Multimedia Tools and Applications (2021) 80:3981–3996
https://doi.org/10.1007/s11042-020-09749-x

Real time object detection and tracking system for video surveillance system

Sudan Jha[1] · Changho Seo[2] · Eunmok Yang[3] · Gyanendra Prasad Joshi[4]

Received: 14 November 2019 / Revised: 28 July 2020 / Accepted: 27 August 2020 /
Published online: 25 September 2020

Abstract

This paper introduces a system capable of real-time video surveillance in low-end edge computing environment by combining object detection tracking algorithm. Recently, the accuracy of object detection has been improved due to the performance of approaches based on deep learning algorithm such as region-based convolutional network, which has two stages for inferencing. One-stage detection algorithms such as single shot detector and you only look once (YOLO) have been developed at the expense of some accuracy and can be used for real-time systems. However, high-performance hardware such as general-purpose graphics processing unit is required to achieve excellent object detection performance and speed. In this study, we propose an approach called N-YOLO which is instead of resizing image step in YOLO algorithm, it divides into fixed size images used in YOLO and merges detection results of each divided sub-image with inference results at different times using correlation-based tracking algorithm the amount of computation for object detection and tracking can be significantly reduced. In addition, we propose a system that can guarantee real-time performance in various edge computing environments by adaptively controlling the cycle of object detection and tracking.

Keywords Object detection · Object tracking · Video surveillance · Edge computing · Artificial intelligence

✉ Gyanendra Prasad Joshi
joshi@sejong.ac.kr

[1] School of Computer Science and Engineering, Lovely Professional University, Phagwara, Punjab 144411, India

[2] Department of Convergence Science, Kongju National University, Gongju 32588, South Korea

[3] Department of Financial Information Security, Kookmin University, Seoul 02707, South Korea

[4] Department of Computer Science and Engineering, Sejong University, Seoul 05006, South Korea

1 Introduction

Object detection has seen substantial improvements during the last couple of years due to the recent advancement in deep learning-based algorithms. Many researchers have investigated object detection by considering each frame of video as image to detect object in video. However, humans do not regard each frame as an independent image in the process of recognizing an object in a video, rather track a major motion in relation to a previous frame.

Thus, video detection technology such as video surveillance, automotive driving, and intelligent robotics is a combination of object detection and object tracking technologies. Recently, algorithms such as region-based convolutional network (R-CNN), single shot detector (SSD), and you only look once (YOLO) have emerged, which show remarkable performance in the field of real time object detection [7, 22, 23]. However, very high-performance hardware is required in order to utilize them in the video surveillance industry.

To overcome this issue, various approaches combining object detection and object tracking technology have been proposed in the literature [9, 10]. However, these attempts also require the graphics processing unit (GPU) to meet the minimum standard of 8 fps, which is the human perception of motion. Devices for the real time video surveillance require low-end specifications, therefore is considerably economic in comparison to that of existing deep learning-based systems that required high-end GPUs. Considering these facts, in this work we propose a solution capable of object detection in GPU-less low-end edge device by applying YOLO algorithm and correlation-based object tracking technology.

The main contribution of this study is to propose the new YOLO (N-YOLO) algorithm based on the characteristics of YOLO. The N-YOLO algorithm improves accuracy performance to detect more detailed objects in the same image by focusing on the grid-based bounding box generation method in the YOLO algorithm. The proposed N-YOLO algorithm is not pipelining of object detection algorithm and object tracking algorithm, but it constructs tracker that performs several YOLO inference engines in parallel and merge the results.

In computer vision, object detection has been seen as one of the classical problems. Specially it deals with detecting the objects existing inside a given image and accordingly their position in the image (i.e. where they are located in the image). This problem is categorized into two broad area; i.e. object detection and object classification. Recent researches have shown that the problem of object detection is more complex than problem of classification, where objects are recognized (easily) but does not give the accurate results about its location (in the image). In addition, classification does not work on images containing more than one object.

Theoretical innovation in our paper: - Most of the research works in YOLO follow convolutional neural network (CNN) [17] for object detection in real-time. In our proposed model, a single neural network is applied to the full image. It then divides the image into regions and predicts bounding boxes and probabilities for each region. These bounding boxes are weighted by the predicted probabilities. We have therefore proposed our own approach called N-YOLO. It divides the image into fixed sized images using correlation-based tracking algorithm, instead of resizing image step in YOLO algorithm. The amount of computation for object detection and tracking is significantly reduced. In addition, our proposed a system guarantees a high real-time performance.

Apart from achieving high accuracy, the proposed model runs in real-time. Also, the non-max suppression (which makes sure the object detection algorithm only detects each object once), it then outputs recognized objects together with the bounding boxes.

In the conventional YOLO, a single CNN simultaneously predicts multiple bounding boxes and class probabilities for those boxes. However, N-YOLO trains on full images and directly optimizes detection performance. This model has a number of benefits over other object detection methods.

Extremely fast: N-YOLO considers the whole image at a single instance during training and testing time and implicitly encodes contextual information about classes. N-YOLO then learns generalizable representations of objects so that when trained on natural images and tested on artwork, the algorithm outperforms other top detection methods.

The conventional approach processes 45 frames per second whereas our proposed algorithm processes (45 + 5) i.e. 50 frames per second which is incredibly fast. N ~ YOLO also understands generalized object representation. From our rigorous algorithms and comparative analysis, we have proved that this is one of the best algorithms for object detection so far. A comparatively similar performance to the R-CNN algorithms has also been shown to justify the same.

In addition to this, the conventional approach's performance has set COCO dataset as a benchmark dataset for object detection in terms of accuracy and training time. They have used object detection (Segmentation) format, stuff segmentation format and image captioning. However, in addition to these, we have applied panoptic segmentation along with keypoint detection format which are the other two significant format of COCO datasets. Our extensive results using all these formats have resulted our algorithm with highest *accuracy and training time.*

We used Anaconda - Python 3.6 (Windows 10), Conda Environment - YOLO.yml, Nvidia GPU (Graphics card), CUDA toolkit, CuDNN and PyTorch YOLO.

The rest of the paper is organized as follows. Section 2 surveys the existing work in the literature. Section 3 describes the purposed N-YOLO and N-YOLO tracker. Experiment results are described in Section 4. Section 5 concludes the paper.

2 Related work

Generally, object detection pipelines start extracting a set of robust features from input images. Then, classifiers or localizers are used to identify objects in the feature space. These classifiers or localizers are run either in the sliding window fashion or the entire image. Conversely, YOLO uses systems administration highlights from the whole image to anticipate each jumping box. It likewise predicts all jumping boxes over all classes for an image at the same time. This implies YOLO system reasons all-inclusive about the full image and every one of the articles in the image.

YOLO V3 is an upgraded version of YOLO that predicts an objectness score for each bounding box using logistic regression [17]. It adds feature pyramid for better detection of small objects. It changes to use a more complex backbone for feature extraction. To better detect small objects, YOLO V3 adds Feature Pyramid. Since YOLO V3 scans the images in one round, its time complexity is much smaller than heavier detectors such as R-CNN [4, 8].

However, YOLO imposes strong spatial constraints on bounding box predictions since each grid cell only predicts two boxes and can only have one class. Therefore, if there are several objects in the grid, there is a limit to the detection performance. Also, if we reduce the size to improve the performance of object detection, the number of bounding boxes for learning and reasoning increases exponentially, so scalability for performance improvement is restricted.

Recently, studies for multi object tracking have been combined with detection algorithm to achieve high performance. Danelljan et al. [3] proposed a *correlation-based object tracking algorithm* that performs by learning discriminative correlation filters based on a scale pyramid representation. This method learns separate filters for translation and scale estimation, which helps restricting the search area to smaller parts of the scale space and provides the freedom of selecting the feature representation for each filter independently. This method improves the performance compared to an exhaustive scale search. The scale estimation approach of this algorithm is generic and can be incorporated into any tracking method with no inherent scale estimation. Therefore, tracking performance depends on the bounding box of the object to be tracked in order to obtain excellent performance.

In order to migrate object detection and tracking services to edge network devices, decisions must be made immediately on-site. In addition, for architectural scalability, it is an indispensable approach to reduce the computational burden on the server or cloud, to perform operations on the edge device, and then to transmit the results to the cloud in terms of efficient use of the network. Regardless, video preparing shows to be a monster computational load for the edge gadget and the fundamental organization undertaking ought to be redistributed to the haze or even cloud focus focuses for execution. “Right” when the shadiness focus is responsible for major activity, close to consistent execution is polished.

Edge gadgets cannot manage the cost of the extra space for parameters and weight respects. Unquestionably, significantly after the dull and taking care of certifiable preparing is re-appropriated to the cloud or other incredible focus focuses, despite they need a tremendous volume of Smash. For instance, the most astounding remaining neural system (ResNet) has three structures with 51, 101, or 201 layers only, which proposes endless parameters to be stacked into the little gadget’s memory. Nikouei et al. [12] proposed a lightweight convolutional neural structure (L-CNN) lessens the computational expense of the CNN itself, missing amazingly any giving up the accuracy of the entire system. Moreover, the structure is unequivocal for human distinctive evidence to reduce vain giant numbers in each layer.

Yuan et al. [20] proposed a fusion method to enhance detection performance by utilizing the contextual information, and to improve the tracking performance and localization accuracy for traffic sign. Authors of [13–15] proposed several solutions to improve the tracking efficiency.

3 N-YOLO-tracker

3.1 YOLO-tracker

3.1.1 Bounding box

To perform object localization and classification in the city, YOLO divides the image into a grid of uniform size and extracts two candidate bounding boxes per grid. In the extracted candidate bounding box, the object can be recognized through the class identifier of the bounding box. The bounding box is then merged through non-maximum suppression (NMS).

To keep the size of the grid constant, YOLO forces the input image to a size that the inference engine can process. The change size can be arbitrarily set within a multiple of 32. The quality of the bounding box differs depending on the resizes the image. In addition, since the number of objects detected in the grid cannot exceed the number of candidates bounding boxes, there are objects that cannot be detected when there are many objects in the grid. Also, when the network model for object reasoning in each Grid becomes considerably small, it can be seen that the bounding box quality of object drops significantly.

In YOLO architecture, the aspect ratio is kept safe so that no information will miss. It is done according to the resolution. For example, by using YOLO, an image size of 1248 × 936 will be resized to 416 × 312 and then pad the extra space with black bars to fit into 416 × 416 network.

Figure 1 illustrates object performance according to resize size.

When resizing the same image to 640 × 320, 12 objects were detected. When resized to 320 × 160, 10 objects were detected. In addition, when the same image was inferred as YOLO V3 Tiny (learning and inference network minimized version), six objects were detected in 640 × 320 size and the object could not be detected in 320 × 160.

Also, for the same object, 640 ×320 showed higher intersection of union (IoU) than Ground Truth. As shown in fig. 1.

3.1.2 Tracking interval

In this paper, we propose a new approach based on video analysis. In video analysis, besides analyzing images, it is significant to explore association among image frames [2, 6, 21, 24]. Most recent studies have focused on tracking-by-detection [10]. Since the amount of object tracking is smaller than object detection, we combine object detection algorithm and object tracking algorithm to achieve real time object detection. Here tracking time interval is directly related with the estimation of distance of the object. These time intervals (small) are accumulated, and the interval with maximum sum is chosen as the one corresponding to the object in the bounding box. To increase the proposed throughput (*Th*) of the method, set the object detection period and track the object for video surveillance between intervals. When the tracking interval is N, the computation per second *Th* can be expressed as follows.

$$\mathrm{Th} = \mathit{DETECTION}(\mathrm{FPS}{-}\mathrm{N}) + \mathit{TRACKING}(N)$$

Where, *DETECTION*(N) is time to detect object of *N* frames and *TRACKING*(*N*) is time to track object of *N* frames. Therefore, the value of *Th* converges close to TRACKING(N) when the tracking interval is very high. However, since object tracking is not perfect and there may be objects newly found or disappearing in the middle of the frame, an appropriate level of interval should be set to improve detection performance.

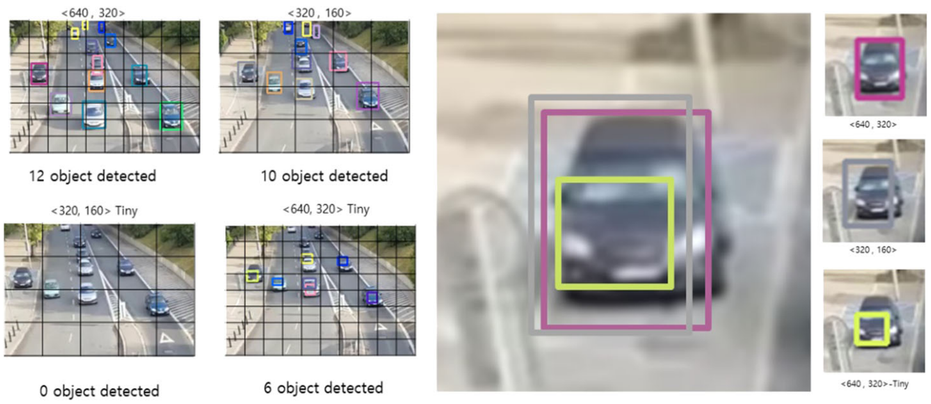


Fig. 1 YOLO performance depends on grid size

Figure 2 depicts the operation of the algorithm according to the tracking interval cycle. Here, the consecutive tracking stages refer to the position prediction. Initially we detect the object with given image step sizes. The object is provided with a unique ID that will help tracking it across the scene. There may be objects newly moving around. We create bound boxes for the same according to their respective position vectors.

3.2 N-YOLO

YOLO is one of the fastest architectures and has been applied in context with very limited computational resources [16]. In this paper, we proposed N-YOLO to improve detection performance in real video surveillance system. Because, YOLO detects an object by resizing the input image, resize size can be determined according to hardware performance and characteristics of environment to be detected. It is advantageous to reduce the resize size for real time object detection.

However, excessive resizing of target image can cause missing when detecting objects. Instead of resizing the image, the proposed method divides image into a certain size that can be deduced from YOLO and infer it sequentially. Figure 3 illustrates the behavior of N-YOLO. N-YOLO can solve the object detection missing of the image and not increase the amount of computation.

3.3 N-YOLO-tracker

Since N-YOLO detects only a part of the image after dividing the image into a certain size, accurate detection is difficult when the object moves between sub-images or exists between them. As shown in Fig. 4 (a), there exist objects that are not detected or not detected correctly compared with the actual image Fig. 4 (b).

The proposed algorithm solves the issue described above according to the inference time of the sub-image by driving the tracking algorithm that tracks the detected objects in each sub-image. Figure 5 describes the outline of N-YOLO-Tracker.

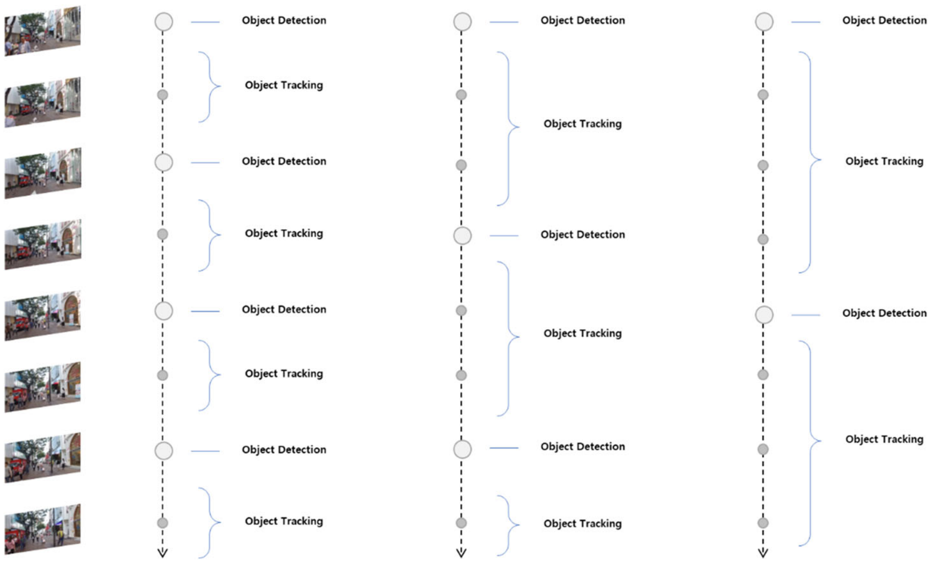


Fig. 2 YOLO-Tracker overview

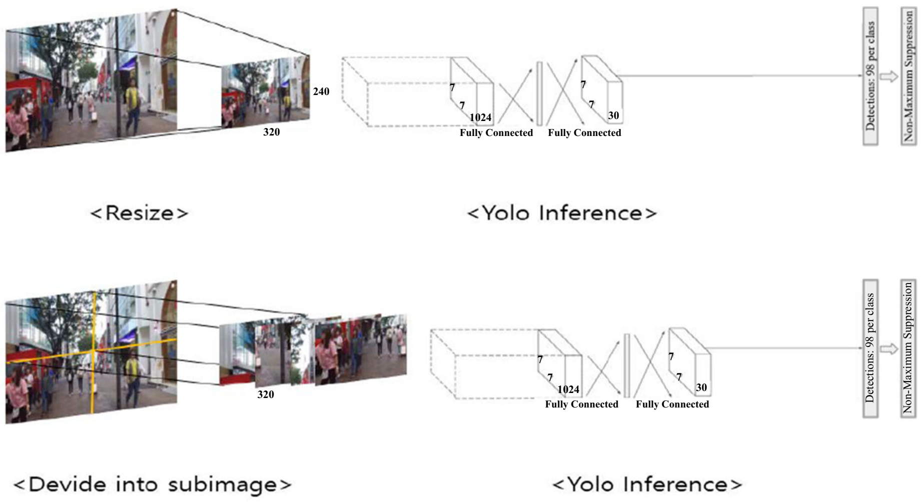


Fig. 3. The concept of N-YOLO method

Let us consider a 2 × 2 image as shown in fig. 4 (a) with two anchors per step with 4 different object classes. This means that we have a shape of 3 × 3 × 16 in y labels. Now, if we use 5 anchor boxes per step with the number of classes increased to 5; the target will be 3 × 3 × 16 × 5 = 3 × 3 × 80. We have applied this method as our training process so that we can map it with a 3 × 3 × 16 target. Secondly, any other new image can also be divided into the same number of grids which we have chosen during the training period. But for each grid, the model will predict an output of shape 3 × 3 × 16 (assuming this is the shape of the target during training time).

The Non-Max Suppression technique will be applied on the predicted boxes to obtain a single prediction per object. This brings to the end of the theoretical aspect of understanding how the YOLO algorithm works, starting from training the model and then generating prediction boxes for the objects. Below are the exact dimensions and steps that our proposed YOLO algorithm follows.

We take an input image of shape (608, 608, 3).

This image is passed to our neural network which returns a (19, 19, 5, 85) dimensional output.

The last two dimensions of the above output are flattened to get an output volume of (19, 19, 425):

Here, each cell of a 19 × 19 grid returns 425 numbers.

425 = 5 × 85, where 5 is the number of anchor boxes per grid.

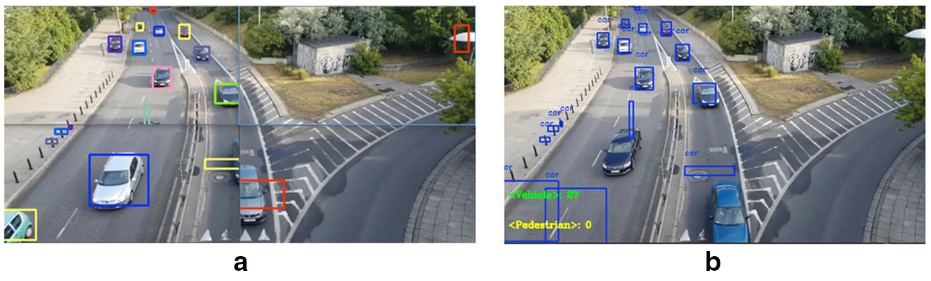


Fig. 4 Problem of object detection method with N-YOLO

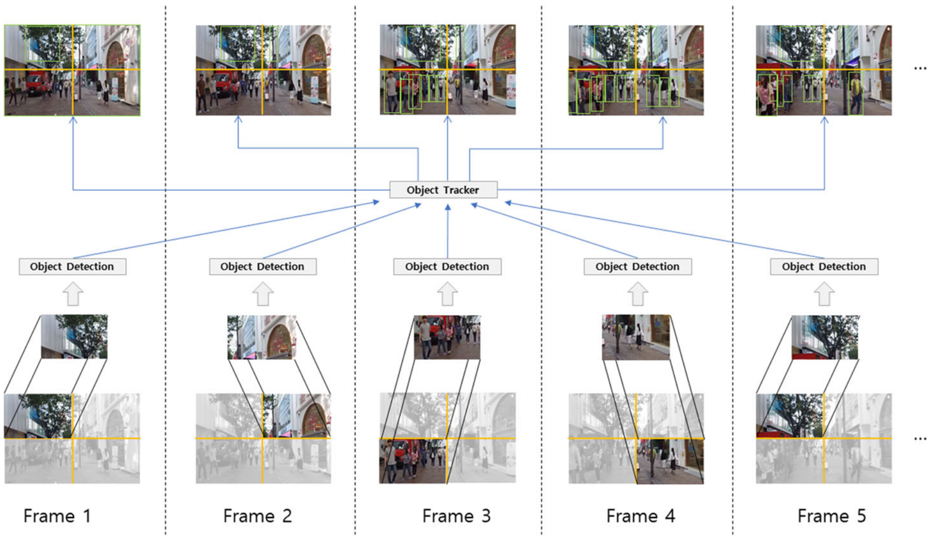


Fig. 5 N-YOLO-Tracker Overview

85 = 5 + 80, where 5 is (pc, bx, by, bh, bw) and 80 is the number of classes we want to detect.
Finally, we do the IoU and Non-Max Suppression to avoid selecting overlapping boxes.

The object tracker manages the detected objects after performing object detection on sub-image of frame 1 to 1 quadrant, and object tracker further manages the object tracking for each sub-image in the frame afterwards. The object tracker updates and tracks the list of detected objects by performing object detection again on the sub-image detected in frame 1 to frame 5. Hence, the accuracy of object detection and tracking according to the parallax between sub-images can be increased, and the increase in the computation amount can be suppressed.

We have considered an image with two anchors per step with 4 different object classes. We have applied this method as our training process so that we can map it with the target image. This is why we have taken 5 frames to achieve more accuracy and prevision. Also, any new image can also be divided into the same number of grids which we have chosen during the training period. But for each grid, the model will predict a different output ranges because it is assumed that this is the shape of the target during training time. We have applied Non-Max Suppression technique on the predicted boxes to obtain a single prediction per object.

In conclusion, in contrast to pipelining object detection and tracking when analyzing video, this method plays a role of detected object manager which merges detection result of sub-image according to time difference. Therefore, when analyzing a video with a larger size resolution, the existing method has a complexity of $O(N^2)$ when the size of the image is increased to N when the size of the image is doubled besides the complexity of O (N) of our proposed method.

4 Experiment

In this section, we demonstrate how N-YOLO-Tracker helps object detection and tracking system improve the performance at a certain image size. We compare the YOLO V3 as a fastest object detection algorithm and N-YOLO-Tracker with several tracking interval environments.

4.1 Setup and evaluation

We developed the object detection and tracking system followed the framework described in section 3. We choose YOLO V3 which trained with the Microsoft Common Objects in COntext (MS COCO) dataset [11]. And result of YOLO V3 is assumed to be Oracle and the performance of YOLO-Tracker and N-YOLO-Tracker is evaluated through Accuracy, Precision, Recall & F1 Score. Four key variables are used to calculate the above items. Each variable is shown below.

In Fig. 6, green cells are the true positive (TP) and true negatives (TN) observations that are correctly predicted. The cells in red are the false positives (FP) and false negatives (FN) observations that We want to minimize.

We define accuracy, precision, recall and F1 score and evaluate these performance measures as follows.

(a) **Accuracy** – Apart from the most intuitive performance measure, accuracy is also a ratio of correctly predicted observation to the total observations. Accuracy can be obtained as follows.

$$Accuracy = TP + \frac{TN}{TP} + FP + FN + TN$$

(b) **Precision** – In our model, precision helps when the costs of FP are high. It can be obtained as follows.

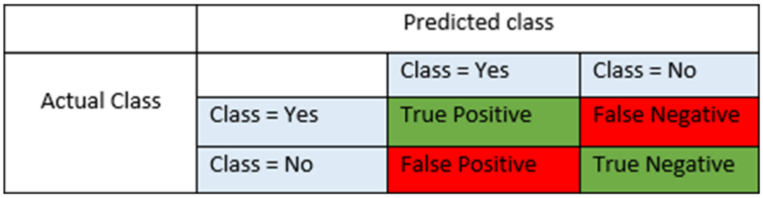
$$Precision = \frac{TP}{TP} + FP$$

(c) **Recall** – In our model, we have used 'recall' when the cost of FN is high using the following relationship.

$$Recall = \frac{TP}{TP} + FN$$

(d) **F1 Score** – F1 Score is the harmonic mean of Precision and Recall, therefore it is a class balanced accuracy measure. This score considers both false positives and FN to strike a balance between Precision and Recall. The higher the F1 score means higher precision and recall. F1 score ranges from 0 to 1 as the best (perfect precision and recall) and worst. F1 score is different in many ways than Accuracy. In fact, Accuracy works best if false positives and false negatives have similar cost. It is better to look at both Precision and Recall, in case the cost of FP and FN are very different.

$$F1\ Score = \frac{2 \times (Recall * Precision)}{(Recall + Precision)}$$



	Predicted class	
Actual Class	Class = Yes	Class = No
Class = Yes	True Positive	False Negative
Class = No	False Positive	True Negative

Fig. 6 Description of TP, FN, FP and TN

4.2 Speed analysis

The major computational cost of the proposed method is object detection tracking for image of video stream and object tracking. When using 5 s video clip of road traffic, the speed of the proposed method is 5 frames per second with Python and without GPU environment at 0 tracking interval. We compare the speed of our method with original *YOLO V3* and *YOLO V3 Tiny* algorithm with same trained model and environment.

As shown in Fig. 7, N-YOLO showed higher throughput performance in both YOLO V3 and Tiny. In the 320 size, the average performance is 55% higher. In the case of the 640-size image, the performance is 103%, and in the case of 960 size image, the performance is 315%.

4.3 Qualitative comparison

From the various experimental results, we have plotted the results of Accuracy, precision, recall, F1 score and IoU for qualitative comparison. The accuracy performance decreases as the tracking interval becomes longer because each object has its own ID. And, if the same object reenters into the frame, the unique ID is again assigned to the same object. The tracking interval, if not maintained at reasonable time slot will cause the ambiguity in object identification. Throughput performance improves, but it converges to a specific value when the interval becomes longer. Therefore, it is advisable to set the appropriate tracking interval according to the characteristics of the input video and the environmental requirements, considering the trade-off between accuracy and throughput.

As shown in Fig. 8 (a), the minimum tracking interval to achieve 6 FPS is 36 frames, with an F1 score of 0.66 and an average IoU value of the experimental results and Ground Truth is 0.62. On the other hand, as in Fig. 8 (b), in N-YOLO V3, the tracking interval to achieve 6 FPS is 5 frames, the F1 score is 0.73, and the average IoU value with the ground truth is 0.75. Thus, the N-YOLO V3 -Tracker has about 10.6% better F1 score and 20.9% better average IoU than YOLO V3 –Tracker (Table 1).

4.4 Loss function

In our work, loss functions provide a static representation of how our model is performing. Our proposed algorithm use loss function for optimization; thus, finding the best parameters (weights) for our data. YOLO optimizes a multi-part loss function that is composed of four parts during training the model. They are.

a. regression loss,
b. confidence loss,
c. classification loss, and
d. the loss responsible for not having any object.

Above mentioned loss functions occur for multi-class object detection. The proportion of the four parts sub-loss is 1:1:1:1; and the total loss function is calculated as below:

Equation ID=t=t b 5pt

$$L_t = \Sigma L_{\mathbf{r}} + \Sigma L_c + \Sigma L_{cl} + \Sigma L_{no}$$

where.

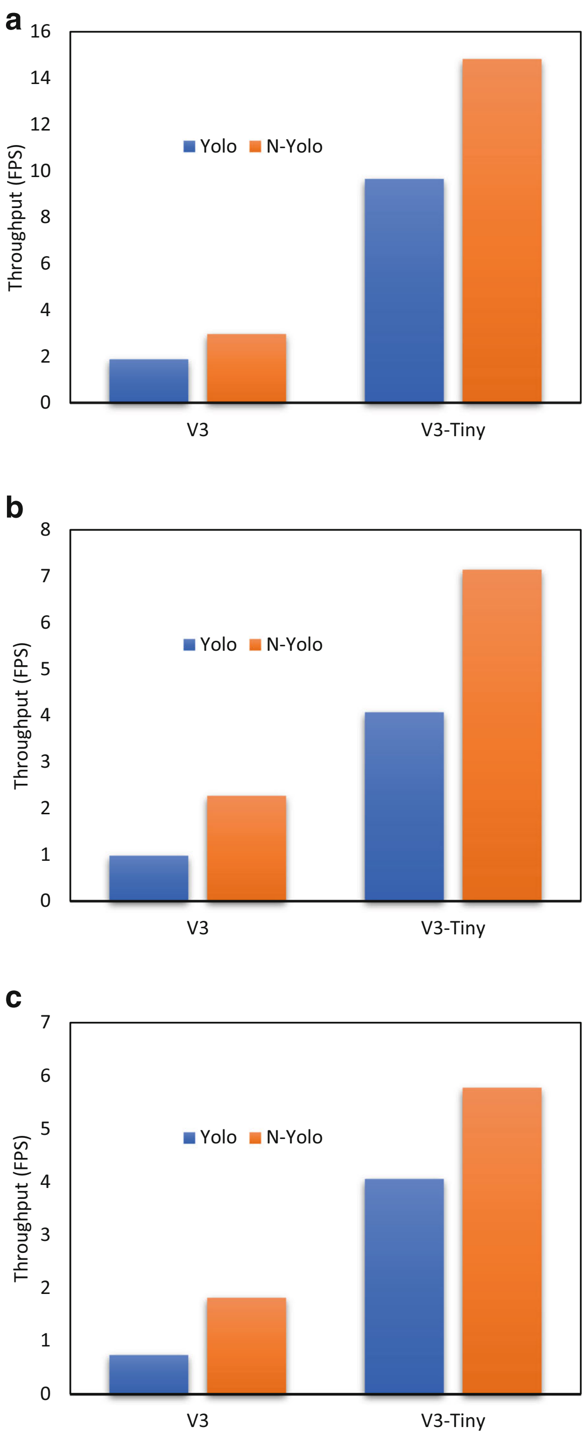
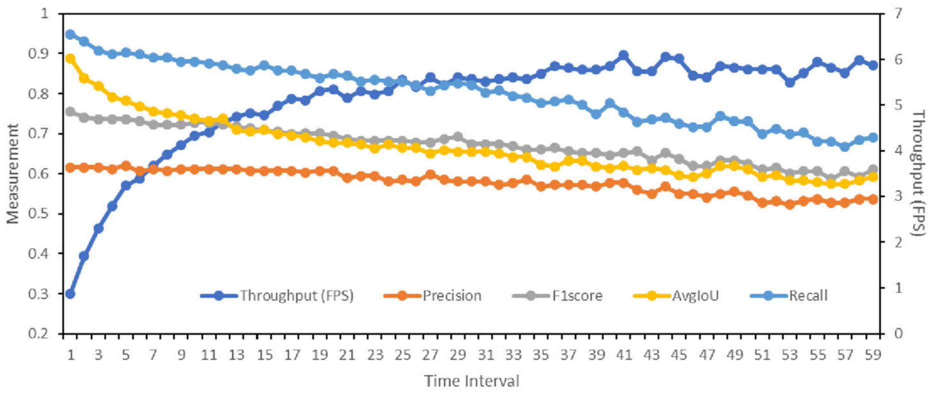
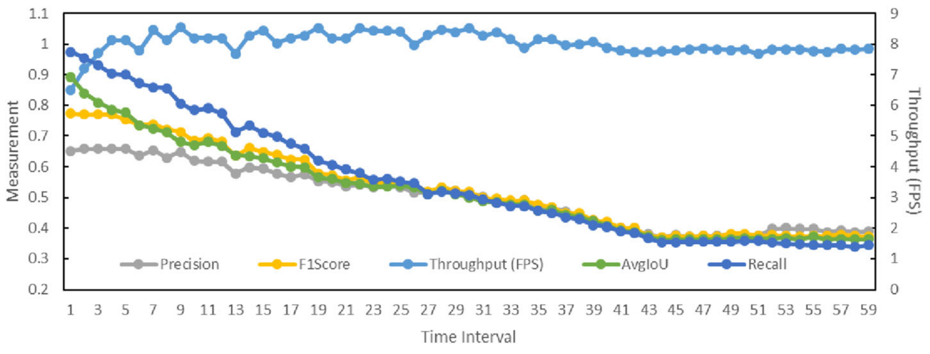


Fig. 7 Comparison of YOLO and N-*YOLO* based on V3 and V3-*T*iny. (a) Experiment results of the video with 320 ×320 resolution, (b) Experiment results of the video with 640 ×320 resolution, and (c) Experiment results of the video with 960 ×320 resolution



(a) YOLO V3



(b) N-YOLO V3

Fig. 8 Graph of YOLO and N-YOLO accuracy according to tracking intervals

L_r	is the loss of coordinate regression,
L_c	is the loss of confidence with objects,
L_{cl}	is the loss of classification, and.
L_{no}	is the loss of confidence with no objects.

However, face detection is a problem of binary classification. To make the total loss function more suitable for face detection, we empirically revise the weights to 2:1:0.5:0.5. The final loss function is as follows.

$$L_t = 2 \times \Sigma L_{\mathbf{r}} + \Sigma L_c + 0.5 \times \Sigma L_{cl} + 0.5 \times \Sigma L_{no}$$

Since there are multiple object predictions per grid, YOLO loss function now performs a max-IoU. It is can be observed from fig. 5 that the bounding box coordinate loss is still a weight linear regression loss.

4.5 Detection and tracking

=In our intuitive algorithm, the currently tracked objects are framed. These the future frames match with the current frames, we update it else the model keeps on training and testing those images.

Table 1 Comparative analysis of our proposed model with the other

Previously proposed Algorithms	**Features**	**Results**
YOLO V3 [6]	- Uses dimension clusters for predicting bounding boxes - Uses independent logistic classifiers - Performs on Open Images Dataset and COCO datasets which have only 9 clusters	FPS = 5 frames, F1 score = 0.73, and average IoU value = 0.75.
Proposed model	*- Uses sub images* *- divides into fixed size images and merges detection results of each divided sub-image with inference results* *- Dataset used is Microsoft Common Objects in COntext (MS COCO) dataset* [21].	*FPS = 36 frames* *F1 score = 0.66* *Average IoU value = 0.62* i.e. *our model has 10.6% better F1 score and 20.9% better average IoU than YOLO V3 -Tracker.*
Al-masnia et.al. [22]	- Deep learning technique was proposed - Used only for breast cancer / breast abnormalities.	- detects the mass location - also distinguishes between benign and malignant lesions - No F1 score
Proposed model	*- More specifically region-based convolutional network us used because it has two stage for inferencing* *- Can be used in the images of multiple domains*	*- can be used for real-time video surveillance in edge computing environments* i.e. *it is applicable for almost all environments.* *- F1 score of 0.66* *- F1 score is calculated using parameters like precision, recall, where an F1 score reaches its best value at 1*
Wang et al. [23]	- the images of training insulator images with the TensorFlow platform were used - Calculates speed and accuracy - Concluded that the accuracy of identification increases with the increase in the number of training insulators	- accuracy of 83.5% achieved only for identification of insulators - no image step size is used
Proposed model	- Images we have used are platform independent - Precision and recall are the major parameters along with accuracy and F1 score	- Our F1 score is applicable wide categories of images - Image step size is used

```
currentlyTrackedObjects = []
For each frame:
 // 1. Try to match the currently tracked object with the detections of this frame
 For each detection:
  doesMatchExistingTrackedObject(detection, currentlyTrackedObjects)
  // if it matches, update the trackedObject position
  matchedTrackedObject.update(detection)
 // 2. Assign unmatched detections to new objects
 currentlyTrackedObjects.add(new trackedObject(detection))
 // 3. Clean up unmatched tracked objects
 For each currentlyTrackedObjects:
  if isUnmatched(trackedObject):
   trackedObject.remove()
```

Here, the challenge is to compare the two detections, how to determine if they are tracking the same object, as mentioned in [5]? Therefore, we have used a distance() function, which compares two detections positions, i.e. current detections and candidate for next frame, and determine their relative distance. If they are considered close enough, we can match them. We used the center of the bbox to compute this distance.

The distance() function is illustrated as follows.

```
function distance(item1, item2) {
  // compute euclidian distance between centers
  euclidianDistance = computeEuclidianDistance(item1, item2)
  if (euclidianDistance > DISTANCE_LIMIT):
    // Do not match
  else:
    // Potential match
}
```

Our results show that the matching is almost 80% of the detections, but the problems like losing the track of the object and assigning it a new ID (even if it is the same object) do exist. We have improved these issues by keeping a memory of the unmatched item for a few frames and avoid removing them directly because sometimes the detection algorithms miss the object for a few frames.

- By predicting the position on the new frame with a velocity vector
- By improving this distance function
- Keep unmatched object in memory

We first integrated the idea of keeping in memory the unmatched items for a few frames which is simply wait a few frames before removing it from the tracked items.

```
  if isUnmatched(trackedObject):
     trackedObject.unmatchedThisFrame()
And the unmatchedThisFrame() function looks like this

function unmatchedThisFrame() {
  nbUnmatchedFrame++
  if nbUnmatchedFrames > 5:
    //Effectively delete the item
    this.remove()
}
```

This makes the tracker more resilient to missing detections from YOLO and avoided some reassignments, but in order to have it more effective, we needed also to predict the next position.

5 Conclusion

In this study, we developed a real-time object detection and tracker based on the N-YOLO-Tracker with YOLO V3 and correlation-based tracker. The proposed method uses image segmentation and image merging with tracking algorithm method for real-time object detection and tracking. Extensive experiments have been done to verify the reliability of the proposed method. This method has scalability for real-time video surveillance in edge computing environments with limited computing power. The experiment results show that the quality of bounding box is important for efficient operation of correlation-based object tracking and combining with YOLO V3 produces the most efficient bounding box among the compared object detection algorithms. In addition, the object tracker is used as a merging manager of detected objects to improve linear scalability.

The limitation of this model is that it does not handle re-entering, because YOLO does not have reentered detection for frames. Secondly, when the tracker gets the object back, it gets a new ID. In our future work, we will include the real time object detection using binary cross entropy loss for the class predictions and classification loss. It is because, from our study and the most recent works, we observe that YOLO uses binary cross entropy. Since, the loss for when there is no object assigned to the grid cell is the same. Additionally, we will include more formal analyses to justify whether the proposed method meets the real-time constraints while improving the accuracy. To verify the preference of the proposed method, we will carry out experiments over big data.

References

1. Al-masni MA, Al-antari MA, Park J-M et al (2018) Simultaneous detection and classification of breast masses in digital mammograms via a deep learning YOLO-based CAD system. Comput Methods Prog Biomed 157:85–94. https://doi.org/10.1016/j.cmpb.2018.01.017
2. Cao X, Guo S, Lin J, Zhang W, Liao M (2020) Online tracking of ants based on deep association metrics: method, dataset and evaluation. Pattern Recogn 103:107233. https://doi.org/10.1016/j.patcog.2020.107233
3. Danelljan M, Häger G, Khan FS, Felsberg M (2014) Accurate scale estimation for robust visual tracking. In: BMVC 2014 - proceedings of the British machine vision conference 2014. British Machine Vision Association, BMVA, pp. 1–11
4. Ding Z, Wong E (2019) Confidence trigger detection: an approach to build real-time tracking-by-detection system. ArXiv 1902(00615):1–9
5. Durand T Tracking things in object detection videos. In: Move Lab https://www.move-lab.com/blog/tracking-things-in-object-detection-videos. Accessed 27 Jun 2020
6. Fu H, Wu L, JianM YY, Wang X (2019) MF-SORT: simple online and Realtime tracking with motion features. In: Zhao Y, Barnes N, Chen B et al (eds) Image and graphics. Springer International Publishing, Cham, pp 157–168
7. Geng Y, Liang R-Z, Li W et al (2017) Learning convolutional neural network to maximize Pos@top performance measure. Bruges, Belgium, pp 589–594
8. Girshick R (2015) Fast R-CNN. In: The IEEE international conference on computer vision (ICCV). IEEE Computer Society, Boston, pp. 1440–1448
9. Huang C-H, Boyer E, Do B et al (2015) Toward user-specific tracking by detection of human shapes in multi-cameras. In: The IEEE conference on computer vision and pattern recognition (CVPR). IEEE Computer Society, Boston, pp 4027–4035
10. Le N, Heili A, Odobez JM (2016) Long-term time-sensitive costs for CRF-based tracking by detection. In: lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics). Springer Verlag, pp 43–51
11. Lin TY, Maire M, Belongie S et al (2014) Microsoft COCO: common objects in context. In: lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics). Springer Verlag, pp 740–755
12. Nikouei SY, Chen Y, Song S, et al (2018) Real-time human detection as an edge service enabled by a lightweight CNN. In: proceedings - 2018 IEEE international conference on EDGE computing, EDGE 2018 - part of the 2018 IEEE world congress on services. Institute of Electrical and Electronics Engineers Inc., pp 125–129

13. Pinho RR, Tavares JMR, Correia MV (2007) An improved management model for tracking missing features in computer vision long image sequences. WSEAS Trans Inf Sci Appl 1:196–203
14. Pinho RR, Tavares JMRS (2009) Tracking features in image sequences with Kalman filtering, global optimization, mahalanobis distance and a management model. Computer Modeling in Engineering & Sciences 46:51–75
15. Pinho RR, Tavares JMR (2005) Correia MV (2005) a movement tracking management model with Kalman filtering, global optimization techniques and mahalanobis distance. Advances in Computational Methods in Sciences and Engineering 1:1–4
16. Redmon J, Farhadi A (2017) YOLO9000: better, faster, stronger. In: The IEEE conference on computer vision and pattern recognition (CVPR). IEEE Computer Society, Hawaii, pp 7263–7271
17. Redmon J, Divvala S, Girshick R, Farhadi A (2016) You only look once: unified, real-time object detection. In: The IEEE conference on computer vision and pattern recognition (CVPR). IEEE Computer Society, Las Vegas, pp. 779–788
18. Tavares J, Padilha A (1995)Matching lines in image sequences with geometric constraints. Portugal, pp 1–3
19. Wang S, Niu L, Li N (2018) Research on image recognition of insulators based on YOLO algorithm. In: 2018 international conference on power system technology (POWERCON). Pp 3871–3874
20. Yuan Y, Xiong Z, Wang Q (2017) An incremental framework for video-based traffic sign detection, tracking, and recognition. IEEE Trans Intell Transp Syst 18:1918–1929. https://doi.org/10.1109/TITS.2016.2614548
21. Zhang G, Liang G, Su F, Qu F, Wang JY (2018) Cross-domain attribute representation based on convolutional neural network. In: Huang D-S, GromihaMM, Han K, Hussain A (eds) Intelligent computing methodologies. Springer International Publishing, Cham, pp 134–142
22. Zhang B, Huang Z, Rahi BH, Wang Q, Li M (2019) Online semi-supervised multi-person tracking with gaussian process regression. MATEC Web Conf 277:01003. https://doi.org/10.1051/matecconf/201927701003
23. Zhang G, Liang G, LiW FJ, Wang J, Geng Y, Wang JY (2017) Learning convolutional ranking-score function by query preference regularization. In: Yin H, Gao Y, Chen S et al (eds) Intelligent data engineering and automated learning – IDEAL 2017. Springer International Publishing, Cham, pp 1–8
24. Zhu J, Zhang S, Yang J (2019) Online multi-object tracking using single object tracker and Markov clustering. In: Zhao Y, Barnes N, Chen B et al (eds) Image and graphics. Springer International Publishing, Cham, pp 555–567