SN Computer Science (2023) 4:718
https://doi.org/10.1007/s42979-023-02131-2

ORIGINAL RESEARCH

Public Social Distance Monitoring System Using Object Detection YOLO Deep Learning Algorithm

Vijayan R[1] · Mareeswari V[1] · Vedant Pople[2]

Received: 20 January 2023 / Accepted: 8 July 2023

Abstract

Computer vision and deep learning are emerging technologies as the backbone system to maintain the public healthcare sector to detect the object and surrounding, especially during the COVID-19 pandemic. Generally, in a single stage, you only look once version 3 (YOLOv3) algorithms promising the best results to detect the object in images, live feeds, or videos by learning features at a faster rate than two-stage algorithms such as R-CNN, fast CNN, and faster CNN. Deep sort methods were employed to track identified people by supporting bounding boxes and calculating the Euclidian distances between the people to maintain social distance. Moreover, the YOLOV3 model requires more computational cost to detect the object at best with a lower detection time. Hence, it motivates us to practice a single graphics processing unit (GPU) with the multithreaded approach to increase the frames per second at detection. The proposed model uses a background modeling method grounded on frame variance accumulation which is used to define the number of frames and weight updating. This approach uses two steps, localization of the object and then the classification of localized objects. Distances between people are calculated and compared with threshold values to facilitate comparison. The threshold limit triggers the alert system which is accessible to people, monitoring many video streams at a time. The model is tested based on processors, threads consumed, and various types of inputs ranging from static images to moving videos. Tiny-YOLOv3 performs with the best frames per second and the least processing time, followed by SPP-YOLOv3 and YOLOv3. The model proves its evidence on various parameters and metrics to work robustly. As well as the reason to adopt YOLOv3 over other YOLOv4 and YOLOV5 is tabulated. This model initiates the curiosity to develop a mobile application with security systems based on IoT and CCTV to monitor crowded places.

This article is part of the topical collection "Research Trends in Computational Intelligence" guest edited by Anshul Verma, Pradeepika Verma, Vivek Kumar Singh and S. Karthikeyan.

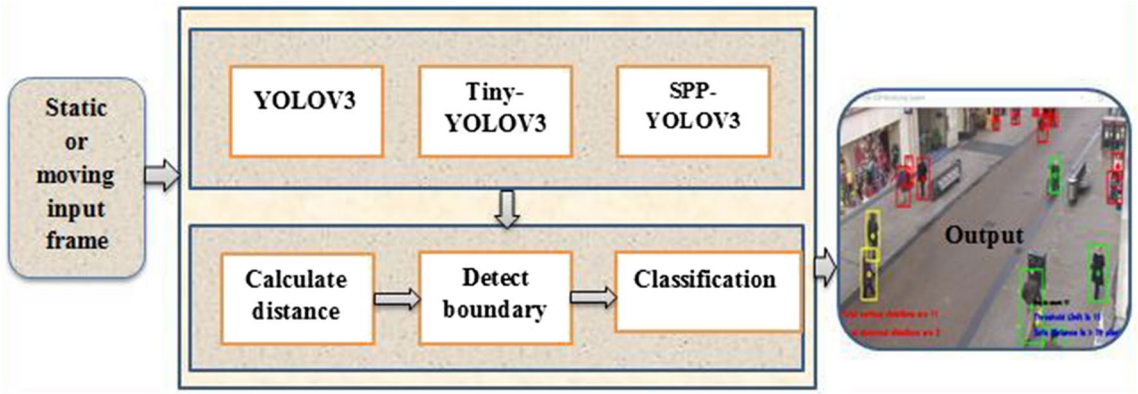
✉ Mareeswari V
vmareeswari@vit.ac.in

[1] School of Computer Science Engineering and Information Systems (SCORE), Vellore Institute of Technology (VIT), Vellore 632014, India

[2] Fulton Schools of Engineering, Arizona State University, Tempe, AZ, USA

Published online: 23 September 2023

Graphical Abstract



Keywords YOLO · Object detection · Object classification · Social monitoring · Deep learning · Computer vision

Introduction

Coronavirus is a group of enveloped viruses, which are known to infect human hosts and cause respiratory disease [1]. A coronavirus disease well-known in 2019, SARS-CoV-2, has triggered a pandemic of respirational disease, titled COVID-19. Preventive measures like wearing a face mask, sanitizing hands, and maintaining social distancing are important [2] and advised by the world health organization (WHO). Rapid transmission is the deadliest trait possessed by the COVID-19 virus, making social distancing the most important measure of them. This would also reduce the burden on the healthcare systems and retain the economy progressing. Commercials are essential to maintain the standard operating procedure (SOP) to control public transmission of the infection.

In overcrowded, indoor, or outdoor settings, it is challenging to manually check for social distance violations. As a result, computers may be utilized to track busy areas using cameras and ensure that the social separation guideline is being followed properly. For this, it is necessary to first identify the people in the video clip, second, determine the social distance between the identified individuals, and third, mark those who adhere to the one-meter social distancing rule as well as those who do not with various colors. People are made sure to abide by the social distance guideline in this way. Only those in the line for payments or entrance/exit at a mall should be required to conform to keep the social distance norand rem. The current work provides an alternative approach for identifying instances of social distancing among people who are captured in a specific region of a video frame. As a result, just a chosen area was treated, while the remaining areas were left out of the dimension, rather than the complete area seen in the video frame. The social distance between the persons in the photograph is found and tracked using object recognition techniques. It is the process of locating and classifying items in an image or video. The first step of this process is to make a bounding box to separate the object of concern and depending on the properties of the confined region, identify the object's class or type. Deep learning methods for object recognition have gained popularity recently. The three most popular deep learning-based object identification techniques are Region-based Convolutional Neural Network (R-CNN) [20], Single Shot Detector (SSD) [3], and You Only Look Once (YOLO) [32].

YOLO belongs to the single-stage object recognition technology that makes use of convolutional neural networks in deep learning. The first version of the YOLO model was created by Redmon et al. [32]. In recent times, a YOLO object detection model is used to isolate humans and classify them [4]. To identify individuals and determine their spatial separation from one another combined the YOLO v3 algorithm with the Deepsort technique. Their study demonstrated the superiority of the Deepsort algorithm and YOLO v3 over other object-detecting methods. Deepsort methods are practiced to monitor recognized people with the support of bounding boxes and assign an ID [5]. Distances between all the pairs of people in the frame are computed simultaneously to make sure all the cases in the input frame are covered. The model uses various parameters like confidence and threshold limits to calculate the criteria for classification. The scope of the project ranges from being used as a single camera classification model, which can later be integrated into the CCTV surveillance systems and then also be there for small resource starving systems [6]. Faster R-CNN, SSD, and

YOLOv3 are compared for object detection by practicing on Google Earth photos, the DOTA dataset [7], and remote sensing images [8] obtained from the GF-1 and GF-2 satellites. According to the results of this study, YOLOv3 has a higher mAP and FPS than SSD and Faster R-CNN models. When YOLOv3 is compared with Mask R-CNN, YOLOv3 works three times better than Mask R-CNN [9].

These authors [10] employed the YOLOV4 algorithm to monitor the social distance in the low-light environment grounded on the video frames captured through motion-less cameras. YOLOV3 produces 95% accuracy when applied on the frame captured in the top view to identifying the person [11]. This [12] social distance monitoring system for the COVID-19 time of year was constructed using the YOLOv4-tiny algorithm. The YOLOv5 algorithm was introduced in 2020 and it has been successfully employed for counting applications in settings with sparse crowds. This work innovates the version YOLO-PC used to construct a system for counting people [13] and achieves higher accuracy. In general, YOLO and SSDalgorithm works better for social distance monitoring [14]. [15] examines YOLOv3, YOLOv4, and YOLOv5. The algorithms are trained and tested using the MS COCOdataset. It was discovered that YOLOv5 performs better concerning accuracy than YOLOv4 and YOLOv3. When compared to YOLOv4 and YOLOv5, YOLOv3 had a faster detection speed, while YOLOv4 and YOLOv5 had the same speed. According to the latest findings [16], the YOLO v3 model exhibited its most impressive accuracy (87.07%) and mAP (89.91%). In reverse, the YOLO v5s model managed to attain the maximum frame rate of up to 18.71. Also, they compared the superior model YOLOv3 and the faster model YOLOv5 with SSDand RestinaNet.

Moreover, YOLOv3 consumes greater computational costs to produce decent detection performance. To improve the object detection process's frames per second (fps), a more reliable method needs to be developed. Hence, it motivates me to practice the multithreaded YOLO approach on a single graphics processing unit (GPU). The primary contribution is the suggestion of a multi-thread method using YOLOv3 to carry out real-time object recognition in massive amounts of video feeds and compared it with other single-thread approaches, as well as comparative research studies on other YOLOv3 variants. Hence, this approach practices the YOLOv3 model that can form the bases for a variety of applications, not limited to social distancing during the COVID-19 pandemic, but to sparser day-to-day chores like maintaining distancing at places like railway ticket booking and religious places.

The following list explains the flow of this proposed article:

- It starts with a review of the social distance monitoring system using YOLO models,
- Shows the system architecture of YOLOv3, TINY-YOLOv3, SPP-YOLO architecture and tabulates configuration settings,
- Illustrates the flow of the proposed working model to depict the steps to detect social distance,
- The various results and their comparative analysis graphs are shown in the results and discussion section.
- Tabulates the latest comparative study on YOLOv3, v4, and v5 and its discussion to stimulate the proposed work.
- Tabulates the recent comparative efficiency analysis based on FPS, CPU usage, GPU power, memory usage, and energy consumption on YOLO models used in intelligent applications running in 5G edge devices.
- It comprises the detection and classification results of single-thread yolov3 illustrated by single-thread serious violations, single-thread abnormal violations, and single-thread moving camera source. It is followed by YOLOv3 single thread graph according to time vs frames per second and time versus processing time.
- Similarly, single-thread tiny-YOLOV3 and single-thread SPP-YOLOV3 have analyzed static output and moving output and their comparison.
- Moreover, alike, multi-thread YOLOV3, multi-thread tiny-YOLOv3, and multi-thread SPP-YOLOv3 have been demonstrated and analyzed.
- Tabulates and visualized the comparative analysis of existing YOLO models.
- The article ends with the conclusion of YOLO models and future practices.

Henceforth, the proposed model is evaluated based on the number of processors, threads used, and different sorts of inputs, including still images and moving video streams. Tiny-YOLOv3, SPP-YOLOv3, and YOLOv3 all perform well in terms of frames per second and processing time. The model backs up its claims with evidence based on a variety of measures and characteristics. This model initiates the curiosity to develop a mobile application with security systems based on IoT and CCTV to monitor crowded places.

Literature Review

Social distancing maintenance is the application of object detection using OpenCV. Earlier works in this domain face general problems like noise in input, a high number of false positives, and redundancy of classified objects in the box. The research approach in [11, 17], and [18] focuses on using regional convolutional neural networks and comparing single-stage and two-stage detection. The proposed models improve multi-task joint optimization and multi-model

information fusion for object prediction, detection, and monitoring. A similar approach is used with a sliding window algorithm [19] for detection and tracking. CNN architectures are categorized into different techniques based on feature map exploitation, spatial exploitation, width, depth, multi-path, channel boosting, and attention [6]. Models use 2 convolutional layers to identify regions and pooling to resize feature maps, and perform classification and regression [20].

The algorithm [21] proposes a GPR hybrid model to improve performance to enhance localization accuracy. Retrieving previous results is done using rHOG [22], to determine stationary objects along with occlusion and abrupt changes. Other methods [23] formulate crowd as Binary Quadratic Programming, to formulate the target's information as the custom of motion, spatial proximity, and grouping constraints and resolves revealing and data association in chorus. Background modeling and subtraction generate the area of interest. A matching score is proposed to get detection results [24]. This method is compared with making mutable size bounding boxes with diverse orientations, concerning radial distance of the image [25].

The object detection part is followed by calculating the spaces between each set of people distinguished. This approach of social removing assessment denotes the people who are drawing nearer than a reasonable cutoff red. The approach involves sliding window-based region proposals in phases, detection, and tracking and monitoring one after the other [19]. Fixed pixels in the captured footage are also used to make the calculations. Certain inaccuracies are caused by anomalies in the social distance measuring method based on the detection box midpoint point values [4]. Faster R-CNN is used for unique, rapid, and precise methods of object detection, while only YOLO can work with live video feeds [26]. Another approach is using a variety of CNN which are different in feature-map exploitation, spatial exploitation, width, depth, multi-path, channel boosting, and attention. Some problems with CNN are overfitting, additional factors than thin deep networks, great spatial and time complexity, and considering the residual facts for defining the weight of every channel [6]. Advanced YOLOv1, has a novel inception model configuration, a focused pooling pyramid layer, with enhanced performance. A model maps high-dimensional to low-dimensional data, hence introducing bias.

The classification categories are not compatible with each other, making it also inadequate [27]. Since single shot multi-box detector (SSD) is grounded on the total eradication of the progression that produces a plan, it is a straightforward approach that needs an object proposal. Additionally, it does away with the following pixel and resampling phases. On the contrary, this model in place of utilizing anchor boxes to group features and assesses another classifier, produces a value for every object class in all boxes, which requires a lot of computation [28]. To make real-time embedded object identification easier, TINY SSD was created. It consists of significantly enhanced layers made up of a collection of non-uniform SSD-based auxiliary convolutional feature layers and a non-uniform fire sub-network. The gap here is reflected in the performance in the long term, since the arguments are denoted by half-precision floating-point numbers, the modal size of the deep neural network is further reduced, which affects the accuracy of object detection [29]. Apart from models, some classifiers use computer vision to calculate the distance between each person in the frame and generate patterns to classify them

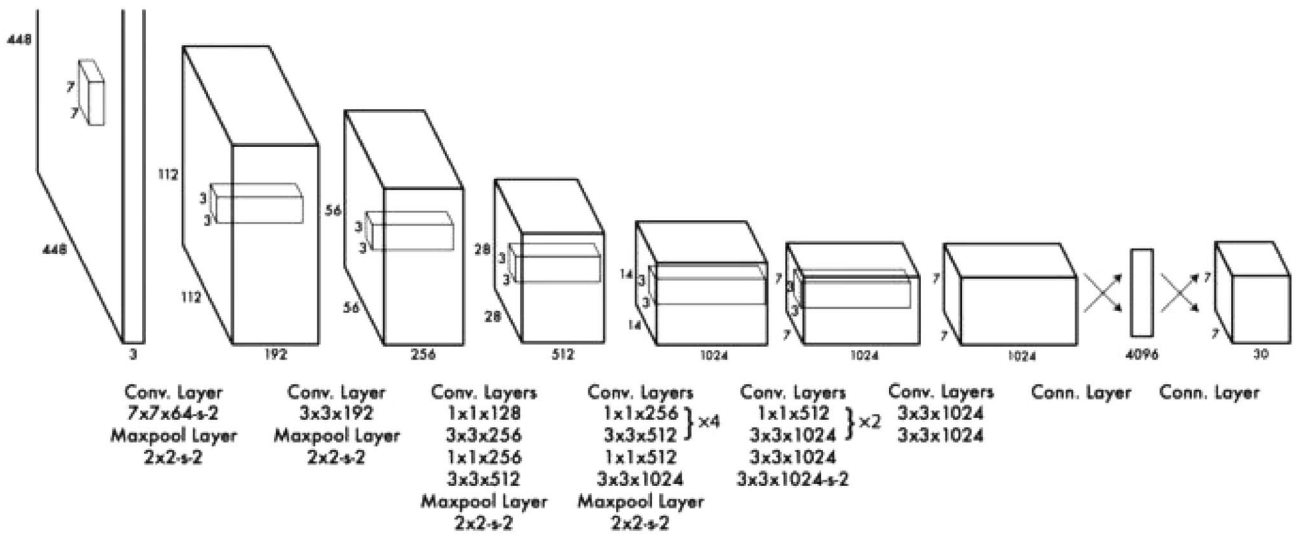


Fig. 1 YOLO architecture

as "Safe" or "Unsafe", displaying the tags as the result of object detection and classification. This classifier can be integrated with surveillance systems. The drawback of this method is that YOLO detects a pedestrian as an object while the partial parts of the body are observable, generating error due to overlapping frames [3]. Since the launch of Darknet, YOLOv3 was among the best enhancements made to an object-detecting system. Reviewers and other industry experts gave this improved version very favorable reviews. However, it has flaws of its own. YOLOv3 is still regarded as an experienced model, although the complexity analysis revealed problems and lacked ideal solutions for the loss function. Subsequently, it was fixed in an improved version of the same model, which was utilized and evaluated for operational improvements [30].

Some systems have been designed to make the whole artificial intelligence concepts for tracing people, social distancing, and checking the temperature. This method trains data from thermal cameras and is implemented on a network of devices that are monitored from a central system. This method gets promising results for people detection, working on fewer resources and making it more efficient [31]. The system sends alerts about unmasked people to respective authorities. The model produces feature maps and usages of two convolutional layers to find the region. The pooling method is used to resize feature maps, to perform classification and regression [20]. Another approach detects an object and distinguishes it as an issue of regression, based on input of sizes, anchored boxes, and extraction network functions. This is different as less optimized boxes are utilized to calculate the last bounding box for each object, rather than selecting the single with the most confidence [32]. A template-matching algorithm is used to mark specific areas. The intersection over union ratio is calculated concerning the spotted person and the area of interest to make a decision [33].

The YOLOv3-spp model is most successful in detecting the objects as compared to YOLOv3 and YOLOv3-tiny. Apart from this, the models yield outcomes in the form of model evaluation and accuracy aspect, and it makes to be more appropriate with great performance on terrestrial videos. To increase the accuracy factor, the authors must work on training and verification procedures for the dataset. In parallel, the most effective detection could be grasped with the objective research drives [34]. A dense connection strengthens the extracting feature and alleviates the vanishing gradient problem and also pools and concatenates the features of the multi-scale local region, to enhance comprehensive learning. The model implements cross-entropy as an alternative to the mean-square error to symbolize object classification damage [35]. Another common approach is using multiple sections with varied expansion ratios to create characteristic maps with diverse receptive fields. Branch outputs are combined to incorporate multi-scale data. The features are extracted from three altered measures and combine low-resolution features using a top-down pathway and lateral connections [36]. Some models use a spatial pyramid pooling (SPP) component presented at the stem of the locator for the size-tuning of the model values. Multi-scaled locale high points were created after pooling and link of the SPP. In addition to that DenseNet structure, many functional layers are joined to integrate additional functional information [37].

The 2-staged detector approaches, RCNN, Fast RCNN, and Faster RCNN are overviewed, while single-stage detectors like YOLO v1, v2, v3, and SSD are covered. The 2-staged detector focuses on accuracy, while the most important about stage detectors is speed. On the other hand, 2-stage detectors afford sufficient accuracy, the computation time is high [38]. Derived from Tiny-YOLOv3, Tinier-YOLO achieves better detection accuracy and efficient real-time outcomes while reducing model size. The SqueezeNet fire module is employed by inspecting the many fire modules along with their places to reduce the number of model parameters decreasing the size of the model. Tinier-YOLO introduces dense connections to reinforce the feature propagation and confirm the extreme facts in the network [39].

The latest introduction of YOLOv5 detects large, small, and tiny objects. They implement a multi-scale method to acquire deep discriminate feature depictions at diverse measures and routinely decide the best appropriate measures for detecting objects in a frame. This modification results in a reduction of many multi-scale parameters compared to the original architecture, resulting in an improvement of precision by a large margin [40]. Specifically created to reduce model size while maintaining object detection performance, Tiny-SSD is a single-shot detection deep CNN for present-day embedded object detection. It is pretended of a much optimized, non-uniform fire

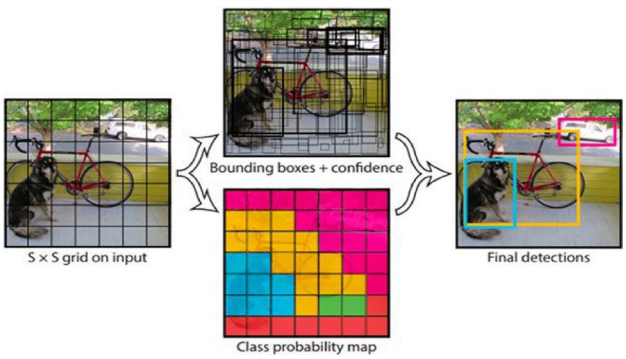


Fig. 2 YOLO test result

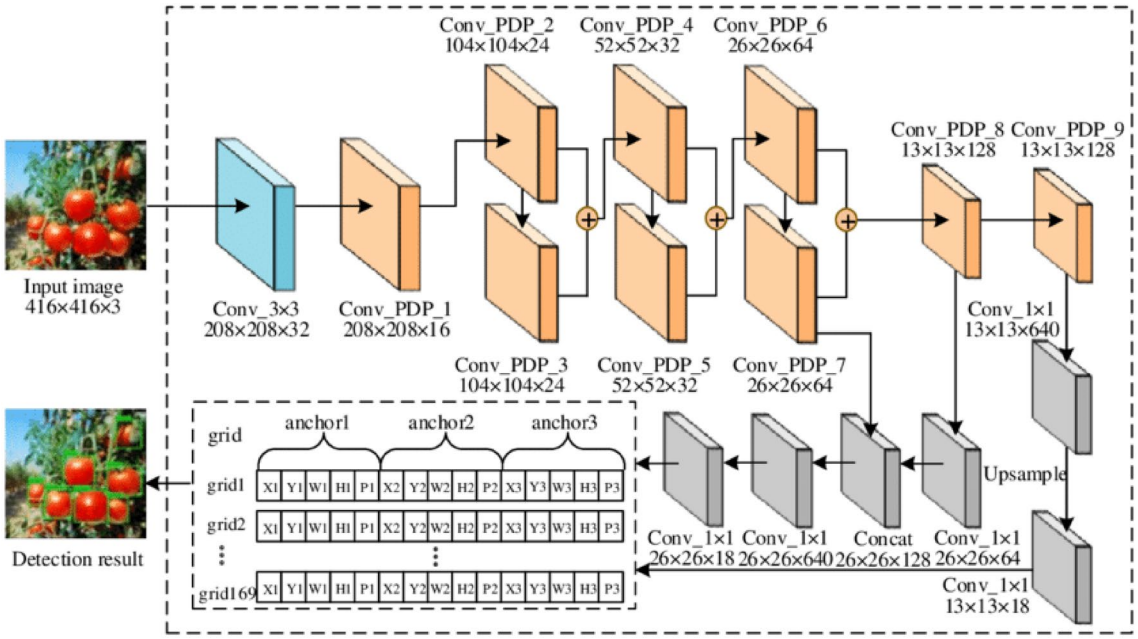


Fig. 3 Tiny-YOLOv3 network

sub-network stack and a non-uniform sub-network stack of extremely optimized SSD-based auxiliary convolutional feature layers. For Tiny-SSD, the arguments are denoted in a half-precision floating-point number layout, which will lead to promoting deep neural network model size reductions whereas devising an insignificant consequence on object detection accuracy [41].

The related comparison of real-time deep learning algorithms from the literature review concluded that YOLOv3 outperforms other deep learning algorithms like Mask-CNN, Faster R-CNN, and SSD in terms of accuracy and speed. According to the literature, YOLOv4 is likewise more accurate than YOLOv3; however, the stated accuracy of YOLOv4 against YOLOv5 is still debatable, as some authors contend that YOLOv4 is superior to YOLOv5, while others contend that YOLOv5 is superior. YOLOv3 works with greater detection speed than YOLOv4 and YOLOv5. Numerous factors, including the various datasets utilized and altered hyperparameters, might be blamed for the disparate reported outcomes. These variations result from specific uses that other researchers have examined. Hence, this proposed work tries to compare the YOLOv3 model with different variations of static and moving inputs.

Materials and Methods

YOLOv3 System Architecture

The YOLO architecture (Fig.1) frames object to be a regression downside to spatially divided bounding boxes and related classes. A neural network forecasts bounding boxes and sophistication probabilities and confidences openly through complete pictures in a single evaluation. Hence the complete detection channel may be a particular network; it will be enhanced end-to-end openly on detection performance.

The YOLO detection network comprises 24 convolutional layers trailed by 2 completely coupled layers. Interspersing 1 × 1 convolutional layer reduces the feature area from previous layers. In the ImageNet classification, the convolutional layers have pertained to work at half the resolution (224 × 224 input image) and then double the resolution for detection. The starting convolutional layers of the network extract feature from the image when the completely linked layers forecast the output probabilities and coordinates. The quick interpretation of YOLO deliberates to drive the boundaries of rapid object detection. The speedy YOLO practices a neural network with lesser convolutional layers (9 rather than 24) and minimum filters in those layers. This network produces the last output (Fig. 2) the 7 × 7 × 30 [42] tensor of predictions.

This design separates the picture into a framework of S*S size. Assuming the focal point of the jumping box of

the article is in that matrix, then, at that point, this network is liable for recognizing that item. Every network predicts jumping boxes with its certainty score. Every certainty score shows how exactly it is that the jumping that predicts contains an article and how exactly it predicts the bounding box organizes concerning ground truth expectation. On the off chance that there is an article present in the picture, the certainty score is equivalent to IoU between ground truth and anticipated boxes. The likelihood is restrictive given the presence of an item in the lattice cell. In any case, the quantity of boxes in every lattice cell predicts just a single bunch of class probabilities.

Scores are generated by multiplying conditional class probabilities and box confidence predictions that produce class-specific confidence for every box. Scores denote both the probability of that class and the prediction [43] that fits the object.

TINY-YOLOv3 Architecture

The Tiny YOLOv3 architecture (Fig. 3) is a shortened variety of the existing YOLOv3 model. The Tiny-YOLOv3 model reduces the number of convolutional layers and uses only 7 convolutional layers for feature extraction. It uses the pooling layers, with a step size of 2 to obtain dimensionality lessening as compared to convolutional layers in YOLOv3 [38]. As compared to YOLOv3 the main advantage is that the network is simple (Fig. 3), the calculation is small, and can run on resource-starving devices. The disadvantage is low accuracy, both in candidate frame and classification accuracy.

The Tiny-YOLOv3 uses the training and loss function (Eq. 1) same as YOLOv3, which mainly is composed of prediction frame position, prediction frame size, prediction class, and prediction confidence [39]. The loss is given by:

$$\frac{1}{n}\sum_{i=1}^{n}\text{loss\_}xy+\frac{1}{n}\sum_{i=1}^{n}\text{loss\_}wh+\frac{1}{n}\sum_{i=1}^{n}\text{loss\_class}+\frac{1}{n}\sum_{i=1}^{n}\text{loss\_confidence} \quad (1)$$

SPP-YOLO Architecture

This architecture (Table 1) presents a better spatial pyramid pool (SPP) to extricate the multi-scale nearby area elements of the items, use the cross entropy to address characterization misfortune and construct and train the model to detect the objects.

Table 1 YOLO-SPP architecture

Layers	Parameters		Output
	Filters	Size/stride	
Conv 1	32	3 × 3/1	416 × 416 × 32
Maxpool 1		2 × 2/2	208 × 208 × 32
Conv 2	64	3 × 3/1	208 × 208 × 64
Maxpool 2		2 × 2/2	104 × 104 × 64
Conv 3	128	3 × 3/1	104 × 104 × 128
Conv 4	64	1 × 1/1	104 × 104 × 64
Conv 5	128	3 × 3/1	104 × 104 × 128
Maxpool 3		2 × 2/2	52 × 52 × 128
Conv 6	256	3 × 3/1	52 × 52 × 256
Conv 7	128	1 × 1/1	52 × 52 × 128
Conv 8	256	3 × 3/1	52 × 52 × 256
Maxpool 4		2 × 2/2	26 × 26 × 256
Conv 9-12	512	3 × 3/1l × 2	Conv 31
Conv 13	512	1 × 1/1 l	26 × 26 × 512
Maxpool 5		3 × 3/1	13 × 13 × 512
		2 × 2/2	
DC Block	1024	3 × 3/1 l × 4	13 × 13 × 2304
Conv 14-21	256 or 512	1 × 1/1	
Conv 22	1024	3 × 3/1	13 × 13 × 1024
Conv 23	512	1 × 1/1	13 × 13 × 512
SPP Block	512	5 × 5/1 l	13 × 13 × 2048
Maxpool 6-8		7 × 7/1 l	13 × 13 × 512
Conv 26		Concat	
		13 × 13/1 l	
		1 × 1/1	
Conv 27	1024	3 × 3/1	13 × 13 × 1024
Reorg Conv 13	1024	/2	13 × 13 × 256
Concat -1, -2		3 × 3/1	13 × 13 × 1280
Conv 30			13 × 13 × 1024
Conv 31	K*5+C	1 × 1/1	13 × 13 ×
Detection			(K*5+C)

The network (Fig. 4) consists of a multi-scale object discovery block, a spatial pyramid pooling block with 3 max-pooling layers, a dense connection block with four dense units, and five laminated convolution-pooling blocks. The SPP block with three max-pooling layers is presented next to the DC block for connecting the nearby area highlights and is removed and met by multi-scale pooling [35]. The Spatial Pyramid Pooling approach improves the existing system by optimizing the connection structure of the backbone network and introduces a multi-scale local region feature extraction, helping in achieving object detection speed that is close to YOLOv2, and higher than conventional methods like De-convolutional Single Shot Detector (DSSD), Scale-transferable Detection Network (STDN) and YOLOv3 [35].

The new loss function is constructed adopts to illustrate the defeat of object detection, we use the cross-entropy of object classification and the mean squared error of the

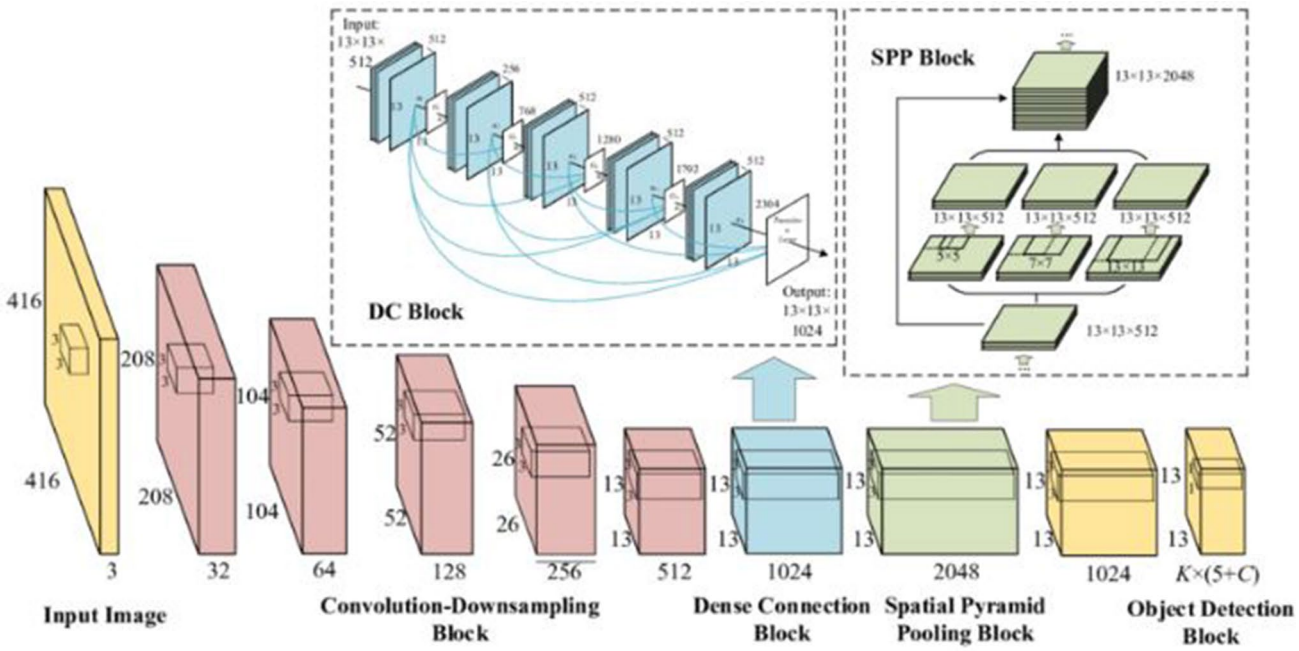


Fig. 4 YOLO-SPP model

coordinate regression. The vanishing-gradient issue can be resolved and the training model can be made robust using cross-entropy to describe the object classification defeat.

Implementation

The model flowchart (Fig. 5) receives the input frame at the beginning of the model. This input frame is checked for the presence of a person by the model. If there is a person in the frame the model continues processing otherwise the frame is discarded. The frame is then transformed according to the view making it ideal for generating the results. If there are more than 2 people detected in the frame, the pairwise distances between all the people are calculated. This calculated distance along with the centroid coordinates and coordinates of bounding boxes around them is calculated.

As the distance between the coordinates is changed with the receiving of continuous frames the distance calculated between people is checked against the MIN_DIST, which is the threshold for the distance between 2 people for maintaining Social Distancing. The frame and color of the Circle are then determined by this factor. The bounding box is given green color if the distance calculated is more than the threshold value or given yellow color if the distance is in the range of more than permitted and less than the threshold value. If the calculated distance is less than MIN_DIST, the bounding box is presented with red color. The red-colored box also increases the number of people violating the norms counter. After this, the frame is looped again for updating the coordinates continuously. This process is repeated by the model until a person is present in the frame. The indices are updated according to the presence of the people, finally ending the process when the input frames are finished or stopped.

Results and Discussion

The results and analysis of all the models are done with respect to two parameters namely, frames per second (FPS) and processing Time (Eq. 2). A frame per second is the rate at which the classified output is displayed on the screen and processing time is the difference between the current time and time at which the last frame was processed. These two factors significantly influence the results of this real-time classification model.

$$\text{Processing time} = \text{Current time} - \text{Time required to process the last frame}$$

$$\text{Frames per second} = 1/(\text{Processing time}) \tag{2}$$

According to the findings [16], the properties of the YOLO models v3, v4, and v5 and the average performance outcomes done on the 3 videos used for the comparative study are given in Table 2. For this implementation, complete IoU as the loss function and relu and leaky relu as the activation function practiced in all models. With YOLO v3

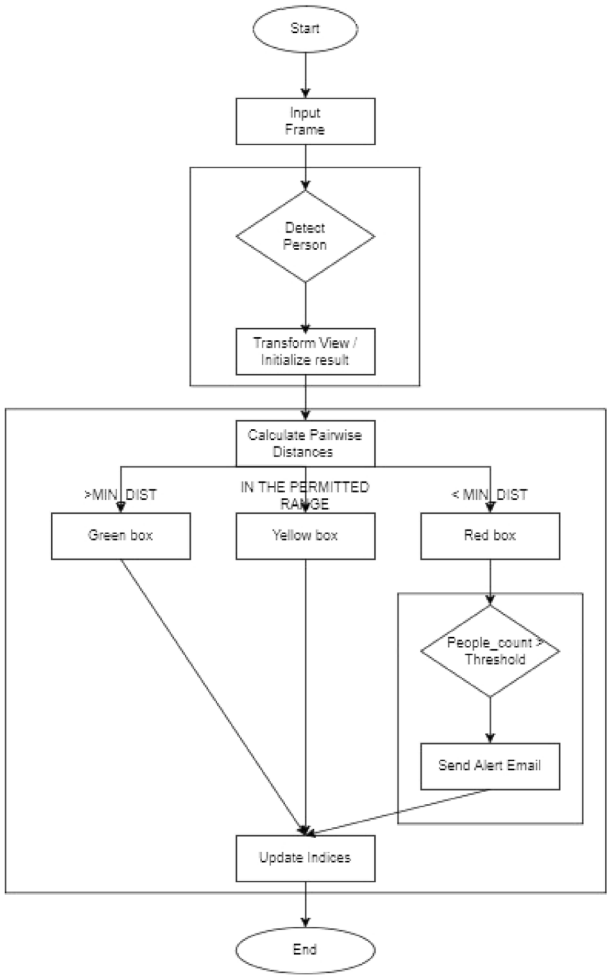


Fig. 5 Model flowchart

Table 2 Comparison on YOLO v3, v4 and v5 models

YOLO variants	Accuracy (%)	Video processing time (S)	mAP(0.5) (%)	FPS
YOLO v3	87.07	35.14	89.91	9.51
YOLO v4	63.79	33.42	63.54	10.49
YOLO v5	73.70	17.48	78.88	18.71

and YOLO v5s, videos were scanned at an average frame rate of 9.51 and 18.71, respectively. YOLO v3 managed to obtain the best accuracy rate of 87.07% despite having the lowest FPS value. The test films were processed by the YOLO v3 algorithm on average in 35.14 s, which suggests that it might have a speed-related performance problem. The YOLO v5s model has been created to solve this issue. The test moving inputs were scanned by the YOLO v5s on average in 17.48 seconds at an FPS of 18.71 with an average accuracy of 73.70%. Concerning the mAP(0.5) metric, the YOLO v3 model attained a superior performance of 89.91% compared with YOLOv4 and YOLOv5. Unfortunately, YOLO v4 results are very poor than v3 and v5 in all comparisons. YOLO v3 and v5s models are superior to one another with respect to mAP, FPS, and accuracy metrics. A trade-off between the two models should be established based on the application. While v5s should be utilized to achieve performance that is almost real-time (i.e., fast speed), V3 should be used if high precision and mAP are needed.

In 5G intelligent solutions, YOLO models are very prevalent in accelerator-based single-board computers such as NVIDIA Jetson Nano, Jetson Xavier NX, and Raspberry Pi 4B with Intel neural compute stick2. It produces greater performance with minimal power consumption to detect the object in AI-based applications. Table 3 noted the performance of YOLOv3 and YOLO-tiny while executed on Jetson Nano to detect the object in two videos for comparison. If the window size (S) is small in the model, then it reaches a high FPS value. It means that increasing frames per second is proportional to increasing the CPU usage to process the frames. Similarly, the same scenario applied to GPU-powered systems will get a small impact on usage. Hence, selecting the CPU or GPU-powered system depends on the level of the task. Likewise, the energy consumption is also reflected as proportional to the S value. It suggests that more concern about the trade-off between inference accuracy and speed while applying AI intelligent applications running on edge devices [44]. Henceforth, the proposed work employed the superior model YOLOv3 with its different processing techniques such as single thread and multi-thread, and compared with the tiny, SPP YOLOv3 and SSD.

Detection and Classification Results

Single-Thread YOLOV3

The single threading detection and classification give smooth output as more of the frames are visible to the user. This also increases the visibility of the user. The frames marked with green color (Fig. 6) are at a safe distance from the user, while the frames that are red mark the pairs of people who are close to each other. The output frame also displays the number of serious violations i.e., 4, abnormal violations, threshold limit, and the safe distance threshold, i.e., 70px.

The current testing video has a stationary camera so more frames are being processed in the classification. This also shows yellow frames for people in the frame (Fig. 7) who are in the range and about to violate the norms. For stationary cameras and single threads, the FPS obtained is 0.628.

In the footage (Fig. 8), the source of the input camera is moving, so more frames are received and more frames need to be processed. This leads to a slight delay in processing

Table 3 Performance of YOLO models on Jetson Nano while resizing window [44]

Models	S Value	FPS	CPU usage (%)	Memory usage (GB)	GPU power (W)	Energy consumption (kJ)
Video1						
YOLOv3	320	2.4	26.8	0.86	3.7	5.43
	608	0.9	25.5	1.19	3.9	16.01
YOLOv3-tiny	320	10.3	30.5	1	3.7	1.18
	608	3.3	27	1.01	3.9	3
Video2						
YOLOv3	320	2.5	30.8	1.31	3.3	3.26
	608	0.8	26.8	1.31	3.4	9.72
YOLOv3-tiny	320	9.9	41.3	1.13	3.3	0.77
	608	3.3	31.8	1.13	3.4	2.18

Fig. 6 Single thread serious violations

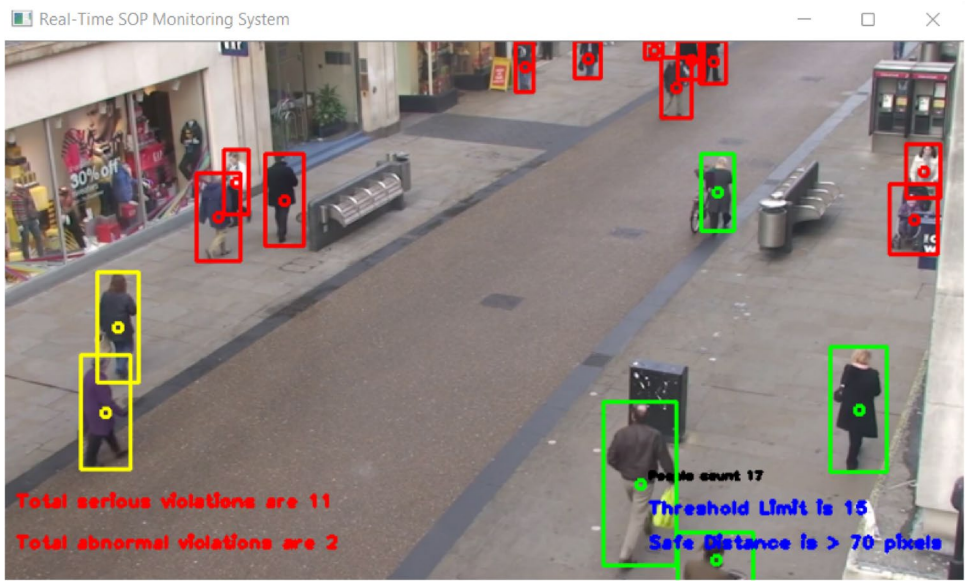


Fig. 7 Single thread abnormal violations

Fig. 8 Single thread moving camera source

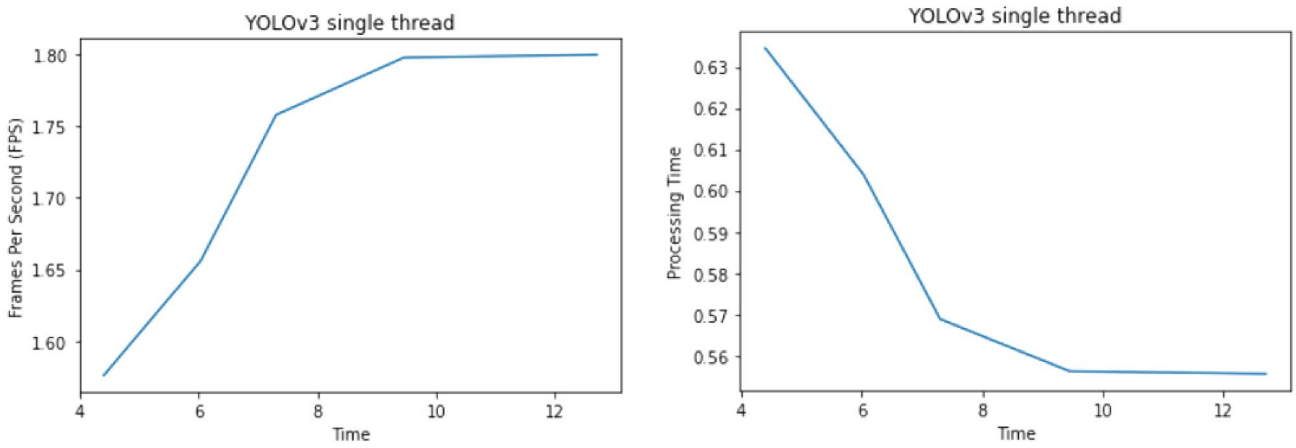


Fig. 9 YOLOv3 Single thread graph

because of moving sources and people in the frame but consistency with the classification, hence the model proves its robustness. The FPS obtained for this test is 0.610.

The graph (Fig. 9) for single-thread FPS concerning time can be observed. FPS increases with time, which portrays that processing speed increases with time.

Single Thread Tiny-YOLOV3

The figure (Fig. 10) shows the result of classification and detection with Tiny-YOLOv3 using a single thread and the camera is static.

In this figure (Fig. 11), the source of input is moving, which results in a slight delay in processing. It is implemented using Tiny-YOLOv3 architecture consuming a single thread.

The graph (Fig. 12) presents the working of Tiny-YOLOv3 with respect to time on both the metrics Frames per second (FPS) and Processing Time.

Single Thread SPP-YOLOV3

The figure (Fig. 13) shows the result of classification and detection with SPP-YOLOv3 working on a single thread and static camera input.

The result of classification and detection with SPP-YOLOv3 working on a single thread and moving camera

Fig. 10 Single thread Tiny-YOLOv3 static output

Fig. 11 Single thread Tiny-YOLOv3 moving output

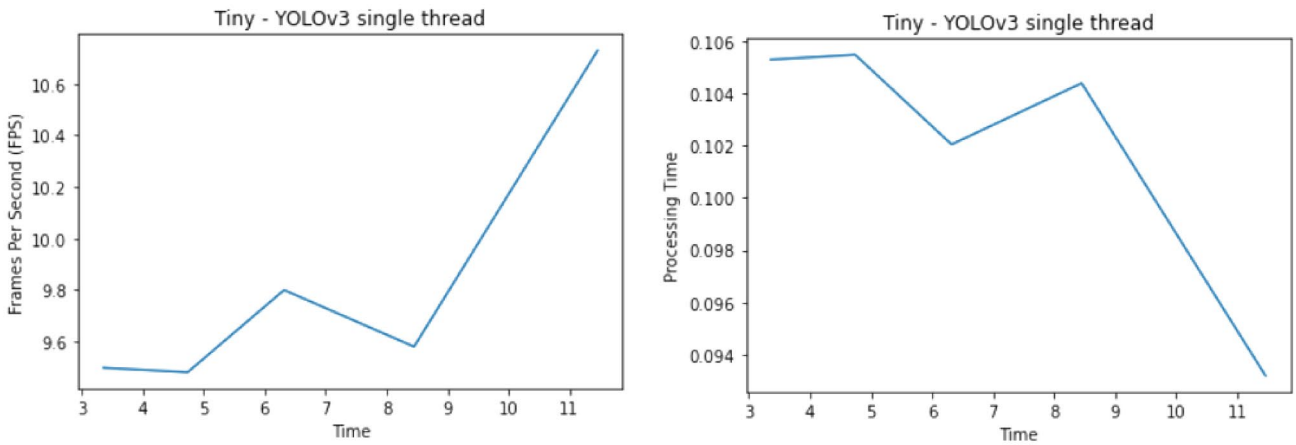


Fig. 12 Tiny-YOLOv3 single thread graph

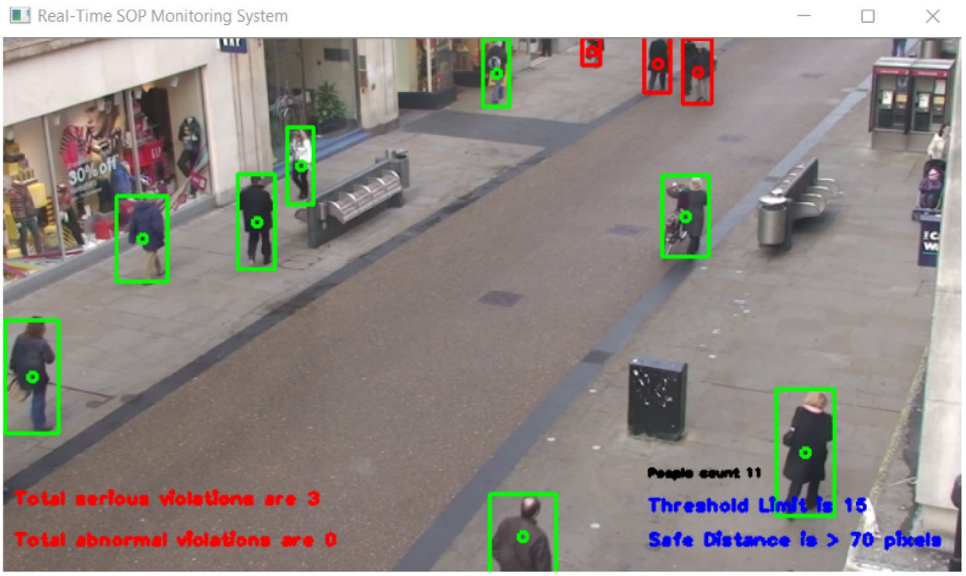


Fig. 13 Single thread SPP-YOLOv3 static output

Fig. 14 Single thread SPP-YOLOv3 moving output

input is illustrated in (Fig. 14). SPP-YOLOv3 works on spatial pooling pyramid architecture and works by increasing the number of spatial layers before feeding to convolutional layers. This results in increased efficiency in classification.

The results of detection and classification are hence concluded in the following graph (Fig. 15) as well. The FPS values range from 1 to 1.5 for classification and processing time from 1.0 to 0.7.

Multi-Thread YOLOV3

The Multi-threading model is implemented to get quicker results and more efficiency in processing power with the existing resources (Fig. 16). As the frame processing computations increase more people are classified in quick successions leading to better results. This test video is of a static camera-captured frame. The FPS obtained for this test video is 0.3559.

The test video with a moving frame in multi-thread performs as compared to the test frame with a single-threaded test video. This test video (Fig. 17) also is a testament to the fact that, as the source of the input frame is moving, more people are moving relatively, and hence more frames are generated, still, there is a comparable amount of processing done. The FPS received for the same case is 0.4129.

The YOLOv3 multi-thread output graph (Fig. 18) shows the result of frames per second (FPS) and processing time with respect to time.

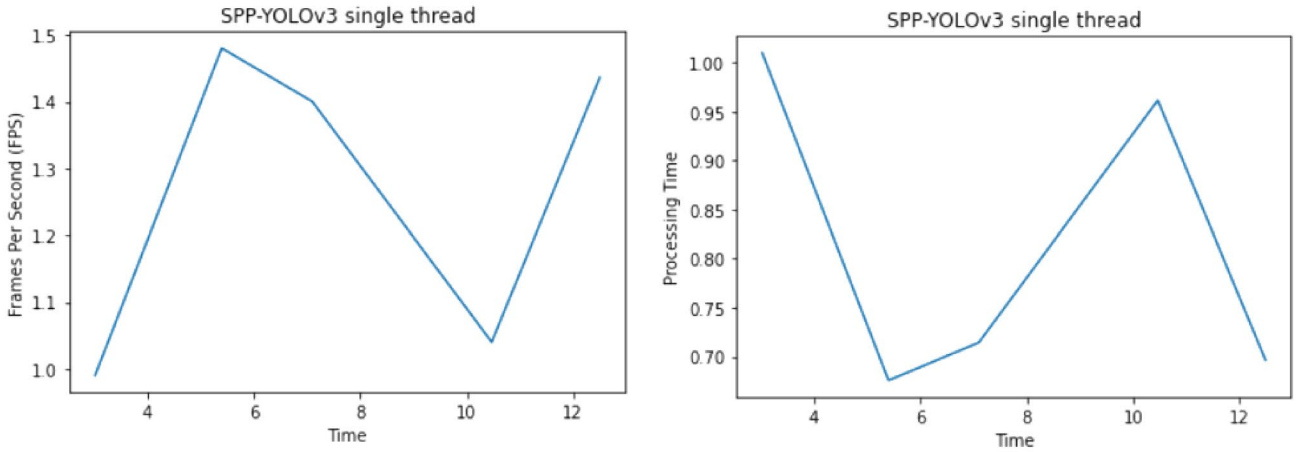
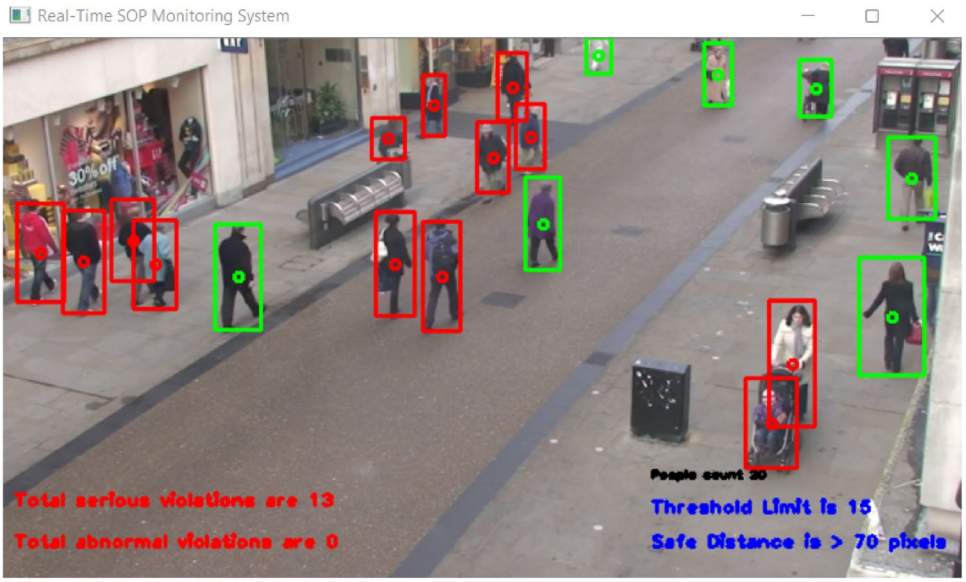


Fig. 15 SPP-YOLOv3 single thread graph

Fig. 16 Multi-thread serious violations YOLOv3 static



Multi-Thread Tiny-YOLOv3

The Tiny-YOLOv3 with multi-thread performs the fastest as it is a lightweight architecture. The result of detection and classification can be seen in the image (Fig. 19).

The Tiny-YOLOv3 shows the best results in moving camera sources (Fig. 20) as well, as more classifications can be processed easily with multi-threading and small architecture. Tiny-YOLOv3 performs the best for classification, boosted by the presence of multi-tasking making it more stable and efficient [38]. Multi-threading makes efficient resource distribution.

The graph (Fig. 21) visualizes the results of Tiny-YOLOv3 consuming multiple threads.

Multi-Thread SPP-YOLOv3

SPP-YOLOv3 performs detection and classification with ease, due to the SPP detector attached at the beginning of the convolutional layers. The image (Fig. 22) is the result of classification with a multi-thread system and static camera input.

The SPP-YOLOv3 architecture performs well while consuming multiple threads and with a moving camera input frame. The figure 23 presents the output of SPP-YOLOv3 with a moving source of input.

The graph (Fig. 24) shows a linear increase in the FPS and vice versa in Processing time suggesting that the processing time of the frames reduces with time and makes it more efficient.

Fig. 17 Multi-thread YOLOv3 moving input

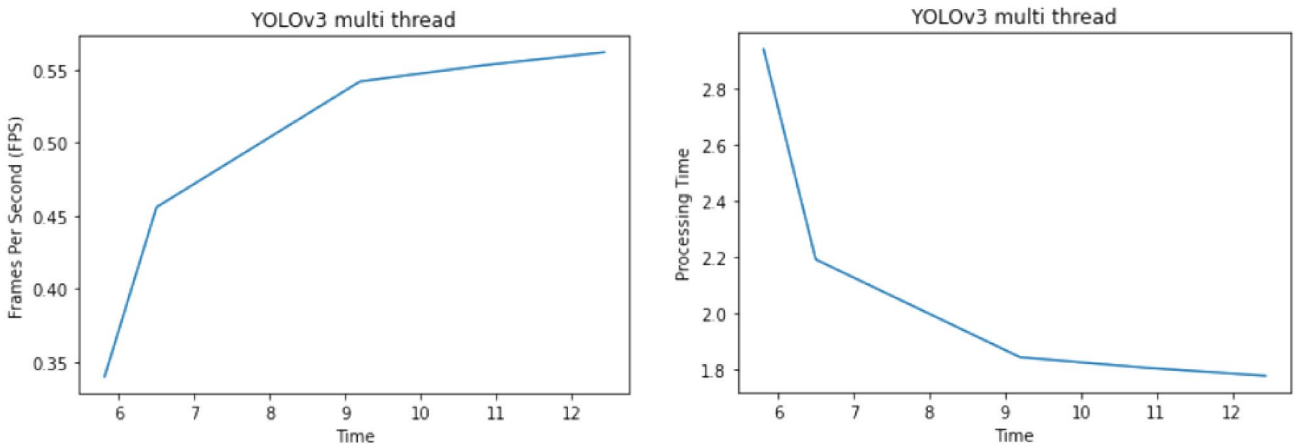


Fig. 18 YOLOv3 multi-thread output graph

Fig. 19 Multi-thread Tiny-YOLOv3 static input

Fig. 20 Multi tread Tiny-YOLOv3 moving input

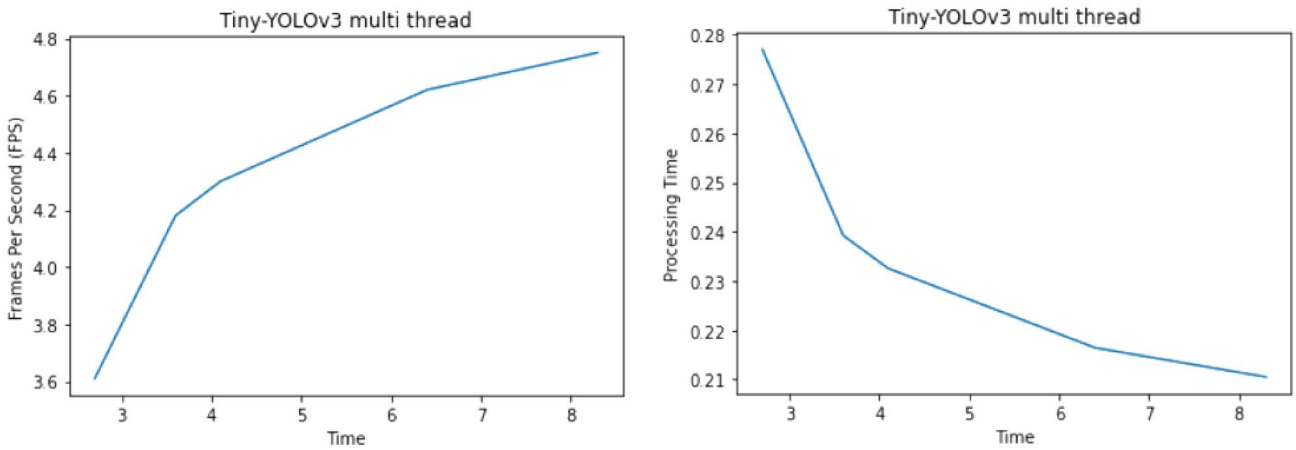


Fig. 21 Tiny-YOLOv3 multi-thread output graph

Fig. 22 Multi-thread SPP-YOLOv3 static input

Fig. 23 Multi-thread SPP-YOLOv3 moving input

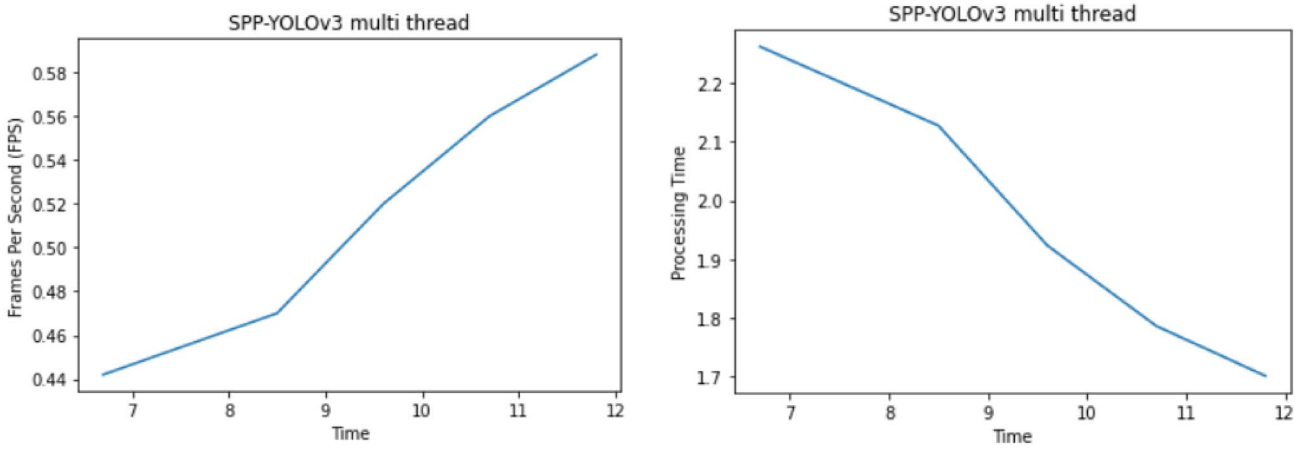


Fig. 24 SPP-YOLOv3 multi-thread output graph

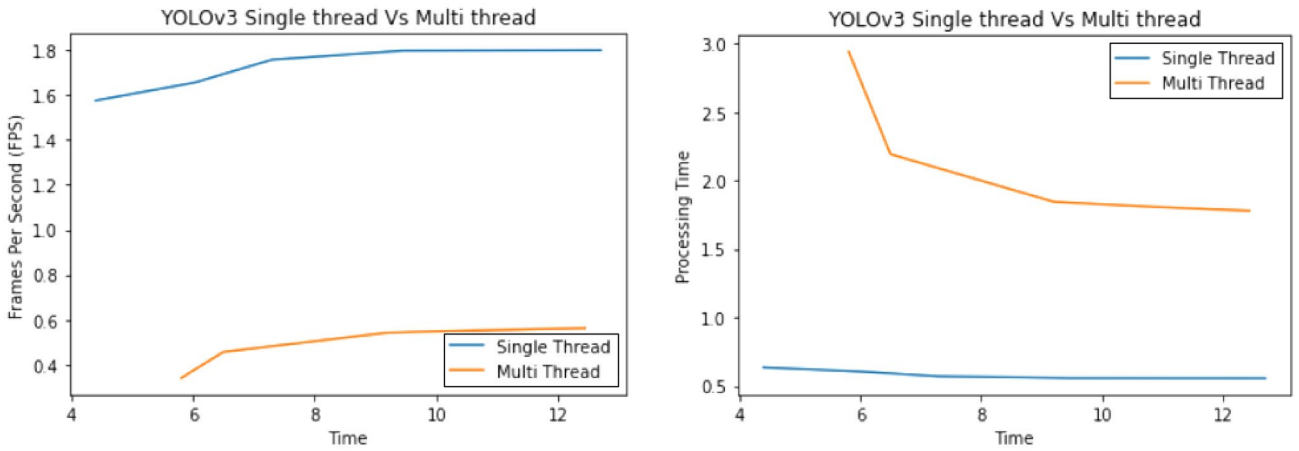


Fig. 25 YOLOv3 comparison

Table 4 Comparison with existing models

Model	Frames per second (FPS)		Processing time	
	Existing	Obtained	Existing	Obtained
YOLOv3	3.5	1.6	0.283	0.625
Tiny-YOLOv3	6.4	9.5	0.156	0.105
SPP-YOLOv3	2	1.5	0.500	0.667
Single Shot Detection	1.7	1.8	0.588	0.556

Bi

Comparative Analysis

The models are tested with Intel Core i5 10th generation processor with integrated GPU, trained and weighted on MS COCO dataset. The models are compared with various existing state of art models.

The YOLOv3 executes single-thread and multi-thread programs efficiently (Fig. 25). The difference in the speeds is observed in the following graph. Single thread achieves competitively more FPS as compared to multi-thread. This underlines the fact that processing time for a single thread is less as compared to multi-thread. The range of FPS is 1.5 to 1.8 for a single thread, and 0.3 to 0.6 for multi-thread (Table 4).

Tiny YOLOv3 has the fastest processing speed among all YOLOv3 models tested (Fig. 26). The execution on single thread and multi-thread are most efficient, compared to the rest. The reduced architecture and parameters make it feasible for development in mobile and embedded devices. As observed from the graph, Tiny YOLOv3 FPS results range from 0.2 to 0.6 in a multi-thread as compared to 1.0 to 1.4 in a single thread.

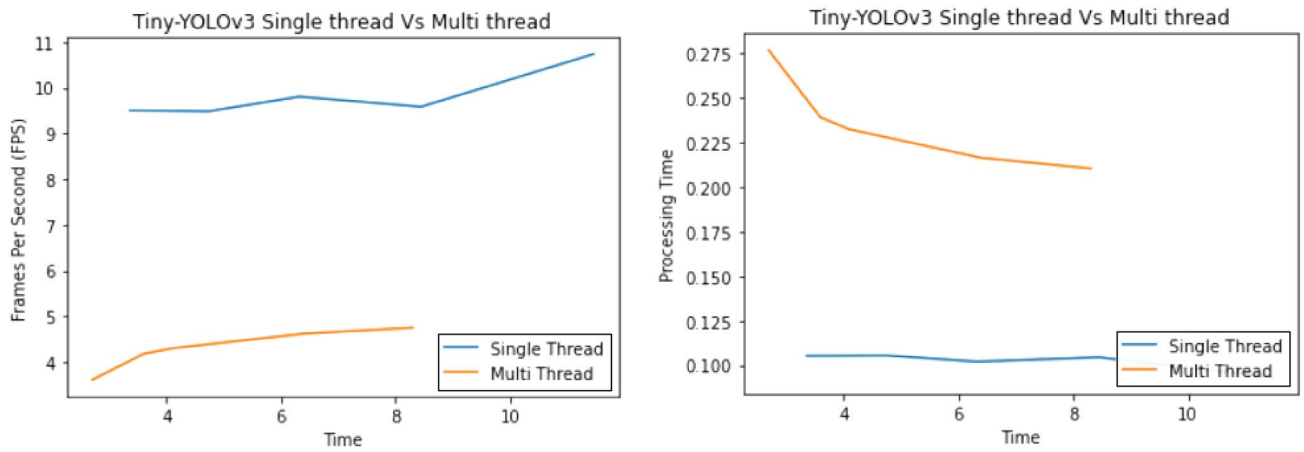


Fig. 26 Tiny-YOLOv3 comparisons

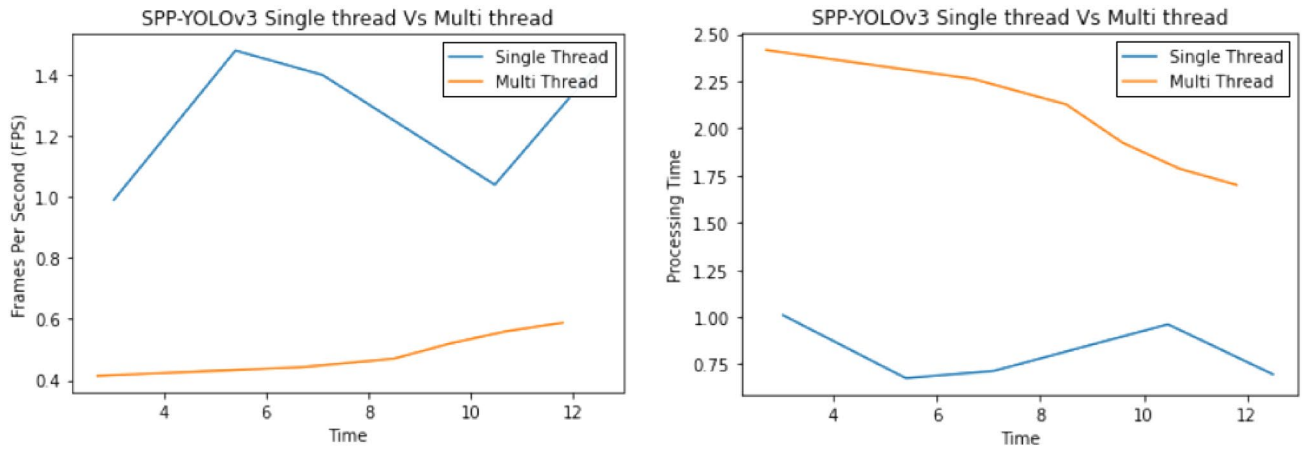


Fig. 27 SPP-YOLOv3 comparisons

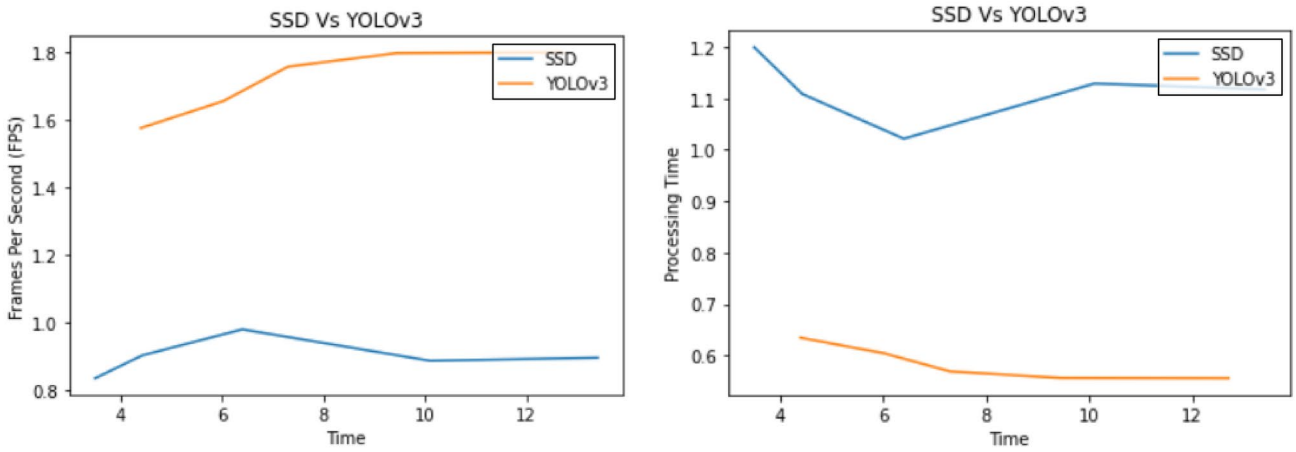


Fig. 28 SSD and YOLOV3 comparison

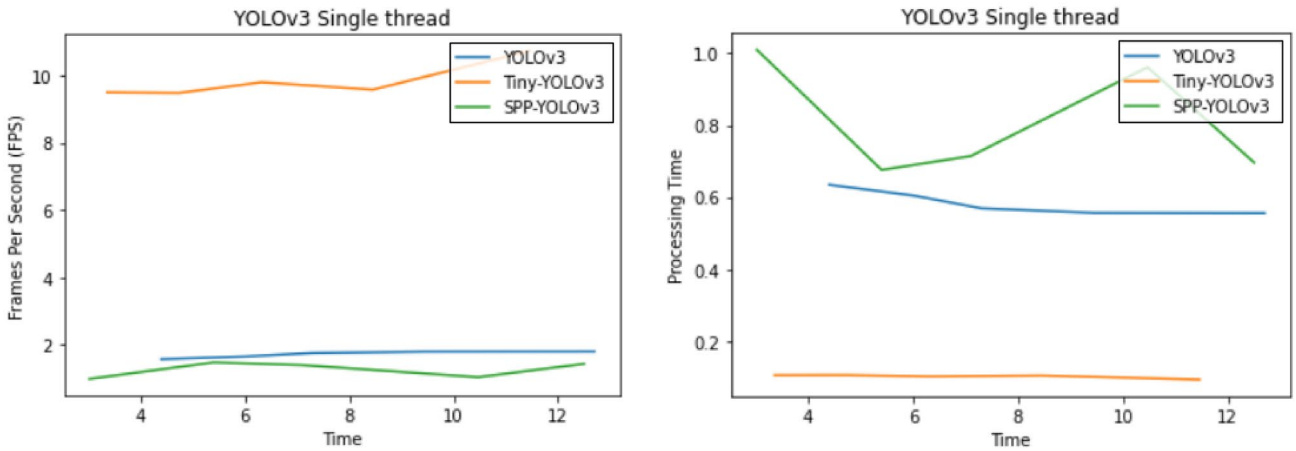


Fig. 29 YOLOv3 single thread comparison

The SPP-YOLOv3 uses 3 different types of max pool sizes for the same images. The model also collects a feature map called "Fixed length representation" before feeding to fixed connected convolutional layers. The range of FPS in single-threaded lies from 1 to 1.4, as compared to 0.4 to 0.6 in multi-thread (Fig. 27).

SSD and YOLOv3, both are single-stage detectors and perform object detection and classification on the given input frame. The fundamental difference in the use of parameters, convolutional layers at the output, receptive fields, aspect ratio, and various grid sizes to calculate confidence values in them leads to the difference in the results [41]. The FPS for YOLOv3 is observed to be higher than the SSD at all points in time in the graphs (Fig. 28).

The YOLOv3 is single-stage detection architecture. On a single thread, all the YOLOv3 models perform invariably to each other. Tiny-YOLOv3 is the best architecture for object detection and classification as it has the highest FPS in a single thread followed by YOLOv3 and SPP-YOLOv3 as observed in the graphs (Fig. 29).

The multi-thread models consumed more than one thread, making the process of detection and classification parallel. This results in more processing time for the input frame as the detections and results have to be rearranged in the final output frame. The Tiny-YOLOv3

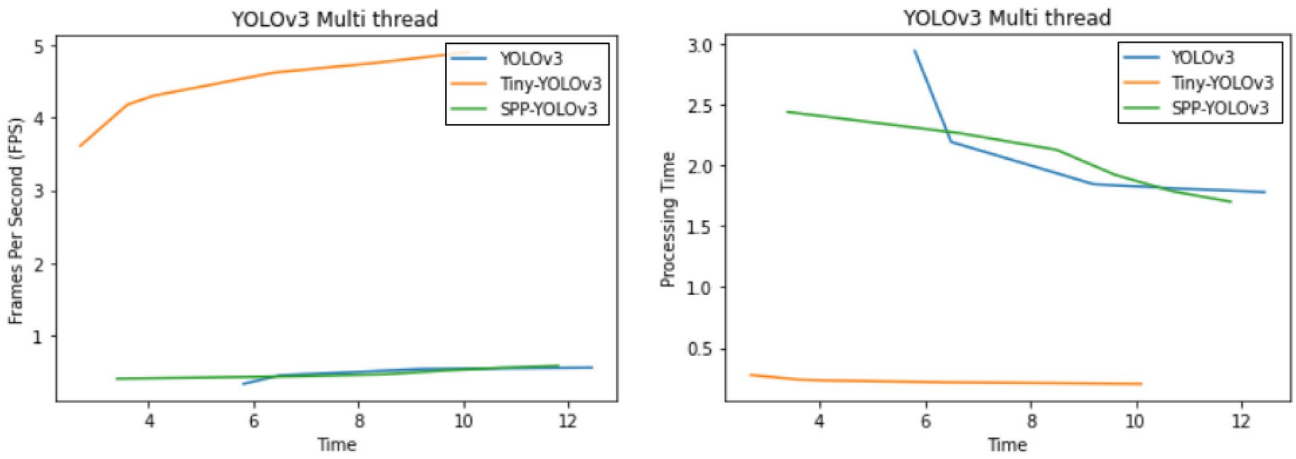


Fig. 30 YOLOv3 multi-thread comparison

again beats its subordinate architecture models. YOLOv3 and SPP-YOLOv3 have comparatively lower FPS and hence higher processing time as compared to Tiny-YOLOv3 [40] as observed from the graph (see Fig. 30).

Conclusion

With the waves of coronavirus, the world has strict rules to avoid gathering people close in such places as restaurants, industries, transportation, real estate, and agriculture. Hence, this proposed model can ensure social distancing in public places using object detection and classification. The proposed model is evaluated based on the number of processors, threads used, and different sorts of inputs, including still images and moving video streams. This model also has the option to use multi-threading for more efficient results using GPU to reduce the computation overhead. The alert system is handy for people, monitoring many video streams at a time, as alerts are generated after the threshold limit is reached. Apart from this, the FPS metric portrays the performance of each model on different static and moving video inputs.

As the result of a comparative study, Tiny-YOLOv3 performs with the best frames per second and the least processing time, followed by SPP-YOLOv3 and YOLOv3. The model proves its evidence on various parameters and metrics to work robustly. Also, the proposed work tabulated the reason to practice YOLOv3 than the latest YOLOv4 and YOLOv5 models according to the latest research work. It strongly recommends that to adopt YOLOv3 to achieve highly accurate results in a person detection system.

For future work, this model can be implemented inside mobile applications and other resource-starving IoT devices. Integration with CCTV cameras can provide an easy surveillance solution and monitor easier for security authorities. Applications of this model can be moreover used at crowded places like railway stations ticket booking centers, huge public gatherings, and religious ceremonies.

Declarations

Conflict of Interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Unhale SS, et al. A review on corona virus (COVID-19). World J Pharm life Sci. 2020;6(4):109–15.
2. Harapan H, et al. Coronavirus disease 2019 (COVID-19): a literature review. J Infect Public Health. 2020;13(5):667–73.
3. Liu W, et al. "Ssd: Single shot multibox detector. European conference on computer vision. Cham: Springer International Publishing; 2016. p. 21–37.
4. Ahmed I, Ahmad A, Piccialli F, Sangaiah AK, Jeon G. A robust features-based person tracker for overhead views in industrial environment. IEEE Internet Things J. 2017;5(3):1598–605.
5. Ahmed I, Ahmad Mi, Ahmad A, Jeon G. Top view multiple people tracking by detection using deep SORT and YOLOv3 with transfer learning: within 5G infrastructure. Int J Mach Learn Cybern. 2021. https://doi.org/10.1007/s13042-020-01220-5.
6. Choi J-W, Moon D, Yoo J-H. Robust multi-person tracking for real-time intelligent video surveillance. ETRI J. 2015;37(3):551–61.
7. Zhao K, Ren X. Small aircraft detection in remote sensing images based on YOLOv3. IOP Conf Ser Mater Sci Eng. 2019;533(1):12056.
8. Li M, Zhang Z, Lei L, Wang X, Guo X. Agricultural greenhouses detection in high-resolution satellite images based on convolutional neural networks: comparison of faster R-CNN, YOLO v3 and SSD. Sensors. 2020;20(17):4938.

9. Dorrer MG, Tolmacheva AE. Comparison of the YOLOv3 and Mask R-CNN architectures' efficiency in the smart refrigerator's computer vision. J Phys Conf Ser. 2020;1679(4):42022.
10. 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. 2021;16(2): e0247440.
11. Ahamad AH, Zaini N, Latip MFA. "Person detection for social distancing and safety violation alert based on segmented ROI." In: 2020 10th IEEE international conference on control system, computing and engineering (ICCSCE), Penang, Malaysia, 2020. p. 113–118.
12. 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. J Real-Time Image Process. 2022;19(3):551–63.
13. Ren P, Fang W, Djahel S. "A novel YOLO-Based real-time people counting approach." In: 2017 international smart cities conference (ISC2), Wuxi, China. 2017. p. 1–2.
14. Shorfuzzaman M, Hossain MS, Alhamid MF. Towards the sustainable development of smart cities through mass video surveillance: a response to the COVID-19 pandemic. Sustain Cities Soc. 2021;64: 102582.
15. Ge Z, Liu S, Wang F, Li Z, Sun J. "Yolox: exceeding yolo series in 2021." arXiv Prepr. arXiv2107.08430. 2021.
16. Gündüz MS, Işık G. A new YOLO-based method for social distancing from real-time videos. Neural Comput Appl. 2023. https://doi.org/10.1007/s00521-023-08556-3.
17. Khan A, Sohail A, Zahoora U, Qureshi AS. A survey of the recent architectures of deep convolutional neural networks. Artif Intell Rev. 2020;53(8):5455–516.
18. Zhao Z-Q, Zheng P, Xu S, Wu X. Object detection with deep learning: a review. IEEE Trans neural networks Learn Syst. 2019;30(11):3212–32.
19. Zhang G, Wang P, Chen H, Zhang L. Wireless indoor localization using convolutional neural network and Gaussian process regression. Sensors. 2019;19(11):2508.
20. Shalini GV, Margret MK, Niraimathi MJS, Subashree S. Social distancing analyzer using computer vision and deep learning. J Phys Conf Ser. 2021;1916(1):12039.
21. Ren S, He K, Girshick R, Sun J. Faster R-CNN: towards real-time object detection with region proposal networks. IEEE Transact Patt Anal Mach Intell. 39(6):1137–1149. https://doi.org/10.1109/TPAMI.2016.2577031.
22. Mareeswari V, Vijayan R, Sathiyamoorthy E, Ephzibah EP. A narrative review of medical image processing by deep learning models: origin to COVID-19. Int J Adv Technol Eng Explor. 2022;9:623–43.
23. Nguyen CT, et al. A comprehensive survey of enabling and emerging technologies for social distancing Part I: fundamentals and enabling technologies. Ieee Access. 2020;8:153479–507.
24. Ansari M, Singh DK. Monitoring social distancing through human detection for preventing/reducing COVID spread. Int J Inf Technol. 2021;13(3):1255–64.
25. Saponara S, Elhanashi A, Gagliardi A. Implementing a real-time, AI-based, people detection and social distancing measuring system for Covid-19. J Real-Time Image Process. 2021;18(6):1937–47.
26. Dehghan A, Shah M. Binary quadratic programing for online tracking of hundreds of people in extremely crowded scenes. IEEE Trans Pattern Anal Mach Intell. 2017;40(3):568–81.
27. Ahmed I, Adnan A. A robust algorithm for detecting people in overhead views. Cluster Comput. 2018;21(1):633–54.
28. Srivastava S, Divekar AV, Anilkumar C, Naik I, Kulkarni V, Pattabiraman V. Comparative analysis of deep learning image detection algorithms. J Big Data. 2021;8(1):1–27.
29. Ahmad T, Ma Y, Yahya M, Ahmad B, Nazir S. Object detection through modified YOLO neural network. Sci Program. 2020;2020:1–10.
30. Huang Y-Q, Zheng J-C, Sun S-D, Yang C-F, Liu J. Optimized YOLOv3 algorithm and its application in traffic flow detections. Appl Sci. 2020;10(9):3079.
31. Womg A, Shafiee MJ, Li F, Chwyl B. Tiny SSD: A tiny single-shot detection deep convolutional neural network for real-time embedded object detection. In: 2018 15th Conference on Computer and Robot Vision (CRV). Toronto, ON, Canada, 2018. p. 95–101.
32. Meivel S, et al. Mask detection and social distance identification using internet of things and faster R-CNN algorithm. Comput Intell Neurosci. 2022;2020:1–13.
33. Suresh K, Bhuvan S, Palangappa MB. Social distance identification using optimized faster region-based convolutional neural network. In: 2021 5th International Conference on Computing Methodologies and Communication (ICCMC). Erode, India, 2021. p. 753–760.
34. Khan MZ, Khan MUG, Saba T, Razzak I, Rehman A, Bahaj SA. Hot-Spot zone detection to tackle COVID19 spread by fusing the traditional machine learning and deep learning approaches of computer vision. Ieee Access. 2021;9:100040–9.
35. Cepni S, Atik ME, Duran Z. Vehicle detection using different deep learning algorithms from image sequence. Balt J Mod Comput. 2020;8(2):347–58.
36. Huang Z, Wang J, Fu X, Yu T, Guo Y, Wang R. DC-SPP-YOLO: Dense connection and spatial pyramid pooling based YOLO for object detection. Inf Sci (Ny). 2020;522:241–58.
37. Zhang X, Gao Y, Wang H, Wang Q. Improve YOLOv3 using dilated spatial pyramid module for multi-scale object detection. Int J Adv Robot Syst. 2020;17(4):1729881420936062.
38. Zhang X, Wang W, Zhao Y, Xie H. An improved YOLOv3 model based on skipping connections and spatial pyramid pooling. Syst Sci Control Eng. 2021;9(sup1):142–9.
39. Adarsh P, Rathi P, Kumar M. YOLO v3-tiny: object detection and recognition using one stage improved model. In: 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS). Tamilnadu, India, 2020. p. 687–694.
40. Fang W, Wang L, Ren P. Tinier-YOLO: a real-time object detection method for constrained environments. IEEE Access. 2019;8:1935–44.
41. Carrasco DP, Rashwan HA, Puig D. T-YOLO: tiny vehicle detection based on YOLO and multi-scale convolutional neural networks. IEEE Transact Pattern Anal Mach Intell. 2021;39(6):1137–49. https://doi.org/10.1109/TPAMI.2016.2577031
42. 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. Las Vegas, NV, USA, p. 779–788.
43. Redmon J, Farhadi A. Yolov3: An incremental improvement. arXiv Prepr. arXiv1804.02767. 2018.
44. Feng H, Mu G, Zhong S, Zhang P, Yuan T. Benchmark analysis of yolo performance on edge intelligence devices. Cryptography. 2022;6(2):16.