements such as anchor boxes that make it competitive with other algorithms in terms of accuracy, but not in the detection of small objects. For this reason, YOLOv3 [25] incorporates the detection of objects in three scales. In YOLOv4 and YOLOv5 [26, 27], several improvements such as mosaic augmentation are incorporated into the training. These improvements give the algorithm an accuracy that was unthinkable a few years before. Recently, the sixth and seventh versions of the algorithm [28, 29] have been developed and modifications have been made to the network architecture.

Since the last two versions are so new, they have not yet been widely evaluated by the scientific community. For this reason, it was decided to use YOLOv5 as the algorithm to detect the objects of interest.

Table 2 Comparison of the different versions of YOLO

Version	Year	Grid	Multiple scale detection	Anchor boxes	Strong augmentation policy
YOLOv1	2016	7 × 7	No	0	No
YOLOv2	2017	13 × 13	No	5	No
YOLOv3	2018	13 × 13 26 × 26 52 × 52	Yes	9	No
YOLOv4	2020	13 × 13 26 × 26 52 × 52	Yes	9	Yes
YOLOv5	2020	13 × 13 26 × 26 52 × 52	Yes	9	Yes

Once the objects of interest, PPE in this case, have been detected, a decision must be made. To do this, two alternatives, called the logical model and the end-to-end model, have been studied. In both cases, it is necessary to have the best possible detection model available, so that the best decision can be made. Section 4.2.2 describes several experiments carried out with the aim of improving previous models.

Logical model: After obtaining the optimal model, the system proceeds to generate alarms based on the identified detections. The alarms are categorized into two types: worker without helmet and worker without vest.

The employed object detection algorithm is capable of detecting both workers and Personal Protective Equipment (PPE) independently. However, the challenge arises in associating each worker with the appropriate PPE. To address this issue, the Intersection over Union (IOU) metric could be utilized. IOU quantifies the extent of overlap between two regions, which are predictions in the context of this application. Nevertheless, due to the considerable size disparity between a worker and a piece of PPE, the IOU value might turn out to be quite low.

To mitigate this size-related limitation, an alternative approach known as Intersection over Class (IOC) is proposed, as shown in Eq. 1

$$IOC = \frac{(Person \cap PPE)_{area}}{PPE_{area}}. \quad (1)$$

To assign PPE to a specific worker, IOC calculations are performed between all workers and the particular PPE in question. The worker–PPE association is established by linking the PPE to the worker with the highest IOC(Person, PPE) value, provided that the IOC surpasses a predetermined threshold. This method ensures that each PPE is allocated only to the nearest worker.

When a worker is detected without the required PPE, an alarm is triggered. To determine the minimal threshold for PPE compliance, a comprehensive analysis of the available data is essential. The relevant investigation to establish this threshold is conducted in Sect. 4.3.1.

End-to-end model: An alternative approach to generate alarms involves the creation of an end-to-end model that directly predicts the alarms. However, this approach comes with its own set of challenges, primarily due to the scarcity of public datasets annotated for this specific task. Unlike other solutions, this method diverges from the conventional labeling of people, helmets, and vests. Instead, it focuses on labeling instances of interest: workers without helmets, workers without vests, and workers wearing both PPE components. A visual representation of this annotation approach is illustrated in Fig. 4b.

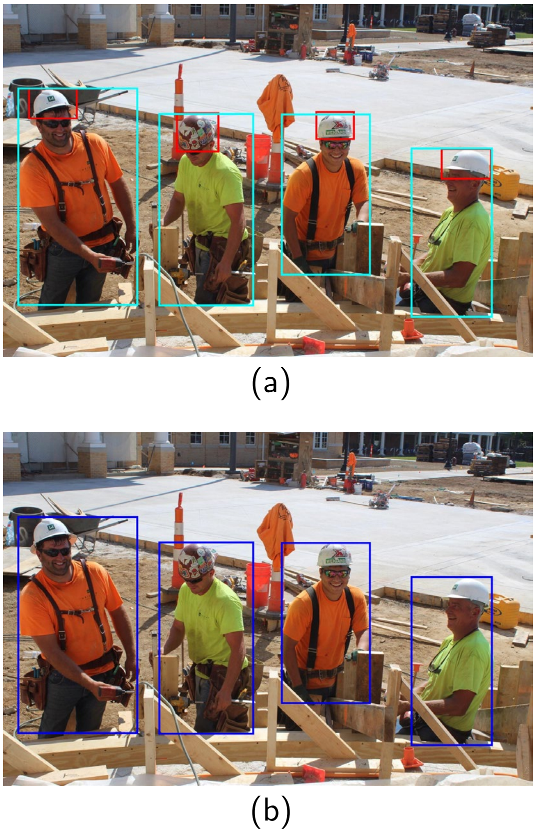


Fig. 4 Labeling of the dataset according to the model to be applied: **a** Logical model: labeling of each of the objects of interest independently. **b** End-to-end model. In both cases, an alarm should be generated for lack of vest

In contrast to the logical model, where individual objects are annotated separately (as shown in Fig. 4a), this end-to-end approach directly annotates alarms. This means that alarms themselves are designated and identified within the dataset. Furthermore, to facilitate the model's ability to distinguish instances that warrant generating an alarm from those that do not, it becomes imperative to include labeling for individuals who diligently adhere to all safety measures-those wearing both helmets and vests.

In Sect. 4.3.2, the feasibility of implementing this end-to-end approach is explored. This entails meticulous adjustments to the data labeling process to accommodate the model's learning needs effectively.

In Sect. 4.3.3, a comparative analysis delves into the outcomes achieved by both the logical model and the end-to-end prediction model. Not only are the results scrutinized, but also a comprehensive examination of the merits and drawbacks of each approach is presented in detail. This comprehensive evaluation seeks to show the trade-offs between the two alternatives and provide insights into their respective performances within the context of the task at hand.

Object tracking algorithms: The main difference between an object tracking algorithm and a detection algorithm is that the latter are limited to detecting objects of interest in an image (or frame). However, the former have the ability to track the object of interest across multiple frames, adjusting a region of interest to follow the object as it moves. These tracking algorithms use an object detection model to locate the objects of interest in each frame. Once the objects are in a frame, they are associated with the objects in the previous frames. In this way, an object in a scene can be tracked. This characteristic is of vital importance for the development of the proposed system, since when a PPE is not detected in one frame but in the previous frames it is, the alarm should not be generated. This is very common in an industrial facility.

The most widely known object tracking algorithm is Simple Online and Real-time Tracking (SORT) [30]. SORT is a real-time object tracking algorithm. It is based on detecting and tracking objects in an image or video sequence using a neural network to detect and classify objects in the scene. Once the objects have been detected, SORT uses a Kalman filter as a label assignment algorithm to assign a unique identity to each object and track its movement over time.

The evolution of SORT is DeepSORT [31]. Like SORT, DeepSORT uses a neural network to detect and classify objects in an image or video sequence. However, instead of using a Kalman label assignment algorithm to assign unique identities to each object, DeepSORT uses another neural network to learn and remember the unique features of each object and improve tracking accuracy over time.

Although DeepSORT improves on the results of SORT considerably, a new version of the algorithm, known as StrongSORT [32], has recently emerged. StrongSORT incorporates a link model without appearance information (AFLink) to associate short tracks into complete trajectories; and a Gaussian-smoothed interpolation (GSI) to compensate for missing detections. These two modifications with respect to DeepSORT make the object tracker metrics better. For this reason, StrongSORT was selected as the object tracking algorithm to develop the system.

Object tracking works with temporal information. For this reason, it is very useful for this work, as there may be occlusions that prevent workers from being visible at all times. For example, if two workers cross each other, noise will be added to the data to be processed. For this reason, using temporal information is very useful. In Sect. 4.3.4, the results of applying StrongSORT in a particular industrial facility are analyzed.

3.3.2 Evaluation metrics

To establish the performance of the system, it is necessary to establish a series of metrics. The most commonly used metric in the field of object detection is Average Precision (AP) [33]. This metric is based on precision and recall. Precision, shown in Eq. 2, is used to measure how trustworthy the predictions are. Recall, shown in Eq. 3, is used to check the percentage of objects that have been detected. Both metrics can be combined in F1, as shown in Eq. 4.

In object detection, the predictions are accompanied by a confidence value. To calculate the AP, first, precision and recall are calculated by filtering by confidence level, and then, the curve is plotted for all levels. The AP is the area enclosed by the curve

$$Precision = \frac{True\ positives}{True\ positives + False\ positives} \tag{2}$$

$$Recall = \frac{True\ positives}{True\ positives + False\ negatives} \tag{3}$$

$$F_1 = \frac{2 \times Precision \times Recall}{Precision + Recall}. \tag{4}$$

With the AP, it is possible to compare different object detectors. The choice of the best object detector is key as it will affect the object tracker metrics. In the case of object tracking, the most widespread metric is Multiple Object Tracking Accuracy (MOTA) [34]. Object tracking takes into account two aspects: the correct detection of the objects of interest, and their correct identification over time. MOTA has these two aspects in mind. For this reason, as shown in Eq. 5, False Negatives (FN), False Positives (FP), Identifier Switches (IDSW), and the number of ground truth objects (GT) are identified for every frame (t). Commonly, this metric is used to evaluate pedestrian tracking in scenes with multiple objects. In this work, what is interesting is to evaluate the tracking of the alarms of each of the PPE, as well as of the workers that are complying with the established safety measures. For this reason, the MOTA is calculated for each of the established classes

$$MOTA = 1 - \frac{\sum_t (FN_t + FP_t + IDSW_t)}{\sum_t GT_t}. \tag{5}$$

4 Experimental setup and results

To implement the real-time PPE verification system, a series of experiments are done to select the best option. The results of these experiments are discussed in this section.

4.1 Application scenario

The safety verification system developed has been tested in a real industrial facility. The final results are shown in

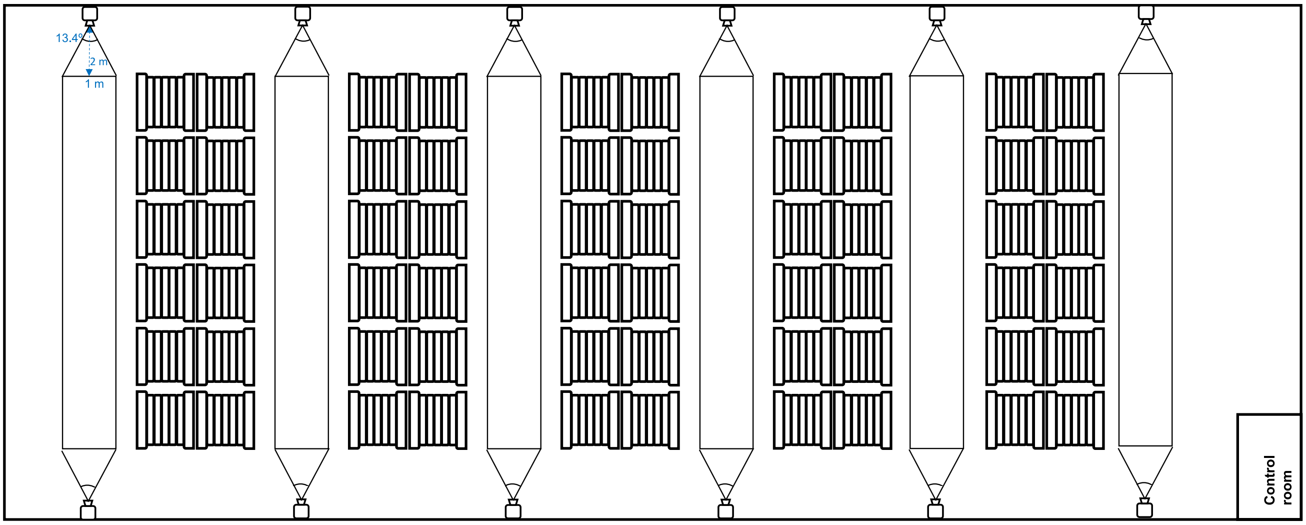


Fig. 5 Diagram showing the layout of the coils and control room used in the industrial facility used to test the feasibility of the PPE verification system

Table 3 Scenarios where the size of industrial facilities is varied

# of corridors	# of Cameras	# of NVIDIA Jetson Nanos	Cost ($)	Power consumption (kW)
10	20	20	1,400.00	0.24
50	100	100	7,000.00	1.2
100	200	200	14,000.00	2.4
200	400	400	28,000.00	4.8

Table 4 Original results (AP) of the CHV dataset with YOLOv5

Model	Person	Helmet	Vest	Average
YOLOv5S	0.8296	0.8411	0.7656	0.8121
YOLOv5M	0.8305	0.8278	0.8214	0.8265
YOLOv5L	0.8311	0.8752	0.8364	0.8475
YOLOv5X	0.8377	0.8851	0.8147	0.8458

Sect. 4.3.4. This facility stores steel coils used for the manufacture of products, such as automobiles, machinery, and household appliances. In the warehouse, the coils are stored in rows, as shown in Fig. 5. This arrangement causes corridors to form between the coils. As the coils are stacked on top of each other, occlusions are generated from one corridor to another. A control room is located in the facility itself, where the alarms generated by the system are controlled.

Thus, to guarantee the safety of the workers at all times, two cameras were placed in each corridor, one at each end. Each camera must cover a viewing angle of 13.4°, since they are located 2 ms away from the beginning of the corridor.

The facility has 37 safety corridors. Following the proposed design, 74 cameras and 74 NVIDIA Jetson Nano devices are required, at a total cost of $5,180.00. Each device (NVIDIA Jetson Nano and camera) has a power consumption of 7.5 watts, so the system consumption is 1.08 kW. Unlike traditional centralized systems, the proposal is highly scalable and the number of devices required can be scaled up or down as needed. Table 3 shows some scenarios in which the cost and consumption for different requirements are calculated. Obviously, the larger the facility, the greater the number of devices required, with the corresponding increase in cost and power consumption.

4.2 Detection results

4.2.1 Object detection with CHV

In the CHV dataset, analyzed in Sect. 3.2.1, YOLOv5 is used to detect the classes of interest. The results obtained are shown in Table 4. However, as part of the research, it was decided to further explore the hyperparametric configuration of YOLOv5 using a technique known as hybrid search. This technique starts with a base configuration, and then, a hyperparameter is varied. If the resulting model is better than the previous one, it becomes the new base. After extensive experimentation, the results shown in Table 5 were obtained. The training configuration used was: 600 epochs, a batch size of 8, and a learning rate varying from 0.01 to 0.002 using Adam as solver. It seems clear that after extensive hyperparametric tuning, the results obtained improve on the original results. For example, if the L models which give the best results are compared for the person class, there is an improvement in AP of 4% and 6% for the helmet and vest classes. These improvements in the detection of the

Table 5 Results (AP) of the CHV dataset with YOLOv5 after hyper-parametric adjustment

Model	Person	Helmet	Vest	Average
YOLOv5S	0.8305	0.9253	0.8628	0.8728
YOLOv5M	0.8622	0.9255	0.9013	0.8963
YOLOv5L	0.8708	0.9366	0.8974	0.9016
YOLOv5X	0.8714	0.9107	0.8725	0.8848

individual classes directly influence the alarm generation, since this is based on the detection of each of the classes.

Figure 6 shows some examples of people, helmet, and vest detection with YOLOv5L. Figure 6a is a clear example of how labeling a dataset can negatively affect the models. The worker on the left is wearing the regulation vest. However, the vest is not labeled. Three workers are labeled on the tower in the background. Since they are so far in the background of the image, they are almost imperceptible to the human eye, and therefore, the model cannot detect them, as shown in Fig. 6b. Figure 6c shows a number of people in the vicinity of an industrial facility. Some of these people are wearing PPE, while others are not. Figure 6d shows how practically all the people and the PPE can be detected, except for the one cut off on the left. Figure 6f and 6h are examples of how the detection of people with and without PPE is feasible when there are no crowds. Figure 6i is a clear example of conditions in which PPE cannot be detected due to the high concentration of people. However, in Fig. 6j, almost all people and helmets are detected, showing the robustness of the trained model. Finally, Fig. 6k shows a person with helmet and vest. In Fig. 6l, all three classes are perfectly detected, although one false positive of the helmet class is generated.

4.2.2 Transfer learning

The model obtained in the previous section is better than the one obtained in [17]. However, as explained in Sect. 3.3.1, the alarm generation is based on the detection of workers and PPE, so the more accurate the detection is, the more reliable the system will be.

Transfer learning is applied to improve the model. Transfer learning consists of using the weights of an already trained model. In this way, predictions can be made without the need for training. The main problem is to find a model that has been trained for the same classes of interest. In the literature, there is no model trained to detect people, helmets, and vests, but there are models for detecting people. Of all the public datasets in which the person class is found, COCO [35] is one of the most extensive with more than half a million images. For this reason, several of the models

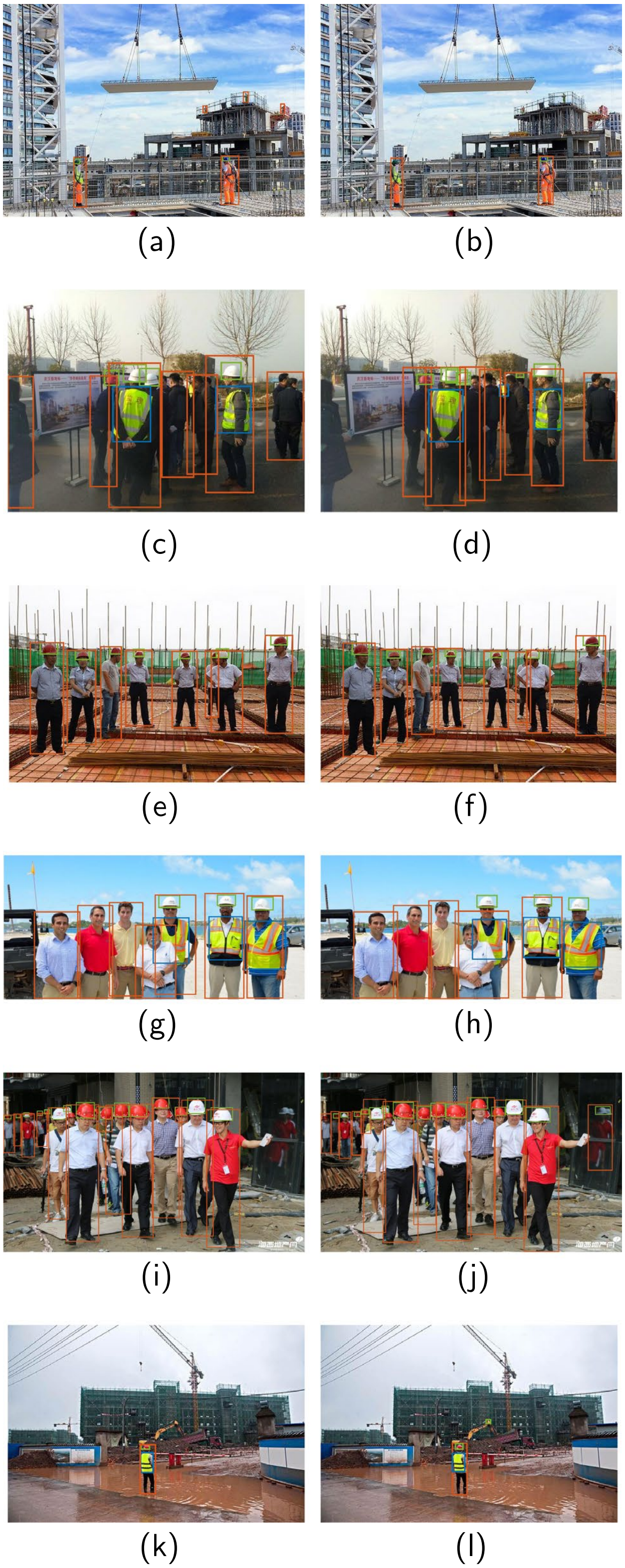


Fig. 6 Examples of test images from the CHV dataset. In the left column is the ground truth. On the right the detections made with YOLOv5L. Detections of people are shown in orange, the helmets in green, and the vests in blue

Table 6 Results of the CHV dataset with YOLOv5 and COCO weights for person class

Model	Precision	Recall	AP
YOLOv5N	0.868	0.758	0.835
YOLOv5S	0.902	0.796	0.854
YOLOv5M	0.882	0.862	0.894
YOLOv5L	0.919	0.829	0.895
YOLOv5X	0.922	0.844	0.844

Table 7 Results of the CHV dataset after fine-tuning with YOLOv5M

	Precision	Recall	AP
Person	0.8974	0.8377	0.9033
Helmet	0.9234	0.9184	0.9551
Vest	0.9038	0.8659	0.9026
Average	0.9082	0.8741	0.9204

trained for this dataset are applied to CHV. Table 6 shows the results obtained.

The main disadvantage of transfer learning is that classes that have not been used in the training of the pre-trained model cannot be detected. To solve this problem, the model is fine tuned. Fine tuning consists of a short training with the target dataset, with a low learning rate to refine the metrics obtained. In this case, the model is refined with YOLOv5M, as shown in Table 6. Although the YOLOv5L model offers slightly better results, this model is more complex and, therefore, has a longer inference time. It is important to note that with fine tuning the training starts with the pre-trained weights, but during training, it is also possible to detect the helmet and vest classes.

After performing fine-tuning with a learning rate of 0.0032 during 200 epochs, the results shown in Table 7 are obtained. This technique not only improves the results obtained, but the model is trained in a shorter time.

4.2.3 Generalization of the model to other datasets

The generated model has been created using public data. The advantage is that it is a large dataset with which a robust model can be created. However, the objective is to implement the system in a real facility. To test the performance of the system, a series of images of the facility where the system will be put into operation are collected and the model is evaluated. Table 8 shows the results. The results obtained are clearly worse than those obtained with the public CHV dataset. This is due to the fact that the images of the real industrial facility do not belong to the dataset used, and therefore, the characteristics of the objects are different. To improve the results obtained in the real industrial facility, a new training is performed. This time, the training starts with from the weights of the previous model and incorporates some images from the real industrial facility. As the objective is to refine the model to fit the industrial facility in question, an initial learning rate of 0.0032 is used, decreasing it in each of the 200 epochs up to 0.000384. Table 9 shows the results obtained. With this new training, the AP improves by 9.6% for the person class, 24% for the vest class, and 1.2% for the helmet class.

Table 8 Results of applying the best model obtained to the new dataset of a particular industrial facility

	Precision	Recall	F_1	AP
Person	0.842	0.691	0.759	0.742
Helmet	0.851	0.830	0.840	0.839
Vest	0.781	0.423	0.548	0.548

Table 9 Results of applying the best model obtained to the new dataset of a particular industrial facility after tuning the model

	Precision	Recall	F_1	AP
Person	0.8666	0.6963	0.8269	0.8387
Helmet	0.8666	0.6963	0.8575	0.8507
Vest	0.9315	0.7945	0.7722	0.7973

These results show that using a public dataset with many examples to generate a model is a good idea, but to have optimal results, it is necessary to refine the model with examples of the facility in which it is to be applied.

4.3 Alarm generation

Once the objects of interest are detected, a decision must be made to determine whether or not an alarm should be generated. To do this, two approaches are proposed: the logical model and the end-to-end model. In Sect. 3.3.1, the differences between both models are explained.

4.3.1 Logical model

To use this model, the first step is to establish a minimum IOC compliance threshold. To do this, data are analyzed. Figure 7 shows the IOC between PPE and workers. The vast majority of workers have PPE in close proximity (high IOC value), but there are many who have PPE but not in close proximity (low IOC). The worker shown on the right of Fig. 8 has a low IOC, because the degree of overlap between the detection of the person and the helmet is not very high. For cases like this, it is decided to set the $IOC = 0$. This means that to assign a PPE to a worker, it is sufficient that both detections overlap. Even so, in the event that a PPE

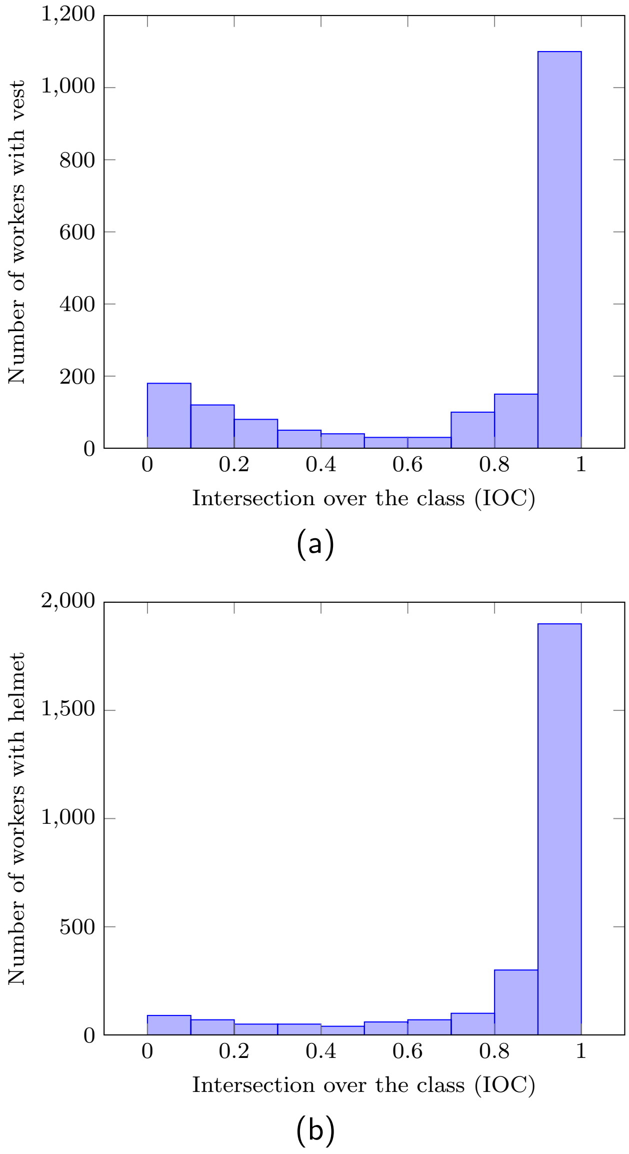


Fig. 7 Intersection over the class (IOC) between the workers and the PPE

overlaps with two or more workers, it will be assigned to the worker with whom it overlaps the most.

Once the IOC threshold is established, the algorithm is applied. Table 10 shows the results obtained. It seems that the detection of workers not wearing a helmet is not very accurate. This is mainly due to the low number of examples of this type of alarms, and the low size of the helmets.

In the original CHV dataset work, four types of helmets are distinguished according to color. This is because the category of the worker depends on the color of the helmet. To detect them, they directly apply YOLOv5. The logical model proposed allows the classification of helmets by applying an image classifier, since it detects them independently. After performing multiple experiments with the Efficienetv2 classifier, the results shown in Table 11 are obtained. Observing the results, it seems clear that it is possible to generate alarms when a worker does not use the helmet, and to classify by categories the workers who are using them.

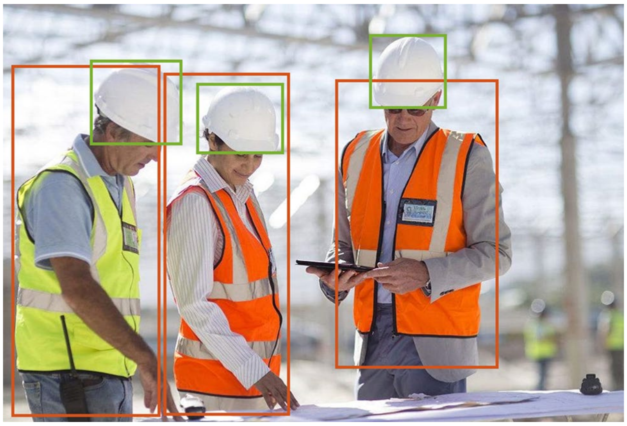


Fig. 8 Example of low IOC between a worker and a helmet. Detections of people are shown in orange, and the helmets in green

4.3.2 End-to-end model

The end-to-end model has a radically different approach from the logical model, because alarms are detected directly (the dataset labels are the alarms for not wearing a helmet and for not wearing a vest; and the workers who use both. See Sect. 3.3.1). After performing various experiments, the results shown in Table 12 are obtained. The end-to-end model slightly improves the results obtained by the logical model.

Figure 9 shows a series of examples where the corresponding alarms are generated using the end-to-end model. Figure 9a shows a worker using both PPE in green. However, a worker who is not wearing a helmet is shown in yellow. As can be seen in this figure, this is a clear example of a false positive, since he is actually wearing a helmet. It should also be noted that workers in the tower in the background are not detected due to lack of resolution, which negatively affects the metrics obtained. In Fig. 9b, two workers who generate alarms for not using the vest are shown in red. Figure 9c–e show workers who are using their PPE. Figure 9f shows several workers who are not using the required vest.

4.3.3 Comparison of models

Both options are valid for generating alarms. However, both have advantages and disadvantages. The logical model offers the possibility of detecting the PPE individually, being able to add new PPE to the alarm generation algorithm if necessary. In addition, there are multiple public datasets with which to train the models. Its major drawback is the need for

Table 10 Results of alarm generation with the logical model: alarms for not wearing a helmet (no helmet alarm) and for not wearing a vest (no vest alarm); and workers who use both (no alarm)

	# of objects	TP	FP	FN	P	R	F_1
No vest alarm	235	171	59	64	0.7435	0.7277	0.7355
No helmet alarm	58	30	27	28	0.5263	0.5172	0.5217
No alarm	207	157	29	50	0.8441	0.7585	0.7990

Table 11 Results of the alarms generation with the end-to-end model

Helmet	Precision	Recall	F_1
Blue	0.9091	0.8889	0.8989
Red	0.9600	0.8889	0.9231
White	0.8977	0.9875	0.9405
Yellow	0.9592	0.8924	0.9246
Mean	0.9315	0.9144	0.9218

Table 12 Results of the alarms generation with the end-to-end model: alarms for not wearing a helmet (no helmet alarm) and for not wearing a vest (no vest alarm); and workers who use both (no alarm)

	Precision	Recall	F_1	AP
No vest alarm	0.8175	0.6809	0.7430	0.7466
No helmet alarm	0.7271	0.4134	0.5271	0.4276
No alarm	0.8092	0.8599	0.8338	0.8835

post-processing. The inference time is 10 ms (NVIDIA RTX 2080 Ti); however, the detections must be post processed to generate the alarms, which adds 150 ms (Intel i7 9700K) to the total time. To classify workers according to category, an image classifier can be applied using the detected helmets as input, adding another 10 ms. In Table 13, the results of applying EfficientNet v2 are shown. The end-to-end model does not have this disadvantage, since it generates the alarms directly. Its major drawbacks are that it cannot detect individual PPE and the lack of public datasets.

Since one of the objectives of this work is to be able to verify PPE in real time, the end-to-end model is chosen, as it does not require post-processing.

4.3.4 Object tracking for alarm generation

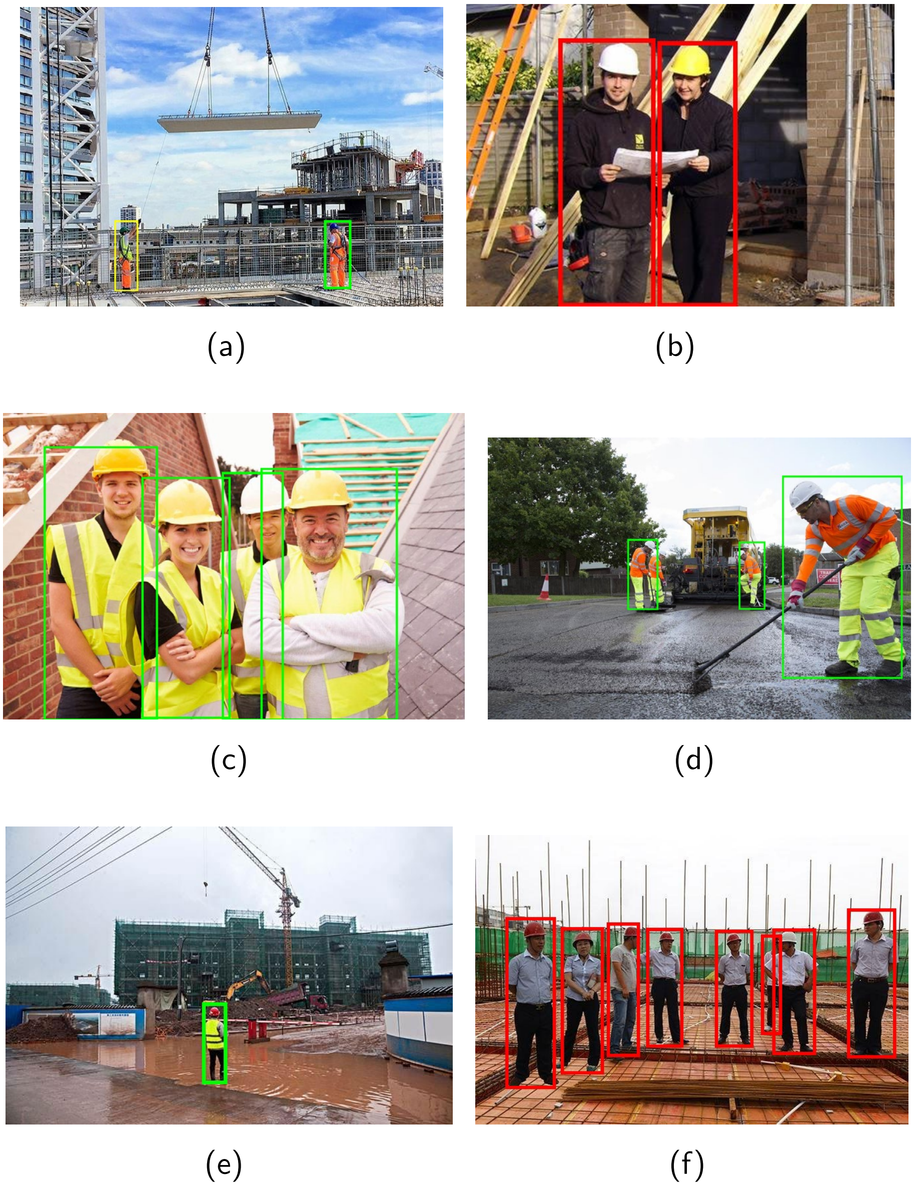
Alarm generation is possible, although for a robust system, it is not sufficient, because too many false positives (FP) are generated, leading the system to generate invalid alarms. For this reason, it is decided to make use of an object tracker, which provides temporal information. With this information, the number of FPs can be reduced considerably.

Figure 10 shows an example of this problem. Of the six frames shown, only in the third one is the worker's vest not detected. If object detection were used to generate the alarms, in this example an alarm, should be generated. However, no human would generate the alarm, because in the remaining five frames, the vest is detected. This is where object tracking plays a crucial role in preventing false alarms. StrongSORT, based on the use of object detectors, was selected as object tracker. In this case, the selected object detector is the YOLOv5 model obtained in Sect. 4.4.2. Three parameters must be configured for the tracker:

- Max_Age: maximum number of frames with which an undetected object is discarded. E.g., if Max_Age is 5, and an object is not detected for 6 frames in a row, the next time it is detected it is be considered as a different object.
- N_Init: number of frames in a row in which an object has to be detected to be considered.
- NN_Budget: to track an object, it is necessary to calculate the distance between the object in the current and previous frames. NN_Budget is the number of frames used to calculate this distance.

As this work is designed to generate alarms when a worker does not use the appropriate PPE, it is necessary to track workers. As the speed at which workers move is not very high, the parameter NN_Budget remains constant. However, MOTA (the most widespread metric for analyzing the accuracy of object trackers) does vary depending on the values assigned to Max_Age and N_Init. Tables 14, 15, and 16 show the MOTA obtained for each of the object classes to be tracked. It seems clear that if an N_Init of 1 is used, which would be equivalent to object detection, the MOTA for all three classes is very low. However, when at least three frames are used to make the decision to generate the alarm, the results improve considerably. Something similar happens with the Max_Age. If a very low value is set, when the corresponding alarm is not detected in one frame, the identifier of that object is discarded. For this reason, if it is detected again in the next frame, another identifier will be assigned to it. As it is really the same object, the tracking would not be correct, affecting the MOTA negatively.

Fig. 9 Examples of alarms detection. Alarm detections for not wearing a vest are shown in orange, alarm detections for not wearing a helmet in yellow, and workers with complete PPE in green



(a) (b)

(c) (d)

(e) (f)

Table 13 Results of EfficientNet v2 to classify helmets by color

Helmet	Precision	Recall	F_1
Blue	0.9091	0.8889	0.8989
Red	0.9600	0.8889	0.9231
White	0.8977	0.9875	0.9405
Yellow	0.9592	0.8924	0.9246
Mean	0.9315	0.9144	0.9218

4.4 Devices and deployment

4.4.1 Selection of the device to be used to detect alarms

For the system to improve worker safety, it is necessary that the model is accurate, and also that it performs the inference as fast as possible. The accuracy of the model is independent of the device, so to be able to select which one is more suitable to the needs, the speed of inference is compared. For this reason, a series of experiments were carried out in which YOLOv5 is tested on both devices using an input size of 704. With the Raspberry Pi v4, Pytorch has been used as a framework, since it is used by default. However, with the NVIDIA Jetson Nano, TensorRT is used. TensorRT is a framework developed by NVIDIA that accelerates convolution operations. Table 17 shows the results. Clearly, the fact that the NVIDIA Jetson Nano incorporates a GPU makes the inference process much faster. For this reason, this device is used in the alarm generation system.

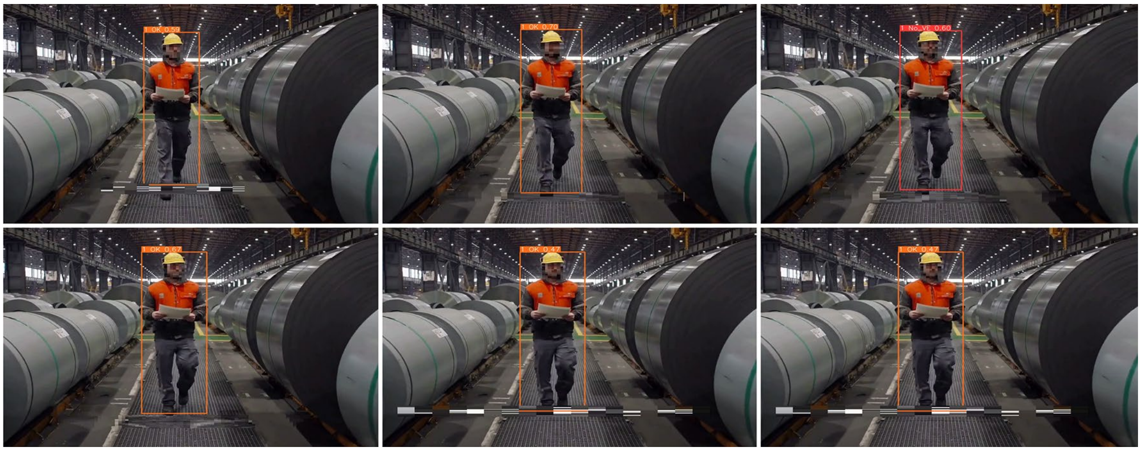


Fig. 10 Example of how an alarm would be generated in an isolated frame, but not as a whole

Table 14 MOTA for alarm due to lack of vest

N_Init	Max_Age						
	1	2	3	4	5	6	7
1	38.42	42.28	48.36	50.28	50.28	50.28	50.28
2	42.25	45.38	60.25	63.84	63.84	63.84	63.84
3	67.28	69.23	72.33	75.66	75.66	75.66	75.66
4	67.35	69.77	72.55	75.89	75.89	75.89	75.89
5	67.42	69.76	72.26	75.91	75.91	75.91	75.91
6	67.55	69.62	72.48	75.90	75.90	75.90	75.90
7	67.54	69.63	72.48	75.88	75.88	75.88	75.88

Table 15 MOTA for alarm due to lack of helmet

N_Init	Max_Age						
	1	2	3	4	5	6	7
1	30.25	32.14	38.25	40.68	40.68	40.68	40.68
2	34.28	42.34	50.34	52.24	52.24	52.24	52.24
3	42.82	53.47	59.14	61.35	61.35	61.35	61.35
4	42.91	53.61	59.82	62.55	62.55	62.55	62.55
5	42.92	53.63	59.84	62.58	62.58	62.58	62.58
6	42.90	53.62	59.83	62.54	62.54	62.54	62.54
7	42.87	53.59	59.80	62.47	62.47	62.47	62.47

Table 16 MOTA for no alarms

N_Init	Max_Age						
	1	2	3	4	5	6	7
1	62.17	64.28	71.47	75.26	75.26	75.26	75.26
2	65.33	74.84	78.25	81.56	81.56	81.56	81.56
3	74.50	81.56	87.59	89.34	89.34	89.34	89.34
4	74.52	81.58	87.63	89.78	89.78	89.78	89.78
5	74.53	81.60	87.61	89.75	89.75	89.75	89.75
6	74.55	81.61	87.60	89.66	89.66	89.66	89.66
7	74.56	81.62	87.60	89.61	89.61	89.61	89.61

Table 17 Inference times with YOLOv5 and an input size of 704 on Rapsberry PI v4 and NVIDIA Jetson Nano

	Raspberry Pi v4 Inference Time (ms)	NVIDIA Jetson Nano Inference Time (ms)
YOLOv5N	4818	65
YOLOv5S	18,008	108
YOLOv5M	54,001	226
YOLOv5L	138,531	408
YOLOv5X	256,307	Out of memory

4.4.2 Matching the model to the device

The choice of the NVIDIA Jetson Nano seems obvious, as its performance for this type of system is more suitable than the Raspberry PI v4. However, the input size used during the experiments, 704, is too large to reach real time. The selected camera reaches 30 FPS, so the device should be able to process images at this speed. For this reason, several tests are performed in which the input sizes of YOLOv5 are varied and the resulting AP is calculated. Table 18 shows the results. All times were measured ten times, and averaged. YOLOv5M is discarded due to the fact that regardless of the input size, it does not approach 30 FPS in any case. YOLOv5S provides adequate processing speed on the NVDIA Jetson Nano with input sizes below 384, Although that it suffers an AP loss of 6% with respect to that obtained for an input size of 704. With YOLOv5N, the required FPS is obtained with input sizes smaller than 512. However, a much lower AP is obtained than with YOLOv5S models that also meet the temporal requirement. For these reasons, it was decided to use the YOLOv5S model with an input size of 384. This model can process images at 25 FPS on an NVIDIA Jetson Nano with an average AP of 0.62.

Table 18 AP–FPS comparison of YOLOv5 models varying input size

Model	Input size	AP	Inference Time (ms)	FPS
YOLOv5N	704	0.566	65	15.3
	672	0.573	75	13.3
	640	0.567	62	16.2
	608	0.575	57	17.5
	576	0.566	50	19.9
	544	0.583	45	22.1
	512	0.565	38	26.2
	480	0.571	34	29.3
	448	0.578	34	29.6
	416	0.563	29	34.8
	384	0.562	27	37.4
	352	0.558	26	37.8
	320	0.540	25	39.4
YOLOv5S	704	0.682	108	9.2
	672	0.651	101	9.9
	640	0.669	89	11.2
	608	0.669	81	12.3
	576	0.683	80	12.5
	544	0.642	73	13.8
	512	0.637	61	16.4
	480	0.653	56	18.0
	448	0.636	50	19.9
	416	0.625	46	21.8
	384	0.622	40	24.8
	352	0.604	35	28.6
	320	0.599	31	31.8
YOLOv5M	704	0.685	226	4.4
	672	0.684	208	4.8
	640	0.687	187	5.3
	608	0.703	171	5.9
	576	0.692	164	6.1
	544	0.690	149	6.7
	512	0.675	117	8.6
	480	0.663	108	9.3
	448	0.663	101	9.9
	416	0.655	92	10.8
	384	0.625	80	12.5
	352	0.651	72	13.9
	320	0.612	64	15.7

4.5 Summary

As indicated in Sect. 3, a number of materials and methods are required to implement the low-cost system for real-time verification of PPE. After extensive experimentation, the optimal configuration is found to be:

- Materials:
  - Dataset: The public CHV dataset is used to generate a model capable of detecting PPE. However, if the model is evaluated with images of the real facility where the system is applied, the accuracy of detection decreases. For this reason, it is necessary to add some images of the facility to train the model. In addition, if transfer learning is used, the results improve considerably.
  - Camera: The IMX219-160 camera was used in the experiments carried out. Good results were achieved with this camera. In addition, it has a low price, which favors the scalability of the system.
  - Computing device: Deep learning image processing is computationally expensive, so the NVIDIA Jetson Nano was selected. The NVIDIA Jetson

Nano has a low price and incorporates a GPU which allows real-time image processing.

- Methods:
  - Alarm generation method: From the experimentation carried out, it can be concluded that object tracking is necessary to decide whether an alarm should be generated. The best results are obtained using YOLOv5S with an input size of 384 and the end-to-end model. With this configuration, the alarms are generated correctly in real time (30 FPS).
  - Metric to evaluate system performance: Since object tracking is used, the metric to evaluate the quality of the alarms is the MOTA. This metric evaluates the quality of the performed actions, and the tracking of the objects of interest.

Using these materials and methods, a low-cost system can be put in place to ensure the safety of workers at all times.

5 Conclusion

In recent years, there has been a stagnation in the reduction of occupational accidents in industrial environments. This study endeavors to contribute to further reducing the accident rate by creating and implementing a real-time system for PPE verification. With a keen focus on encouraging widespread adoption, careful consideration has been given to the system's cost-effectiveness. Additionally, a decentralized and highly scalable solution is proposed, enabling companies to dynamically adjust the monitored areas based on demand.

Upon evaluating this cost-effective real-time PPE verification system within a real industrial facility, where steel coils are stored, it is concluded that it is an effective solution. The system facilitates continuous monitoring of safety parameters in industrial facilities, immediately notifying management when these parameters exceed predetermined thresholds. Notably, the ease of use of the system reduces the need for large investments in training and technical assistance. In essence, this affordable real-time PPE verification system proves to be a secure, efficient, and economical avenue for upholding worker safety.

However, a significant drawback of this system pertains to its reliance on camera placement and potential occlusions. While the employed object tracker mitigates the impact of occlusions, extended presence within a blind spot may impede the verification of PPE adherence for workers.

As a prospect for future research, the exploration of model optimization methods is recommended. This endeavor seeks to enable the execution of these models on edge computing devices with larger image sizes, thereby enhancing accuracy and system performance.

Acknowledgements This work was supported in part by the Spanish National Plan for Research, Development and Innovation under Project No. PID2021-124383OB-I00.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature.

Data availability The data used to train the model are public. They can be accessed through the following link: https://github.com/ZijianWang1995/ppe_detection [Accessed Online on March 28 2023]. The data of the private facility where the created system is evaluated are not available.

Declaration

Conflicts of interest The authors have no conflicts of interest to declare that are relevant to the content of this article. All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript.

References

1. Eurostat, "Accidents at work statistics," accessed on 29 Jun 2022. [Online]. Available: https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Accidents_at_work_statistics
2. U. B. of Labor Statistics, "Census of fatal occupational injuries summary, 2020," accessed on 06 Jul 2022. [Online]. Available: https://www.bls.gov/news.release/cfoi.nr0.htm
3. Eurostat, "Development of fatal accidents at work for the five nace sections with the highest risk levels, eu, 2010-2019," accessed on 29 Jun 2022. [Online]. Available: https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Accidents_at_work_statistics#Accidents_2010_to_2019
4. U. B. of Labor Statistics, "Nearly 50 years of occupational safety and health data," accessed on 06 Jul 2022. [Online]. Available: https://www.bls.gov/opub/btn/volume-9/nearly-50-years-of-occupational-safety-and-health-data.htm
5. Xu, Y., Wang, M., Feng, Y., Xu, Y., Li, Y.: "Does managers' walking around benefit workplace safety? a safety climate intervention field study," Safety Science, vol. 161, p. 106062, (2023). [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925753523000048
6. He, K., Zhang, X., Ren, S., Sun, J.: "Deep residual learning for image recognition," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June (2016)

7. Gagliardi, A., de Gioia, F., Saponara, S.: "A real-time video smoke detection algorithm based on kalman filter and cnn," Journal of Real-Time Image Processing, vol. 18, no. 6, pp. 2085–2095, Dec 2021. [Online]. Available: https://doi.org/10.1007/s11554-021-01094-y
8. Saponara, S., Elhanashi, A., Zheng, Q.: "Developing a real-time social distancing detection system based on yolov4-tiny and bird-eye view for covid-19," Journal of Real-Time Image Processing, vol. 19, no. 3, pp. 551–563, Jun 2022. [Online]. Available: https://doi.org/10.1007/s11554-022-01203-5
9. Son, H., Kim, C.: "Integrated worker detection and tracking for the safe operation of construction machinery," Automation in Construction, vol. 126, p. 103670, (2021). [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0926580521001217
10. Li, J., Zhao, X., Zhou, G., Zhang, M.: "Standardized use inspection of workers' personal protective equipment based on deep learning," Safety Science, vol. 150, p. 105689, (2022). [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925753522000297
11. Khan, N., Saleem, M.R., Lee, D., Park, M.-W., Park, C.: "Utilizing safety rule correlation for mobile scaffolds monitoring leveraging deep convolution neural networks," Computers in Industry, vol. 129, p. 103448, (2021). [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0166361521000555
12. Chern, W.-C., Hyeon, J., Nguyen, T.V., Asari, V.K., Kim, H.: "Context-aware safety assessment system for far-field monitoring," Automation in Construction, vol. 149, p. 104779, (2023). [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0926580523000390
13. Barro-Torres, S., Fernández-Caramés, T.M., Pérez-Iglesias, H.J., Escudero, C.J.: "Real-time personal protective equipment monitoring system," Computer Communications, vol. 36, no. 1, pp. 42–50, (2012). [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0140366412000060
14. Li, J., Zhang, J., Zhang, X., Wang, S.: "Lightweight helmet detection algorithm based on improved YOLOv5," in Second International Conference on Advanced Algorithms and Signal Image Processing (AASIP 2022), K. Subramaniyam, Ed., vol. 12475, International Society for Optics and Photonics. SPIE, (2022), p. 124751P. [Online]. Available: https://doi.org/10.1117/12.2659641
15. Deng, Z., Yao, C., Yin, Q.: "Safety helmet wearing detection based on jetson nano and improved yolov5," Advances in Civil Engineering, vol. (2023), p. 1959962, May 2023. [Online]. Available: https://doi.org/10.1155/2023/1959962
16. Kamal, R., Chemmanam, A.J., Jose, B.A., Mathews, S., Varghese, E., "Construction safety surveillance using machine learning," International Symposium on Networks. Computers and Communications (ISNCC) **2020**, 1–6 (2020)
17. Wang, Z., Wu, Y., Yang, L., Thirunavukarasu, A., Evison, C., Zhao, Y.: "Fast personal protective equipment detection for real construction sites using deep learning approaches," Sensors, vol. 21, no. 10, (2021). [Online]. Available: https://www.mdpi.com/1424-8220/21/10/3478
18. Meddeb, H., Abdellaoui, Z., Houaidi, F.: "Development of surveillance robot based on face recognition using raspberry-pi and iot," Microprocessors and Microsystems, vol. 96, p. 104728, (2023). [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0141933122002575
19. Sati, V., Sánchez, S.M., Shoeibi, N., Arora, A., Corchado, J.M.: Face detection and recognition, face emotion recognition through nvidia jetson nano. In: Novais, P., Vercelli, G., Larriba-Pey, J.L., Herrera, F., Chamoso, P. (eds.) Ambient Intelligence - Software and Applications, pp. 177–185. Springer International Publishing, Cham (2021)
20. Kumar, V.S., Ashish, S.N., Gowtham, I., Balaji, S.A., Prabhu, E.: "Smart driver assistance system using raspberry pi and sensor networks," Microprocessors and Microsystems, vol. 79, p. 103275, (2020). [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0141933120304348
21. Girshick, R., Donahue, J., Darrell, T., Malik, J.: "Rich feature hierarchies for accurate object detection and semantic segmentation," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June (2014)
22. Liu, W., Anguelov, D., Erhan, D., Szegedy, D., Reed, S., Fu, C.-Y., Berg, A.C.: "Ssd: Single shot multibox detector," in European conference on computer vision. Springer, (2016), pp. 21–37
23. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: "You only look once: Unified, real-time object detection," in Proceedings of the IEEE conference on computer vision and pattern recognition, (2016), pp. 779–788
24. Redmon, J., Farhadi, A.: "Yolo9000: better, faster, stronger," in Proceedings of the IEEE conference on computer vision and pattern recognition, (2017), pp. 7263–7271
25. Redmon, J., Farhadi, A.: "Yolov3: An incremental improvement," arXiv preprint arXiv:1804.02767, (2018)
26. Bochkovskiy, A., Wang, C.-Y., Liao, H.-Y.M.: "Yolov4: Optimal speed and accuracy of object detection," arXiv preprint arXiv:2004.10934, (2020)
27. G. Jocher, A. Stoken, J. Borovec, NanoCode012, ChristopherSTAN, L. Changyu, Laughing, tkianai, yxNONG, A. Hogan, lorenzomammana, AlexWang1900, A. Chaurasia, L. Diaconu, Marc, wanghaoyang0106, ml5ah, Doug, Durgesh, F. Ingham, Frederik, Guilhen, A. Colmagro, H. Ye, Jacobsolawetz, J. Poznanski, J. Fang, J. Kim, K. Doan, and L. Yu, "ultralytics/yolov5: v4.0 - nn.SiLU() activations, Weights & Biases logging, PyTorch Hub integration," January 2021. [Online]. Available: https://doi.org/10.5281/zenodo.4418161
28. Li, C., Li, L., Jiang, H., Weng, K., Geng, Y., Li, L., Ke, Z., Li, Q., Cheng, M., Nie, W. et al.: "Yolov6: a single-stage object detection framework for industrial applications," arXiv preprint arXiv:2209.02976, (2022)
29. Wang, C.-Y., Bochkovskiy, A., Liao, H.-Y.M.: "Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," arXiv preprint arXiv:2207.02696, (2022)
30. Bewley, A., Ge, Z., Ott, L., Ramos, F., Upcroft, B.: "Simple online and realtime tracking," . IEEE International Conference on Image Processing (ICIP) **2016**, 3464–3468 (2016)
31. Wojke, N., Bewley, A., Paulus, D.: "Simple online and realtime tracking with a deep association metric,". IEEE International Conference on Image Processing (ICIP) **2017**, 3645–3649 (2017)
32. Du, Y., Song, Y., Yang, B., Zhao, Y.: "Strongsort: Make deepsort great again," arXiv preprint arXiv:2202.13514, (2022)
33. Padilla, R., Passos, W.L., Dias, T.L.B., Netto, S.L., da Silva, E.A.B.: "A comparative analysis of object detection metrics with a companion open-source toolkit," Electronics, vol. 10, no. 3, (2021). [Online]. Available: https://www.mdpi.com/2079-9292/10/3/279
34. Dendorfer, P., Rezatofighi, H., Milan, A., Shi, J., Cremers, D., Reid, I., Roth, I., Schindler, K., Leal-Taixé, L.: "Mot20: A benchmark for multi object tracking in crowded scenes," arXiv preprint arXiv:2003.09003, (2020)
35. Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) Computer Vision - ECCV 2014, pp. 740–755. Springer International Publishing, Cham (2014)