



# A new YOLO-based method for real-time crowd detection from video and performance analysis of YOLO models

Mehmet Şirin Gündüz<sup>1</sup> · Gültekin Işık<sup>1</sup>

Received: 13 July 2022 / Accepted: 8 November 2022 / Published online: 30 January 2023  
© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2023

## Abstract

As seen in the COVID-19 pandemic, one of the most important measures is physical distance in viruses transmitted from person to person. According to the World Health Organization (WHO), it is mandatory to have a limited number of people in indoor spaces. Depending on the size of the indoors, the number of persons that can fit in that area varies. Then, the size of the indoor area should be measured and the maximum number of people should be calculated accordingly. Computers can be used to ensure the correct application of the capacity rule in indoors monitored by cameras. In this study, a method is proposed to measure the size of a prespecified region in the video and count the people there in real time. According to this method: (1) predetermining the borders of a region on the video, (2) identification and counting of people in this specified region, (3) it is aimed to estimate the size of the specified area and to find the maximum number of people it can take. For this purpose, the You Only Look Once (YOLO) object detection model was used. In addition, Microsoft COCO dataset pre-trained weights were used to identify and label persons. YOLO models were tested separately in the proposed method and their performances were analyzed. Mean average precision (mAP), frame per second (fps), and accuracy rate metrics were found for the detection of persons in the specified region. While the YOLO v3 model achieved the highest value in accuracy rate and mAP (both 0.50 and 0.75) metrics, the YOLO v5s model achieved the highest fps rate among non-Tiny models.

**Keywords** Real-time video processing · Person detection · People counting · Area estimation · Deep learning · YOLO

## 1 Introduction

According to the report published by the World Health Organization (WHO) [1], it has been revealed that the coronavirus disease (COVID-19) has infected millions of people globally and caused deaths on a large scale. For a long time, countries stopped doing the daily activities on which their livelihood depended. Countries have started to carry out their daily life activities by applying the health protocols prescribed by WHO. One of the health measures is to prevent people from crowding in indoor spaces. To achieve this, the maximum number of people that the indoor area can accommodate is determined according to its size. When the number of people per square meter is greater than a certain threshold, an action can be taken accordingly.

It is difficult to manually control whether the maximum number of people has been reached in many places, such as banks, train stations, shopping malls, and schools. Therefore, computers can be used to provide this control in indoor spaces monitored by cameras. For this reason, in this study, a region within the indoor area in the video recording was selected and the size of that region was estimated. A threshold value was determined as one person per square meter. Then, the people in this area were counted. If there are more people than the capacity of the area, it is marked in red; otherwise, it is marked in green.

Deep learning is one of the popular topics in the field of machine learning. Deep learning algorithms have been successfully applied in different modalities and fields. Image processing [2], machine translation [3], and natural language processing [4] are some of them. Deep learning is also used in other fields apart from these. For example, in recent years, deep learning algorithms are used for sound classification [5], language recognition [6], or cancer diagnosis [7] as well.

There are two types of methods for person detection: Region of Interest (ROI) and Line of Interest (LOI). In

✉ Gültekin Işık  
gultekin.isik@igdir.edu.tr

<sup>1</sup> Department of Computer Engineering, Iğdir University, Iğdir, Turkey

the ROI-based method, the number of people in a limited region is estimated. In the LOI-based method, on the other hand, the number of people crossing a predetermined line is estimated [8]. Deep learning-based models are used in the ROI-based system approach. Deep learning is frequently used in object detection and classification tasks. This situation encouraged researchers to use deep learning on people counting problems, as well. In this study, deep learning methods were used with an ROI-based approach.

Object detection is made to locate objects within the image and classify each object. For this, the detected object is enclosed in a bounding box and the class of this object is tried to be estimated. The YOLO models use deep learning methods for object detection and the first version was developed by Redmon et al. [9].

Detecting and counting people in indoor spaces have been one of the important tasks of video surveillance systems [10]. Since this task has become important, an increase has been observed in studies conducted in recent years. YOLO models are frequently used to detect people, count them, and measure the distance between them. For example, Punn et al. [11] used the Deepsort [12] algorithm together with the YOLO v3 model for pedestrian detection and distance measurement. Compared to other object detection algorithms, they demonstrated the success of YOLO v3 [13] and Deepsort algorithm. Rahim et al. [14] used YOLO v4 [15] for person detection in low-light situations and monitored social distance with motionless time of flight (ToF) cameras. Person detection from the top view was investigated with YOLO [16]. According to this study, with the YOLO v3 model, people were detected and counted from the top view with an accuracy of up to 95%.

The YOLO v5 model [17] was used for crowd detection [18]. It has been observed that the YOLO v5 model gives good results in counting applications where there is no dense crowd. Ren et al. [19] proposed the YOLO-PC algorithm, which is an advanced version of YOLO. They used YOLO, which has a high accuracy rate, to develop a people counting system. High accuracy rates were obtained with this proposed algorithm. Menon et al. [20] used INRIA and ShanghaiTech datasets to count pedestrians using the YOLO v3 model. While the tests performed on the INRIA dataset resulted in an accuracy of 96.1%, relatively less accuracy was obtained with 87.3% in the ShanghaiTech dataset.

In addition to these studies, crowd detection was performed using image and sound features together, since feature extraction is difficult in low-quality images [21]. In this study, crowd counting is considered as a regression problem. Convolutional Neural Network (CNN) is used to extract image features and VGG network architecture is used for audio features. Also, in a different study [22], quantum feature extraction was performed in addition to the classical CNN architecture for crowd feature extraction.

In this study, YOLO models were used (1) to detect and count the people, and (2) to estimate the area of the specified region of the indoor space. An algorithm is proposed to estimate the area of a specified region where people are present and to determine the maximum number of people to settle in this area. For this proposed algorithm, YOLO v3, YOLO v4, YOLO v5s, YOLO v3-Tiny, and YOLO v4-Tiny models were used and performance analyses of these models were made. Accordingly, the YOLO v3, YOLO v5s, and YOLO v4 models detected persons within a specified region with 96.89%, 96.12%, and 94.57% accuracy rates, respectively.

## 2 Materials and methods

In this study, it is focused on calculating the area of a region with predetermined borders in indoor spaces and determining the maximum number of people who can settle in this region. For this, cameras that observe indoor spaces are used. YOLO models were used to detect and count people inside the specified area. In this section, the YOLO models and dataset used are introduced.

### 2.1 YOLO models

YOLO (You Only Look Once) are models that perform object detection processes in real time, with high accuracy and quickly. YOLO is a single-stage Convolutional Neural Network (CNN)-based model. In the single-stage model, objects can be detected without the need for a preliminary stage. Examples of single-stage detectors are SSD [23] and YOLO models. YOLO applies a single Convolutional Neural Network model to the image and splits the image into grids. Each grid makes an estimation of the bounding boxes and the associated confidence score. According to the estimated confidence score, the class of the object in the bounding box is determined.

YOLO v3 uses a new network to perform feature extraction. This network is connected to the network used in YOLO v2 Darknet-19 [13]. YOLO v3, whose classifier network is more successful than other versions of YOLO, is based on a pre-trained 53-layer Darknet-53 network with Imagenet [24] dataset [13]. This architecture consists of 53 convolutional layers. It is slower than YOLO v2 as it has more layers. YOLO v3 contains five residual blocks. Each residual block consists of multiple residual units. The residual blocks found in YOLO v3 are not present in YOLO v2. There is a mixed approach between residual network (ResNet) elements.

This approach divides the network into consecutive  $1 \times 1$  and then  $3 \times 3$  convolutional layers to perform feature extraction. 53 more layers are added on top of this for the detection

task, resulting in a 106-layer fully convolutional architecture for YOLO v3.

Bochkovskiy et al. [15] proposed the YOLO v4 algorithm, an advanced version of YOLO v3. YOLO v4 has improved over YOLO v3 in terms of both speed and accuracy. It has been observed that the CSPDarknet53 feature extraction model gives better results in YOLO v4 [15]. YOLO v4 architecture is given in Fig. 1. YOLO v4 architecture has three different blocks: backbone, neck, and dense prediction blocks. Figure 2 shows the main structure of YOLO v4 architecture.

The pipeline consists of three parts: backbone, neck, and head. CSPDarkNet53 [25], which is used as the backbone, is used for feature extraction and is one of the factors that increase the accuracy of the system [26]. The Neck is the layer between the Backbone and the Head. An intermediate

layer called neck has been added to obtain more information while estimating objects. Head predicts classes and location of objects (e.g., person). It also calculates the size (width and height) and the coordinates of the bounding boxes [27].

There is a CSPDarknet53 network in the backbone. The network takes an image or its frame as input. The backbone is responsible for extracting the features of the image or frame. Backbone divides the current layer into two parts, DenseNet and CSPDenseNet (Fig. 3). One of the outputs from the first layer goes to the Dense block, and the other goes directly to the next transition layer, as shown in Fig. 3b.

The Dense block includes Batch Normalization, Rectified Linear Unit (ReLU), and a convolution layer. Each layer of the Dense block takes as input the feature maps of all previous layers and helps to find complex features of an image. YOLO v4 has CSPDarknet53, while YOLOv3 has

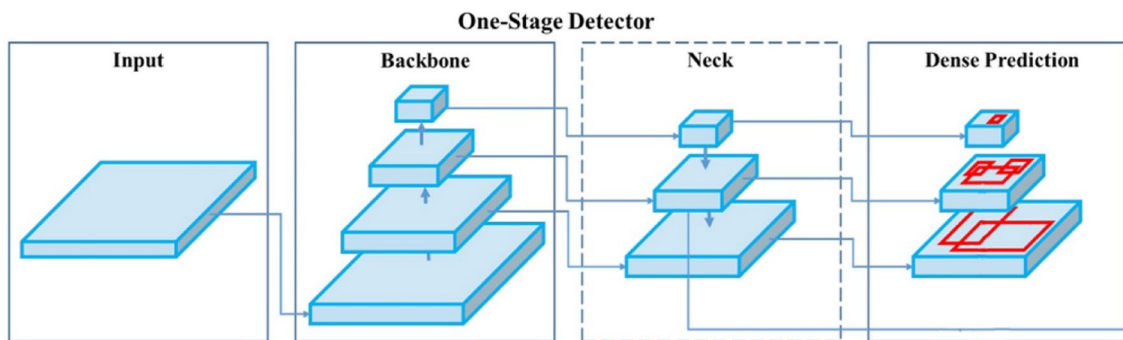


Fig. 1 YOLO v4 architecture [15]

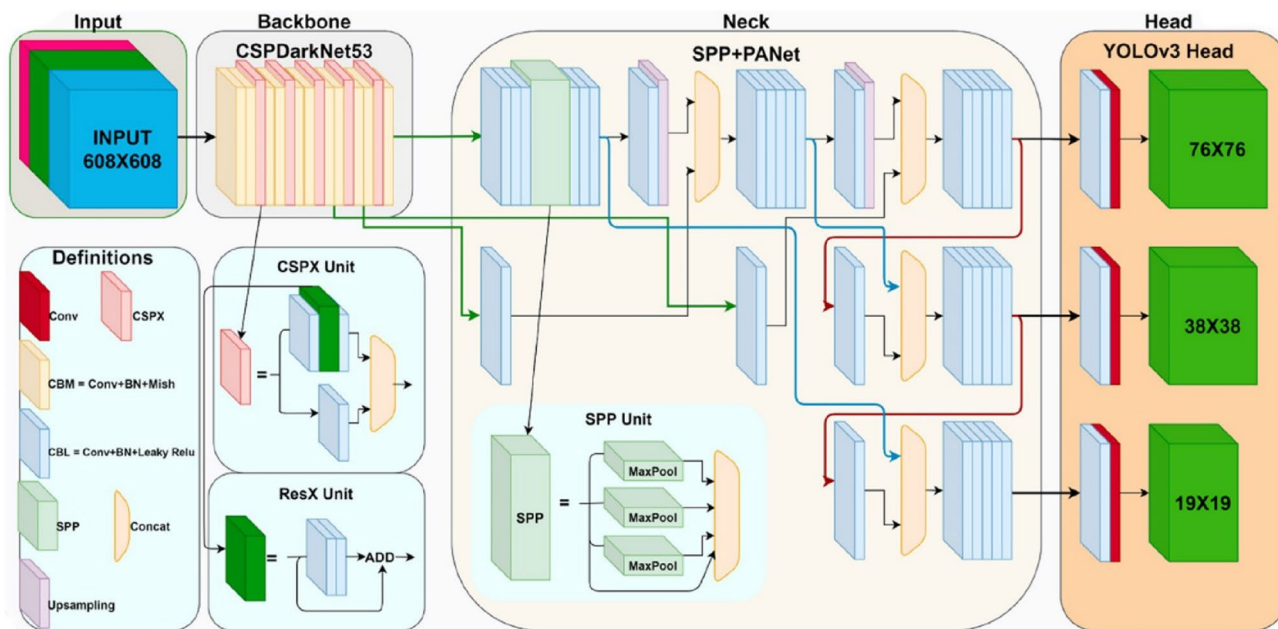


Fig. 2 Main structure of YOLO v4 architecture [28]

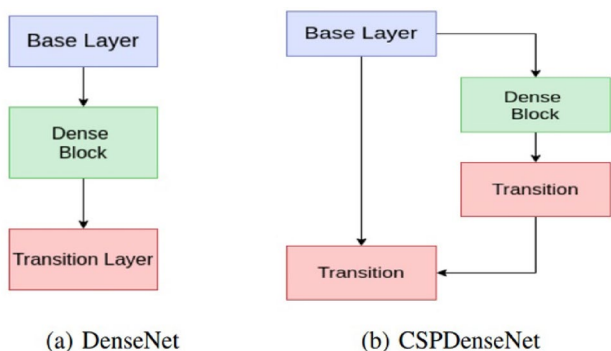


Fig. 3 Structure comparison between a Standard DenseNet and b CSPDenseNet [29]

Darknet53 as its backbone. YOLO v3 uses Feature Pyramid Network (FPN), while YOLO v4 uses Spatial Attention Module (SAM) and Path Aggregation Network (PAN) instead of FPN. However, in YOLO v4, there is no max pooling and average pooling [15].

YOLO v5 was developed by Jocher et al. [17]. The latest and fastest version of YOLO models is YOLO v5 [17]. YOLO v5 differs from previous versions as it is a PyTorch [30] implementation. As in YOLO v4, Cross-stage Partial Networks (CSP) is used in the backbone and Path Aggregation Network (PANet) is used in the neck [31]. In the head part, the model used in YOLO v4 is used. In YOLO v5, Leaky ReLU (LReLU) is used in hidden layers and sigmoid activation function is used for object detection in the last layer. In YOLO v5, the default optimization algorithm for training is Stochastic Gradient Descent (SGD) [32].

The network structure of YOLO v5 is divided into three parts: backbone, neck, and output. The backbone part extracts features from the input images. The neck part combines the extracted features and creates a feature map, and detects objects from the feature maps in the output part [33]. There are two types of CSP in YOLO v5. One of them is used in the backbone and the other in the neck network. While the CSP network in the backbone consists of one or more residual units, the CSP network in the neck replaces the residual units with CBL modules (Conv2D, Batch Normalization, and Leaky ReLU). CSP connects the front and back layers of the network. Thus, it increases the inference

speed by reducing the size of the model [34]. The Spatial Pyramid Pooling (SPP) layer aggregates object-related features and produces fixed-length vectors to other layers. In other words, it performs the process of collecting some information about the object at a deeper stage of the network hierarchy (between convolutional and fully connected layers) without clipping the object [35]. By adjusting the width and depth of the YOLO model, four models with different parameters are obtained, namely YOLO v5s, YOLO v5m, YOLO v5l, and YOLO v5x.

YOLO v3 and YOLO v4 models have been minimized and optimized, and a new version (Tiny) has been developed. The network size of tiny models and the number of convolutional layers in the CSP backbone have been significantly reduced. YOLO v5 is named differently in itself (s, m, l, x). YOLO v3, YOLO v3-Tiny, YOLO v4, YOLO v4-Tiny, and YOLO v5s models were used in this study. Summary information of the models used is given in Table 1 [17]. YOLO v5s has 224 layers and 7.2 million trainable parameters. *B* in Table 1 indicates how many anchor boxes will be used for each detection, and *C* indicates the number of classes.

### 2.2 Dataset

MS COCO dataset was used in this study. This dataset is a large-scale object detection, segmentation, and image caption dataset [36]. MS COCO has a total of 1.5 million object instances belonging to 80 object categories, including the person category. Since this study is only for person detection, images in the person category in the MS COCO dataset were used.

### 3 Our proposed approach: area calculation algorithm for a specified region in a video

The aim of this study is to estimate the area of a specified region (Fig. 4) within the space seen in a video. The area of these regions can be calculated in pixels with the formula in Eq. 1. However, due to reasons such as image resolution, camera angle, and video quality, the size in pixels does not represent the actual size of the area. Actual size should be in square meters. For this, pixel-square meter conversion

Table 1 Characteristics of YOLO models (v3, v4, v5s, and Tiny) [17]

	YOLO v3	YOLO v3-Tiny	YOLO v4	YOLO v4-Tiny	YOLO v5s
# of Layers	107	24	162	38	224
Weight (MB)	236	33.7	245	23.1	28.8
AP@0.50	58.2	35.4	48.7	40.2	56.8
Backbone	DarkNet53	DarkNet53	CSPDarkNet53	CSPDarkNet53	CSPDarkNet53
Neck	FPN	FPN	SPP and PANet	SPP and PANet	PANet
Head	$B \times (5 + C)$	$B \times (5 + C)$	$B \times (5 + C)$	$B \times (5 + C)$	$B \times (5 + C)$



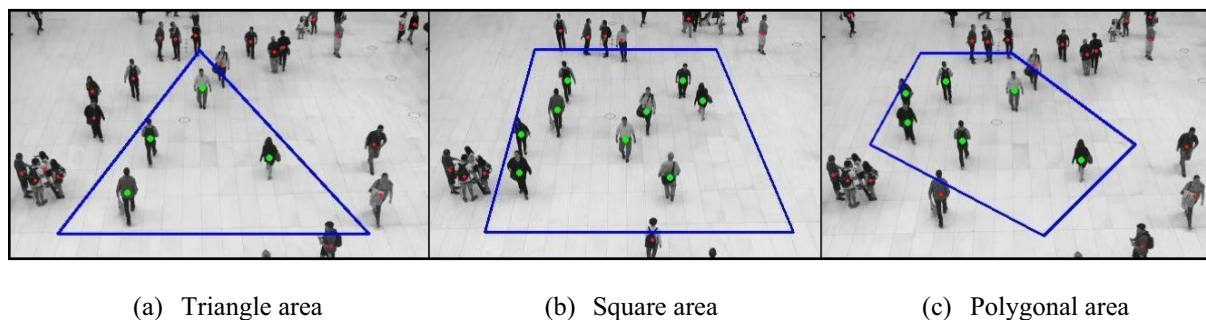


Fig. 4 Determination of a region as desired

should be done. Estimating the actual size ( $m^2$ ) from an image is a challenging problem. To overcome this, an object of known size is taken as a reference in the image [37]. Here, the reference object used in this study is a person.

If the area occupied by an average person (Eq. 2) is known, the area of the region where these people are can be estimated.

The method proposed in this study is based on the bounding boxes of persons detected by the YOLO models. YOLO models generate a rectangular shaped bounding box for each of the persons it detects in the image. The height and width values of the bounding boxes are provided by the model in pixels. The areas of these rectangles are calculated according to Eq. 2 and averaged. If this calculated average area value ( $px^2$ ) is proportioned to the area occupied by an average person ( $m^2$ ), a conversion coefficient from pixels to square meters is found. Using this coefficient, the area of any region in the image can be estimated.

Data such as height and shoulder width vary by geography. For example, according to a study [38] in which anthropometric measurements were made in Turkey, the average height was found to be 1708 mm in men and 1598 mm in women (avg.  $\sim 1.65$  m). In the same study, the average shoulder width was found to be 475 mm in men and 366 mm in women (avg.  $\sim 0.4$  m). In this study, the area covered by an average person is taken as 0.66 square meters (Eq. 2).

It is checked whether the persons detected in the video are within the predetermined borders. Using the heights ( $h$ ) and widths ( $w$ ) of the persons detected in the region, the area of each person (Eq. 2) is expanded to approximately  $1 m^2$  (Eq. 3). The square meter of the specified region is estimated (Eq. 5) and the number of people required to be in this region is determined (Eq. 5). The area calculation algorithm of the specified region is shown in Fig. 5

$$A = \text{abs} \left\{ (x_1y_2 - x_2y_1) + (x_2y_3 - x_3y_2) + \dots (x_ny_1 - x_1y_n) \times \frac{1}{2} \right\}. \tag{1}$$

The area of the specified region is found as  $R_x px^2$  with the help of Eq. 1 in pixels. The  $x$  and  $y$  values given here indicate the vertex coordinates of the specified region (polygon). If a person's area is represented by  $P_1$ , height  $h$ , and shoulder width  $w$ , then the rough area of a person can be found by Eq. 2. As a result of Eq. 2, the area of a person is found as  $0.66 m^2$

$$P_1 = h \times w. \tag{2}$$

Since 1 person per square meter is taken as the threshold value, if the area occupied by a person is expanded to  $1 m^2$  (Eq. 3), the area of the specified region can be found in  $m^2$ . For this, it is necessary to add  $3/5$  of a person's area to their area (Eq. 3)

$$P_2 = P_1 + P_1 \times \frac{3}{5} \\ P_2 \cong 1m^2. \tag{3}$$

As a result of Eq. 3, the area of a person ( $P_2$ ) becomes approximately  $1 m^2$ . The width and height of the persons (bounding boxes) detected in the specified region are known in pixels. Using Eq. 2, the average area ( $B_{x_1}$ ) of the bounding boxes is calculated. Equation 4, which is the same as the transformation in Eq. 3, should be applied to this value ( $B_{x_2}$ ).

To calculate the area of the specified region in  $m^2$ , the area of the region in  $pixels^2$  must be divided by the value  $B_{x_2}$ . The  $R_m$  value obtained in this way gives the area of the specified region in square meters. The  $R_m$  value also indicates the human capacity of the specified region

$$B_{x_2} = B_{x_1} + B_{x_1} \times \frac{3}{5} \tag{4}$$

$$R_m = R_x / B_{x_2}. \tag{5}$$

### 3.1 Advantages of the proposed algorithm

The area of the specified region is calculated according to the height and width of the persons detected. Since the

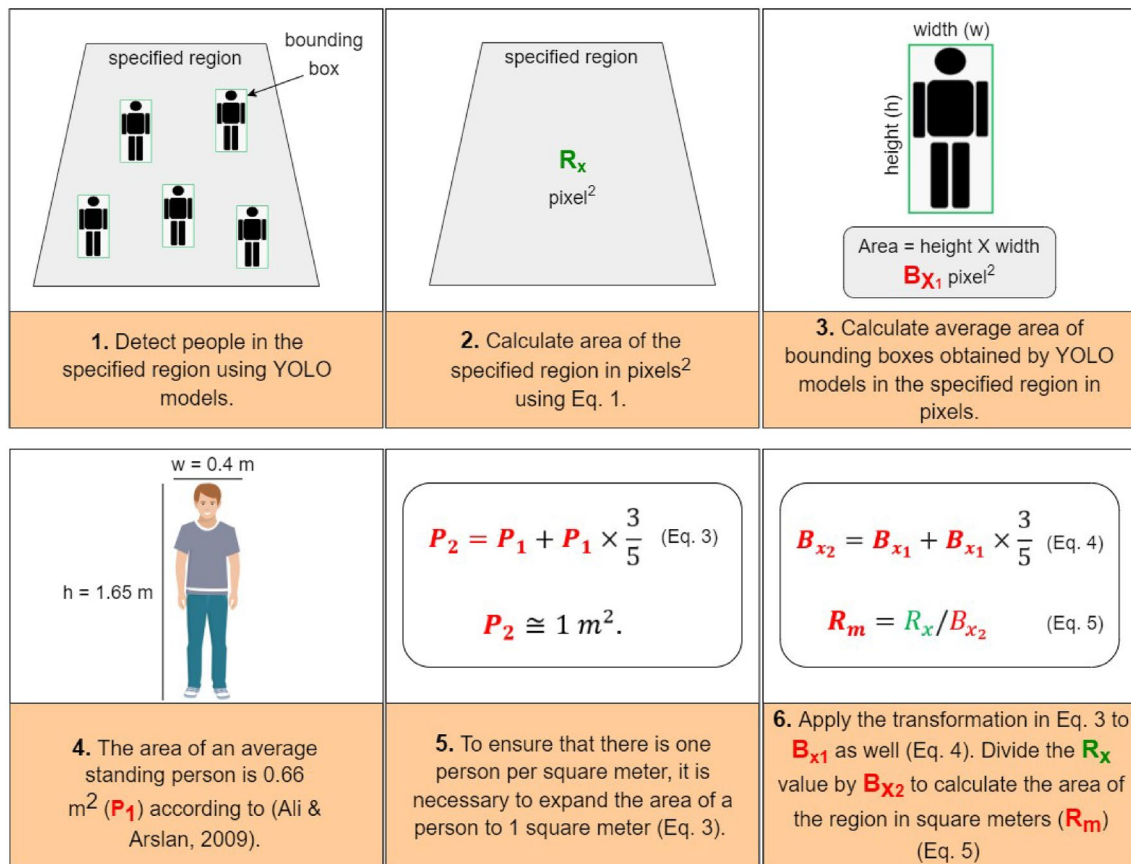


Fig. 5 Area calculation algorithm for a specified region in a video

height and width of the people are adapted according to the camera resolution, changing the camera resolution will not adversely affect the operation of the algorithm. If the area in pixels<sup>2</sup> of the specified region remains the same, realistic results are obtained even if the video resolution changes. Because in videos of different resolutions, the height and width of persons detected vary according to the resolution.

### 3.2 Disadvantages

Height and width of persons are determined by bounding boxes. The height and width of the bounding box may turn out to be different than it should be, for example when the video resolution is bad or persons are not clearly visible. In such a case, the area calculation may be different.

## 4 Experimental study

In the study, Python programming language and OPENCV Library were used together. Deepsort algorithm [12] was used as the person tracking algorithm. The Deepsort algorithm uses a CNN model for object classification. Thanks to

the CNN model, the most distinctive feature of the object to be classified is determined and the classification process is performed. Each object detected by the deepsort algorithm is passed through the neural network and a vector is obtained. Two objects are associated using these vectors. This vector is called the *appearance feature vector*. Therefore, it takes into account previous and current frame information to estimate the current frame without needing to process the entire video at once. The convolutional neural network continues to be trained until satisfactory success is achieved.

In order for object tracking to be carried out, first object detection must be performed. The object detection process is the process of detecting objects using a bounding box. The location, type, and class of the detected object are determined. These determined features are assigned to YOLO as a class label. The class label used in this study is *person*. After YOLO learns the person’s characteristics, it divides the image into cells called grids and determines the bounding boxes in case there is a person in the cell. In case of more than one bounding box per person, it uses non-maximum suppression (NMS) to reduce the number of bounding boxes to one per person. Finally, the number of people is found using the number of bounding boxes in the image or video.

In this study, to count people, first of all, a region with certain borders should be selected on the video. These specified regions can be shaped as triangles, squares, or polygons (Fig. 4). In a frame, persons inside the specified region are counted as follows: all objects except “person” in the MS COCO dataset are discarded. In this way, persons were detected on the images using pre-trained YOLO models. However, instead of all the persons in the image, only those within the specified region were taken into account. A simple counter was used here. The counter was incremented by 1 if the center point of the bounding boxes produced by YOLO were within the specified region, and they were not included in the counting if they were not within the region. In this way, the counter was increased by 1 for each person detected in the region in each frame. Then, the area of the specified region is calculated in pixel type. Calculation is made in such a way that the maximum number of persons will fit in the area that we have determined as 1 person per square meter. Examples from the real-time experimental environment are shown in Fig. 7. The image resolutions here are  $800 \times 600$  and  $1024 \times 768$  pixels.

In this study, other parameters except YOLO models were kept the same for consistency of results and fair comparison. The characteristics of the video used also in [40], such as the number of frames, video duration, and resolution, are given in Table 2.

For example, all experiments were done using frames of the same resolution. Using the video given in Table 2, the results obtained from the algorithms of detecting persons, counting people, calculating the area of the specified region, and determining the maximum number of persons to settle in this area are given in Table 3. The steps followed to find the maximum number of people the specified region can take are given in Fig. 6. YOLO-based algorithms were studied with input image dimensions of  $416 \times 416$  pixels. During the application, 0.4 Intersection over Union (IoU, Eq. 8) threshold and 0.4 confidence threshold values were kept constant in all YOLO models.

The YOLO v5 model is not supported by OpenCV as it is written with the PyTorch application. YOLO v5s model was converted and run in ONNX format supported by OpenCV. Tests were performed using  $1024 \times 768$  pixels and  $800 \times 600$  pixels resolutions, respectively, on the images in Fig. 7a and b. Then, the persons in this region were counted. If more people are detected than the specified region capacity, they are marked in red, otherwise in green (Fig. 8).

**Table 2** Characteristics of the video used [39] in the testing phase

Data	# of frames	Duration (s)	Resolution
Video	341	13.64	$800 \times 600$ – $1024 \times 768$

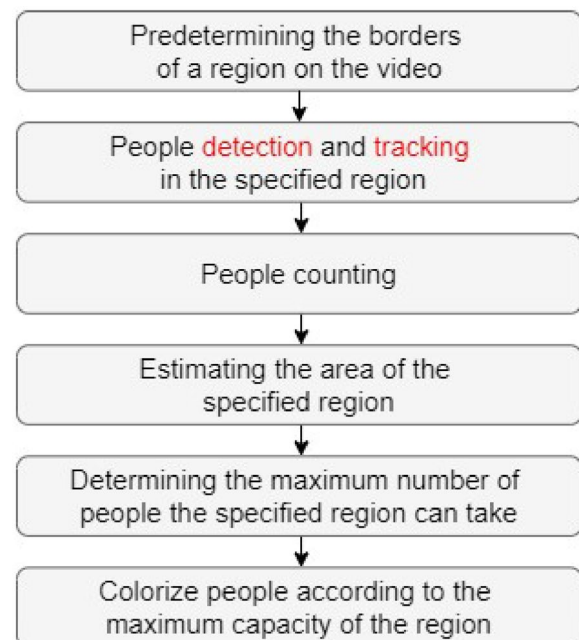
**Table 3** Characteristics of the specified regions given in Fig. 7 and calculation results

	Figure 7a	Figure 7b
Resolutions of images	$1024 \times 768$	$800 \times 600$
Areas of the region in pixels <sup>2</sup>	72,600	72,500
Estimated areas of the region in square meters	18.8	28.1
Number of persons detected in the region	8	11
Maximum capacity of the region	19	28

#### 4.1 Computing environment

Deep learning-based applications often require high-level graphics processing units (GPUs). This situation causes high hardware costs. Deep learning models such as YOLO and MASK R-CNN can run at 1–3 FPS on the CPU. To overcome this problem, NVIDIA GPUs were used together with the OPENCV library. In this study, a computer with Intel Xeon CPU @ 2.20.

GHz 6 MB Cache Memory, NVIDIA Tesla k80 GPU, and 16 GB Memory features were used. CUDA and cuDNN installations have been made, so that the models can run on the GPU.



**Fig. 6** The proposed method to estimate the number of people in a specified region

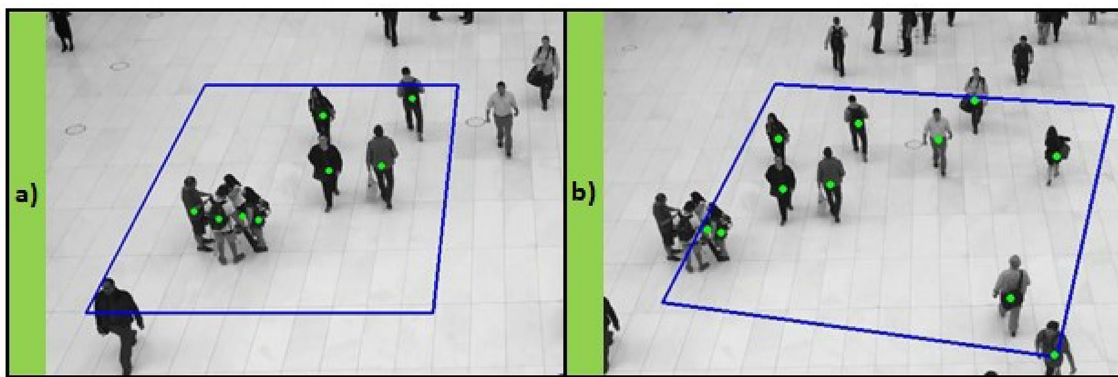


Fig. 7 Samples taken from the experimental environment: **a** 1024×768 pixels and **b** 800×600 pixels

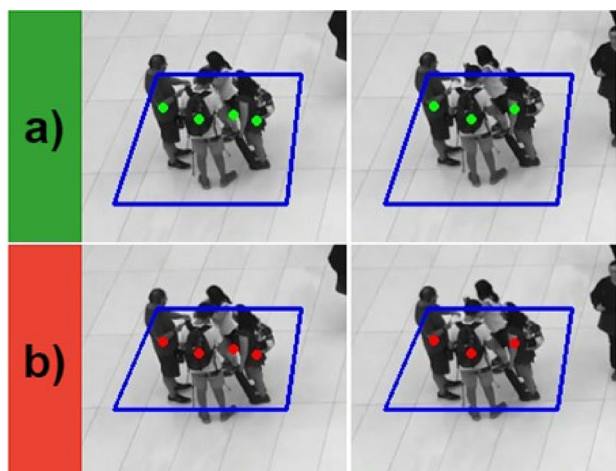


Fig. 8 **a** Number of persons detected in the region < The max capacity. **b** Number of persons detected in the region > The max capacity. Note that the area at the top (**a**) is larger than the area at the bottom (**b**)

## 4.2 Performance metrics

Different metrics are used in this study to measure the performance of the proposed method to count the people in a specified region within the video and to estimate their accuracy.

### 4.2.1 Accuracy rate (Acc)

It shows how close the estimation values of the developed method are to the real values, and it is good if it is high. Accuracy rate formula is given in Eq. 6 (TP: True Positive, TN: True Negative, FP: False Positive, FN: False Negative)

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

### 4.2.2 Mean average precision

A common evaluation metric called Mean Average Precision (mAP) provides a single number as the mean of the Average Precision (AP) values for all classes. This makes it possible to assess a model’s performance using just a single number. As a result, mAP is the evaluation metric that object detection algorithms utilize the most frequently. It is calculated as

$$mAP = \frac{\sum_{q=1}^Q AveP(q)}{Q} \tag{7}$$

where  $Q$  represents the total number of queries in the set and  $q$  represents the average precision query.

Since there is only a “person” class in our study, the number of classes will be 1. mAP (0.5) means that the confidence threshold (IoU) is greater than 0.5 when calculating AP. If  $IoU \geq 0.5$ , the prediction is considered correct

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union} \tag{8}$$

### 4.2.3 FPS

Inference speed (millisecond, ms) or frames per second (FPS) are the most popular metrics for real-time applications. Utilizing a trained deep learning algorithm to generate predictions is known as inference. FPS is the term used to describe how many images can be processed in a second. Real-time applications need low inference speeds or high FPS values.

## 5 Results and discussion

The areas of the specified regions in Fig. 7a and b are 72,600 and 72,500 pixels<sup>2</sup>, respectively. Regions of the same size were determined in the same video frame and different



resolutions were used. Thus, it has been determined to what extent the algorithm gives consistent results. The values calculated by YOLO v5 for the areas of the specified regions found in Fig. 7a and b are shown in Table 3.

Based on the data in Table 3, the area of the region whose boundaries are determined in Fig. 7a has been estimated as 18.8 m<sup>2</sup>. Since the calculation is made in such a way that 1 person per square meter a maximum of 19 persons can fit inside this area. In the video frame from which the image was taken, the total number of persons in this region was found to be 8. In Fig. 7b, the area of the region whose borders are maximum of 28 persons can fit in this area.

The width and height of persons will appear higher in a 1024 × 768 pixels image than in an 800 × 600 pixels image. Therefore, if the same size region is selected in both figures, the capacity of the region determined in the high-resolution image will be lower.

In Table 4, activation and loss functions used by YOLO models, estimated accuracy, video processing time, and FPS values are given. The YOLO v3 read videos at 9.91 FPS, while the YOLO v5s read at 20.06 FPS. Although the FPS value of YOLO v3 was the lowest, the estimated accuracy rate reached the highest value with 88.39%. The YOLO v3 algorithm processed the 14.3 s video in 36.09 s on the NVIDIA CUDA-capable GPU. It is clear that the YOLO v3 algorithm has a speed issue. The YOLO v5s model can be used to eliminate this speed issue. YOLO v5s read 14.3 s video in 17.84 s, reaching 20.06 FPS.

The fact that the video processing time and the FPS values of the YOLO v5s algorithm are close to each other shows the compatibility of this algorithm in terms of time and speed. YOLO v5s correctly predicted persons in the specified area of the video with an accuracy rate of 78.14%.

Despite the reduction in the pre-trained weight size of the YOLO v5s, this result is satisfactory. YOLO v5s is more successful than other models except YOLO v3 in terms of accuracy.

It was analyzed whether persons in the specified regions were correctly detected in each frame of the videos, and the results are given in Table 5. The *Total Number of Persons in the Specified Region* is the total number of persons in the region whose borders are determined in all frames of the video. The *Total Number of Persons Detected in the Specified Region* represents the number of persons detected by the YOLO algorithm used. The inability of the YOLO v4 and YOLO v5s algorithms to detect some persons is due to the IoU and confidence threshold being taken as 0.4. Detection takes place when IoU and confidence threshold values are lowered, but it is not recommended that these values be lower than 0.3.

In Tiny models, especially in the YOLO v3-Tiny model, if the IoU and confidence threshold are set to 0.3 and below, object recognition still does not occur in some frames. This is due to the significant reduction of the number of layers of the YOLO v3-Tiny model. YOLO v3-Tiny is the model with the smallest number of layers (Table 1). In the YOLO v3 algorithm, persons who should be in the specified area were detected with an accuracy rate of 96.89%, while this rate was 96.12% in the YOLO v5s algorithm. Again, according to mAP (0.5) values, YOLO v3 gave the highest result with 98.8%. The fact that YOLO v3 produces higher values than YOLO v5s can be explained by the fact that the YOLO v3 model reaches more frames during video processing. Besides, the YOLO v3-Tiny model gave the lowest results in terms of accuracy and mAP (0.5) values, with 58.13 and 75.9 values, respectively.

**Table 4** Performance comparisons of YOLO models for detecting persons in specified regions

Model	Loss	Activation function	Prediction accuracy %	Video processing time (Sec)	FPS
YOLO v3	CIoU	ReLu	88.39	36.09	9.91
YOLO v3-Tiny	CIoU	ReLu	70.50	7.58	47.22
YOLO v4	CIoU	Leaky ReLu	73.91	32.59	10.98
YOLO v4-Tiny	CIoU	Leaky ReLu	73.14	8.22	43.55
YOLO v5s	CIoU	Leaky ReLu	78.14	17.84	20.06

**Table 5** Metrics for detecting persons in specified region (mAP 0.5)

Model	Total # of persons in the specified region	Total # of persons detected in the specified region	Accuracy (%)	mAP 0.5 (%)
YOLO v3	129	125	96.89	98.8
YOLO v3-Tiny	129	75	58.13	75.9
YOLO v4	129	122	94.57	96.1
YOLO v4-Tiny	129	107	82.94	86.1
YOLO v5s	129	124	96.12	98.5

**Table 6** Metrics for detecting persons in specified region (mAP 0.75)

Model	Total # of persons in the specified region	Total # of persons detected in the specified region	Accuracy (%)	mAP 0.75 (%)
YOLO v3	129	99	76.74	79.2
YOLO v3-Tiny	129	35	27.13	36.3
YOLO v4	129	88	68.21	71.3
YOLO v4-Tiny	129	61	46.95	59.1
YOLO v5s	129	96	74.44	78.6

Since people will be more prominent in a video with a high resolution or people close to the camera, taking the IoU and confidence threshold values as 0.75 does not pose any problem. However, when the IoU and the confidence threshold are 0.75 in low-resolution images, many people in the video images are not recognized. Table 6 shows the results of people in the specified region according to the mAP (0.75) metric. According to the mAP (0.75), YOLO v3 and v5s outperformed other models. The scores were quite low in the tiny models. According to the mAP (0.50) and mAP (0.75), the difference in the number of people recognized by the models is 26 in YOLO v3, 34 in YOLO v4, and 28 in YOLO v5s. In Tiny models, and the difference is 40 for v3-Tiny and 46 for v4-Tiny between mAP (0.50) and mAP (0.75).

## 6 Conclusion

In the COVID-19 pandemic, some rules have emerged that have changed our lives. One of these rules is that there must be a certain number of persons in indoor spaces. According to this rule, the capacity of indoor areas is limited and this capacity should not be exceeded. Then, the size of the indoor area should be calculated and the persons in this area should be counted. In this study, using the Python programming language, OpenCV library, and YOLO models, the area of a region determined by us in the video was estimated in square meters. After determining the area of the specified region, the capacity of the region was calculated approximately. The proposed algorithm to calculate the area of the specified region is based on the bounding boxes that the YOLO models produce when detecting persons. Since there were only persons in the study, pixel-squaremeter conversion was made based on the real-life sizes of persons. The properties of bounding boxes produced by YOLO models when detecting persons vary from model to model. These properties are the sizes of bounding boxes and the prediction values they produce for detecting persons. Therefore, in this study, five models of YOLO were used and their performances were compared.

According to the results of the study, the YOLO v3, YOLO v5s, and YOLO v4 models detected the persons

within the specified region with an accuracy of 96.89%, 96.12%, and 94.57%, respectively, in Table 5. After testing a certain number of videos, the YOLO v3 and YOLO v5s models gave much better results than the others. Considering the results, there is a direct correlation between the number of persons in the specified area and object detection. In test videos, YOLO v5s run faster than YOLO (v3, v4) models in terms of frames per second (FPS). In real-time video processing, analysis processes may be slower as the camera properties, computer performances, and Internet speed come into play. Analyses may differ from image to image. In images with high resolution, the results are closer to reality. Preventing scanning of all objects in the MS COCO dataset increases the scanning speed. In this study, only the person object is emphasized. In this way, models can be made more useful as needed.

YOLO v3-Tiny and YOLO v4-Tiny models have lower object recognition rates than YOLO v3, YOLO v4, and YOLO v5s, as the number of layers and weights is significantly reduced. However, it is sufficient in terms of FPS and speed and does not strain the computer system. Therefore, the YOLO v4-Tiny model can be used on low-performance devices. Real-time object detection performance of YOLO v3 and YOLO v4 models decreases on low-equipped computers. Therefore, they are not recommended for use in low-equipped systems. In the YOLO v5s model, satisfactory results were obtained in terms of accuracy, video read time, and FPS values, despite the low weight size and high number of layers. Therefore, the YOLO v5s model can be used on low- and high-performance devices.

In future studies, it is planned to measure the distance between people, inspired by the approach developed in this study. In this way, it will be investigated whether the social distance between people is among reasonable values. In addition, the application and comparison of one-stage (YOLO, SSD, RetinaNet, etc.) and two-stage (R-CNN, Fast R-CNN, Faster R-CNN, etc.) object detection architectures to the problems mentioned here will be discussed.

## Declarations

**Conflict of interest** The authors did not receive support from any organization for the submitted work. The authors declare they have no financial interests. The authors have no conflicts of interest to declare that are relevant to the content of this article.

## References

- WHO: Coronavirus disease (COVID-19) advice for the public (2020). Retrieved July 15, 2022, from: <https://www.who.int/emergencies/diseases/novelcoronavirus-2019/advice-for-public>
- Akhtar, N., & Mian, A.: Threat of adversarial attacks on deep learning in computer vision: a survey. In IEEE Access (C. 6, ss. 14410–14430). Institute of Electrical and Electronics Engineers Inc. (2018)
- Cho, K., van Merriënboer, B., Bahdanau, D., Bengio, Y.: On the properties of neural machine translation: Encoder–decoder approaches. Proceedings of SSST 2014- 8th Workshop on Syntax, Semantics and Structure in Statistical Translation. (2014)
- Young, T., Hazarika, D., Poria, S., & Cambria, E.: Recent trends in deep learning based natural language processing [Review Article]. In IEEE Computational Intelligence Magazine (C. 13, Sayı 3, ss. 55–75). Institute of Electrical and Electronics Engineers Inc. (2018)
- Bayat, S., Işık, G.: Recognition of Aras bird species from their voices with deep learning methods. J. Inst. Sci. Technol. **12**(3), 1250–1263 (2022). <https://doi.org/10.21597/jist.1124674>
- Işık, G., Artuner, H.: Turkish dialect recognition using acoustic and phonotactic features in deep learning architectures. Int. J. Inform. Technol. **13**, 207–216 (2020)
- Pacal, I., Karaman, A., Karaboga, D., Akay, B., Basturk, A., Nalbantoglu, U., Coskun, S.: An efficient real-time colonic polyp detection with YOLO algorithms trained by using negative samples and large datasets. Comput. Biol. Med. **141**, 105031 (2022). (ISSN 0010-4825)
- Chen K., and Kämäräinen J. K.: Learning to count with back-propagated information, In 22nd International Conference on Pattern Recognition. IEEE, 2014. p. 4672–4677. (2014)
- Redmon, J., Divvala, S., Girshick, R., and Farhadi, A.: You only look once: Unified, real-time object detection, in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 779–788. (2016)
- Gao, C., Li, P., Zhang, Y., Liu, J., Wang, L.: People counting based on head detection combining Adaboost and CNN in crowded surveillance environment. Neurocomputing **208**, 108–116 (2016)
- Punn, N. S., Sonbhadra, S. K., Agarwal, S., Rai, G.: Monitoring COVID-19 social distancing with person detection and tracking via fine-tuned YOLO v3 and Deepsort techniques (2020). arXiv preprint [arXiv:2005.01385](https://arxiv.org/abs/2005.01385)
- Wojke, N., Bewley, A., Paulus, D.: Simple online and realtime tracking with a deep association metric. In 2017 IEEE International Conference on Image Processing (ICIP) pp. 3645– 3649. IEEE. (2017)
- Redmon, J., & Farhadi, A.: Yolov3: An incremental improvement. (2018). arXiv preprint [arXiv:1804.02767](https://arxiv.org/abs/1804.02767)
- Rahim, A., Maqbool, A., Rana, T.: Monitoring social distancing under various low light conditions with deep learning and a single motionless time of flight camera. PLoS ONE **16**(2), e0247440 (2021)
- Bochkovskiy, A., Wang, C. Y., Liao, H. Y. M.: Yolov4: optimal speed and accuracy of object detection. (2020). arXiv preprint [arXiv:2004.10934](https://arxiv.org/abs/2004.10934)
- Ahmad, M., Ahmed, I., Adnan, A.: Overhead view person detection using YOLO. In 2019 IEEE 10th annual ubiquitous computing, electronics & mobile communication conference (UEMCON) pp. 0627–0633. (2019)
- Jocher, G.; Changyu, L.; Hogan, A.; Yu, L.; Rai, P.; Sullivan, T. ultralytics/yolov5: Initial Release (v1.0). Zenodo (2020). <https://doi.org/10.5281/zenodo.3908560>
- Purwar, R.K., Verma, S.: Analytical study of YOLO and its various versions in crowd counting. In: Intelligent data communication technologies and internet of things, pp. 975–989. Springer (2022)
- Ren, P., Fang, W., Djahel, S.: A novel YOLO-Based real-time people counting approach, In: 2017 international smart cities conference (ISC2). IEEE, p. 1–2. (2017)
- Menon, A., Omman, B., Asha, S.: Pedestrian Counting Using Yolo V3. In 2021 International Conference on Innovative Trends in Information Technology (ICITIIT) (pp. 1–9). IEEE. (2021)
- Hu, R., Mo, Q., Xie, Y., Xu, Y., Chen, J., Yang, Y., Wu, E.Q.: Avmsn: an audio-visual two stream crowd counting framework under low-quality conditions. IEEE Access **9**, 80500–80510 (2021)
- Hu, R., Tang, Z.R., Wu, E.Q., Mo, Q., Yang, R., Li, J.: RDC-SAL: refine distance compensating with quantum scale-aware learning for crowd counting and localization. Appl. Intell. (2022). <https://doi.org/10.1007/s10489-022-03238-4>
- Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., Berg, A. C.: SSD: single shot multibox detector. In European conference on computer vision. Springer, Cham. pp. 21–37. (2016)
- Deng, J., Dong, W., Socher, R., L.-J. Li, K. Li, Fei-Fei, L.: ImageNet: a large-scale hierarchical image database. IEEE Computer Vision and Pattern Recognition (CVPR). (2009)
- Jiang, Z., Zhao, L., Li, S., Jia, Y. Real-time object detection method based on improved YOLOv4-tiny. (2020). [arXiv:2011.04244](https://arxiv.org/abs/2011.04244)
- Yu, J., Zhang, W.: Face mask wearing detection algorithm based on improved YOLO-v4. Sensors **21**(9), 3263 (2021)
- Mahto, P., Garg, P., Seth, P., Panda, J.: Refining yolov4 for vehicle detection. Int J Adv Res Eng Technol (IJARET) **11**(5), 409–419 (2020)
- Pacal, I., Karaboga, D.: A robust real-time deep learning based automatic polyp detection system. Comput Biol Med **134**, 104519 (2021). (ISSN 0010-4825)
- Wang, C. Y., Liao, H. Y. M., Wu, Y. H., Chen, P. Y., Hsieh, J. W., Yeh, I. H.: CSPNet: A new backbone that can enhance learning capability of CNN. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops pp. 390–391. (2020)
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Chintala, S.: PyTorch: an imperative style, high-performance deep learning library. In Advances in Neural Information Processing Systems 32 pp. 8024–8035. (2019)
- Liu, S., Qi, L., Qin, H., Shi, J., & Jia, J. (2018). Path aggregation network for instance segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition pp. 8759–8768.
- Wan, J., Chen, B., Yu, Y.: Polyp detection from colorectum images by using attentive YOLOv5. Diagnostics **11**(12), 2264 (2021)
- Mastromichalakis, S.: ALReLU: A different approach on Leaky ReLU activation function to improve Neural Networks Performance. (2020). arXiv preprint [arXiv:2012.07564](https://arxiv.org/abs/2012.07564)
- Zhou, F., Zhao, H., Nie, Z.: Safety helmet detection based on YOLOv5, 2021 IEEE International Conference on Power Electronics, Computer Applications (ICPECA), pp. 6–11. (2021)

35. He, K., Zhang, X., Ren, S., Sun, J.: Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **37**(9), 1904–1916 (2015)
36. Lin T.Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L. and Dollár, P.: Microsoft COCO: common objects in Context. *Computer Vision and Pattern Recognition*, 2–8. (2015)
37. Jünger, M., Mellmann, H., Spranger, M.: Improving vision-based distance measurements using reference objects. In: *Robot soccer world cup*, pp. 89–100. Springer, Heidelberg (2007)
38. Ali, İ., Arslan, N.: Estimated anthropometric measurements of Turkish adults and effects of age and geographical regions. *Int. J. Ind. Ergon.* **39**(5), 860–865 (2009)
39. People passing by-free HD stock footage-no copyright-people walking mall-YouTube, [online] Retrieved October 1, 2022, from: <https://www.youtube.com/watch?v=L3-XaUzaJHU>
40. Valencia IJC, Dadios EP, Fillone AM, Puno JCV, Baldovino RG, Billones RKC (2021) Vision-based crowd counting and social distancing monitoring using Tiny-YOLOv4 and DeepSORT. In 2021 IEEE International Smart Cities Conference (ISC2). IEEE. pp. 1–7

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.

**Mehmet Şirin Gündüz** has a master's degree in Mechatronics Engineering. Mehmet has been working as an Informatics Teacher for 10 years.

**Gültekin Işık** is an assistant professor in the Department of Computer Engineering at Iğdır University, Turkey since May 2019. He received his Ph.D. degree from the Department of Computer Engineering, Hacettepe University, in Feb 2019 under the supervision of Assoc. Prof. Harun Artuner. His research interests are; machine learning, artificial neural networks (deep learning) and language/dialect/signal recognition.