International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Detection of Traffic Control Devices during Bridge Construction Projects Using Deep Learning Application

Mrs. M. Parvathi

Head of the Department, Computer Science, Latha Mathavan Arts and Science College, Madurai, Tamilnadu

Abstract

Recent days during the time of road construction so many accidents may occur due to the objects in the construction area cannot capture during travel. Some method must be implemented to classify and locate instances in the image. In deep learning area it extends its application to this field for object detection. Even though a method is identified it has risk factors such as real-time detection, changeable weather, and complex lighting conditions. In this paper we are discussing the algorithm used for detection and a next version of that algorithm to detect the object with more precision. And also the number frames capture during driving is also more than other algorithm. Here we discussed about YOLO algorithm, and which is the best algorithm that applies different aspects such as key generic object detection frameworks, categorized object detection applications in traffic scenario, evaluation metrics, and classified datasets are included. The idea in YOLO v7 is to avoid that there is an identity connection when a convolutional layer with residual or concatenation is replaced by re-parameterized convolution.

Keyword: YOLO, evaluation metrics, deep learning

Existing Method

The existing methodology is consider three different types of improved deep-learning based object detection algorithms, i.e. Multiple stage detectors like R-CNN, Faster R-CNN (Faster Region-Based CNN) and Single Stage Detectors like Yolo (You Only Look Once), Single Shot Detector (SSD).

Proposed method

YOLO v7 is a real time object detection system that falls under the category of single shot detectors. It processes the entire image in one forward pass, making it faster compared to some other methods. It has a more efficient and streamlined architecture compared to its predecessors. It utilizes a series of convolutional laers to predict bounding boxes and class probabilities. It aims to strike a balance between speed and accuracy, making it suitable for various applications where real-time detection is crucial. It is primarily designed for object detection, and it excels in detecting objects in images or videos with a singly pass.It is used to detect the object with more precision and faster. And also it can capture number frames in a fastest manner.

Object Detection Methods

Generally, object detection methods can be classified as either neural network-based or non-neural approaches. Also, some of them are rule-based, where the rule is predefined to match specific objects. Non-neural approaches require defining features using some feature engineering techniques and then using a method such as a support vector machine (SVM) to do the classification.

Some of the non-neural methods are:

- Viola-Jones object detection method based on Haar features
- Scale-invariant feature transform (SIFT)
- Histogram of Oriented Gradients (HOG)
- Other methods based on template, shape, or color matching

On the other hand, neural network techniques can do end-to-end object detection without explicitly defining features. They are far more accurate than non-neural based and are typically built on convolutional neural networks (CNN).

Some of the neural network methods are:

- Region-Based Convolutional Neural Networks (R-CNN, Fast R-CNN, etc.)
- Single Shot Detector (SSD)
- Retina-Net
- You Only Look Once (YOLO)

Challenges in Object Detection

In object detection, the bounding boxes are always rectangular. As a result, if the object contains the curvature part, it does not help determine its shape. In order to find precisely the shape of the object, we should use some of the image segmentation techniques.

Some non-neural methods may not detect objects with high accuracy or may produce a large number of false-positive detections. Although neural network methods are more accurate, there are some drawbacks. For example, they require a large amount of annotated data for training. Training is often expensive in time and space and, as a result, prolonged on standard computers.

Object detection is the task of detecting instances of objects of a specific class within an image or video. Basically, it locates the existence of objects in an image using a bounding box and assigns the types or classes of the objects found. For instance, it takes an image as input and generates one or more bounding boxes, each with the class label attached. These algorithms are powerful enough to handle multi-class classification and localization and objects with multiple occurrences.

Object detection is a combination of two tasks:

- Image classification
- Object localization

Image classification algorithms predict the type or class of an object in an image among a predefined set of classes that the algorithm was trained for. Usually, input is an image with a single object, such as a cat. Output is a class or label representing a particular object, often with a probability of that prediction. Object localization algorithms locate the presence of an object in the image and represent its location with a bounding box. They take an image with one or more objects as input and output the location of

one or more bounding boxes using their position, height, and width.

In order to solve these challenges, we can use the YOLO algorithm. Thanks to the transfer learning capabilities, we would be able to use already pre-trained models or spend some time fine-tuning models with our data. Furthermore, the YOLO algorithm is one of the most popular methods for performing object detection in real-time because it achieves high accuracy on most real-time processing tasks while maintaining a reasonable speed and frames per second, even on devices accessible to almost everyone.

You Only Look Once (YOLO)

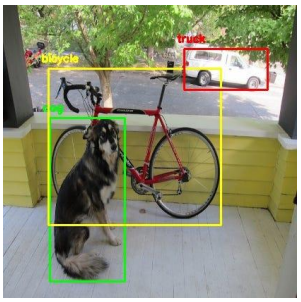
You Only Look Once (YOLO) is one of the most popular model architectures and object detection algorithms. It uses one of the best neural network architectures to produce high accuracy and overall processing speed, which is the main reason for its popularity. If we search Google for object detection algorithms, the first result will be related to the YOLO model. YOLO algorithm aims to predict a class of an object and the bounding box that defines the object location on the input image. It recognizes each bounding box using four numbers:

- Center of the bounding box ( )
- Width of the box ( )
- Height of the box ( )

In addition to that, YOLO predicts the corresponding number for the predicted class as well as the probability of the prediction

Working principle of YOLO

With the existence of a grid, it's possible to detect one object per grid cell instead of one object per image. For each grid cell, we can encode a vector that will describe the cell. For instance, the first cell from the top-left doesn't have any object, and we describe it as:



(1) where is the probability of the object class, and are coordinates of the center of the bounding box, relative to the cell, and are width and height of the bounding box relative to the whole image, and are 0 or 1 depending on which class represents the bounding box (for cat and for dog). Vector consists of symbols because if the first component is equal to zero, then the rest of the components can have random numbers are they are not taken into consideration. Next, if we take the cell that contains the center of the blue bounding box with the cat, we'll have a vector

(2) Following this procedure, if we define one vector for each grid cell, the whole image is represented with nine vectors with size 7 or 3x3x7 tensor. This means that in our data set, each image sample is labeled with one 3x3x7 tensor. Using that data set, we are able to create a training and test set and train

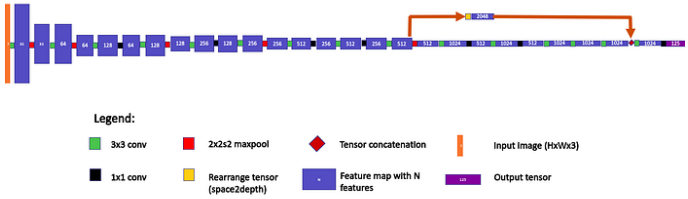
the convolutional network, which is exactly how YOLO works. Using CNN, YOLO is able to predict all objects in one forward pass and that is the reason for its full name "You Only Look Once".

YOLO architecture

Yolo-V1 Architecture

Yolo-V1 was the first appearance of the 1-stage detector concept. The architecture employed ***batch normalization*** (***BN***) and ***leaky ReLU activations***, that were relatively new at the time. I'm not going to elaborate on V1 since it's pretty outdated and lacks some of the strong features that were introduced later.

Yolo-V2 Architecture



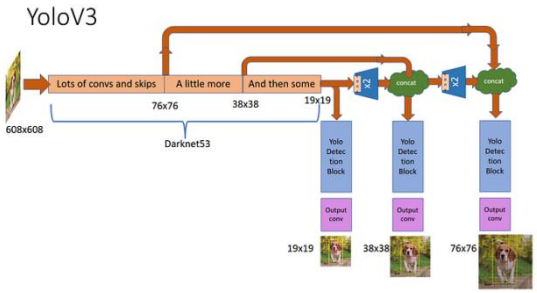
Yolo-V2 Contains 22 convolutions and 5 maxpool operations. Feature map height represents spatial resolution. The 125-feature output is for VOC PASCAL dataset with 20 classes and 5 anchors.

In version Yolo-V2 the authors, among other changes, removed the fully-connected layer at the end. This enabled the architecture to be truly resolution-independent (i.e. — the network parameters can fit any input resolution). This doesn't necessarily mean that the network will perform well on any resolution. A resolution augmentation routine was employed for that during training. Redmon created multiple flavors of Yolo-V2, including smaller, faster (and less accurate) versions, like ***Tiny-Yolo-V2*** etc.

Tiny-Yolo-V2 has an extremely simple architecture since it doesn't have the strange bypass and rearrange operation that likes its older sibling. The tiny version is just a nice, long chain of convolutions and max pools.

YOLO-V3 Architecture

Inspired by ***ResNet*** and ***FPN*** (Feature-Pyramid Network) architectures, ***YOLO-V3 feature extractor***, called ***Darknet-53*** (it has 52 convolutions) contains skip connections (like ResNet) and 3 prediction heads (like FPN) — each processing the image at a different spatial compression.



YOLO-V3 architecture.

Like its predecessor, Yolo-V3 boasts good performance over a wide range of input resolutions. In ***GluonCV's model zoo*** you can find several checkpoints: each for a different input resolutions, but in fact the network parameters stored in those checkpoints are identical. Tested with input resolution 608x608 on COCO-2017 validation set, Yolo-V3 scored 37 mAP (mean Average Precision). This score is

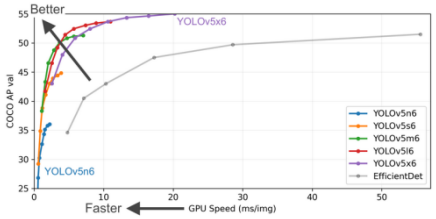
identical to GluonCV's trained version of ***Faster-RCNN*-ResNet50**, (a faster-RCNN architecture that uses ResNet-50 as its backbone) but 17 times faster. In that model zoo the only detectors fast enough to compete with Yolo-V3 (Mobilenet-SSD architectures) scored mAP of 30 and below.

YOLOv5 **Architecture**

Up to the day of writing this article, there is no research paper that was published for YOLO v5 as mentioned here, hence the illustrations used bellow are unofficial and serve only for explanation purposes.

It is also good to mention that YOLOv5 was released with five different sizes:

- **n** for extra small (nano) size model.
- **s** for small size model.
- **m** for medium size model.
- **l** for large size model
- **x** for extra large size model

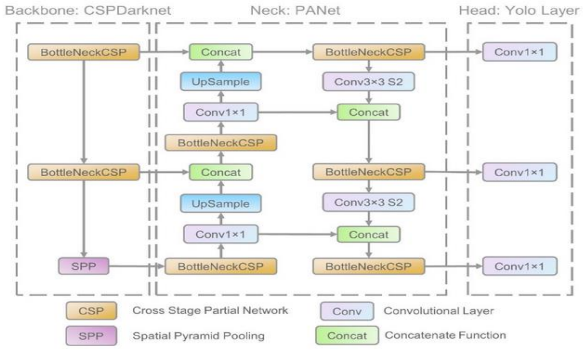


Performance of YOLOv5 different sizes models

There is no difference between the five models in terms of operations used except for the number of layers and
parameters as shown in the table below.

Model	size (pixels)	mAP^{val} 0.5:0.95	mAP^{val} 0.5	Speed CPU b1 (ms)	Speed V100 b1 (ms)	Speed V100 b32 (ms)	params (M)	FLOPs @640 (B)
YOLOv5n	640	28.0	45.7	**45**	**6.3**	**0.6**	**1.9**	**4.5**
YOLOv5s	640	37.4	56.8	98	6.4	0.9	7.2	16.5
YOLOv5m	640	45.4	64.1	224	8.2	1.7	21.2	49.0
YOLOv5l	640	49.0	67.3	430	10.1	2.7	46.5	109.1
YOLOv5x	640	50.7	68.9	766	12.1	4.8	86.7	205.7
YOLOv5n6	1280	36.0	54.4	153	8.1	2.1	3.2	4.6
YOLOv5s6	1280	44.8	63.7	385	8.2	3.6	12.6	16.8
YOLOv5m6	1280	51.3	69.3	887	11.1	6.8	35.7	50.0
YOLOv5l6	1280	53.7	71.3	1784	15.8	10.5	76.8	111.4
YOLOv5x6 + TTA	1280 1536	55.0 **55.8**	72.7 **72.7**	3136 -	26.2 -	19.4 -	140.7 -	209.8 -

All the YOLOv5 models are composed of the same 3 components: **CSP-Darknet53** as a backbone, **SPP** and **PANet** in the model neck and the **head** used in YOLOv4.



Network architecture for YOLO v5

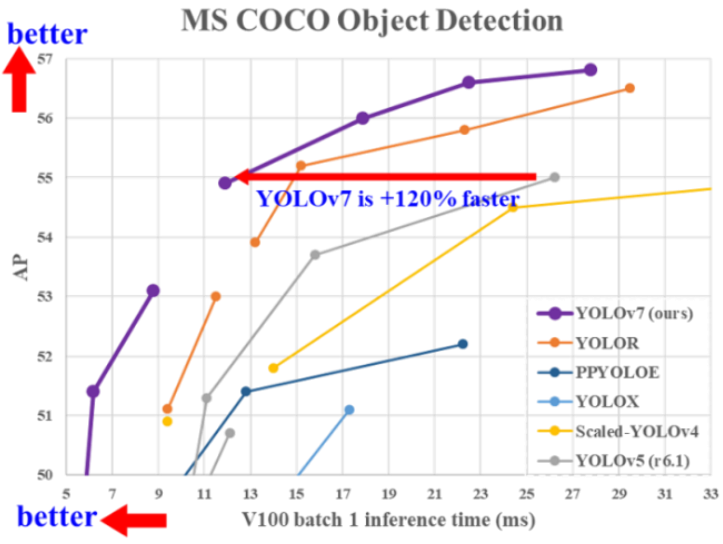
Algorithm Overview:

The architecture consists of various parts, broadly they are - The input which comes first and it is basically what we've as our set of training images which will be fed to the network - they are processed in batches in parallel by the GPU. Next are the Backbone and the Neck which do the feature extraction and aggregation. The Detection Neck and Detection Head together can be called as the Object Detector.
And finally, the head does the detection/prediction. Mainly, the Head is responsible for the detection (both localization and classification).Because YOLO is a one-stage detector it does both of them simultaneously (also known as Dense Detection). Whereas, a two-stage detector does them separately and aggregates the results (Sparse Detection)

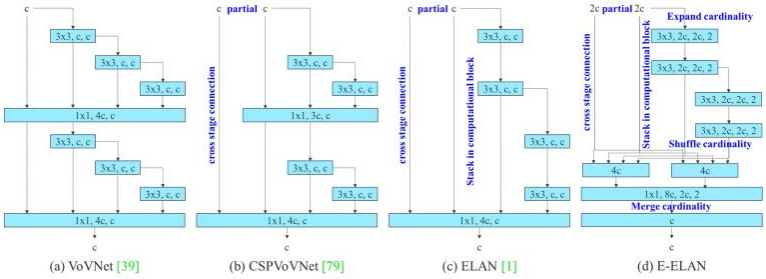
YOLO v7 method

The YOLO (You Only Look Once) v7 model is the latest in the family of YOLO models. YOLO models are single stage object detectors. In a YOLO model, image frames are featurized through a backbone. These features are combined and mixed in the neck, and then they are passed along to the head of the network YOLO predicts the locations and classes of objects around which bounding boxes should be drawn. YOLO conducts a post-processing via non-maximum supression (NMS) to arrive at its final prediction.
There are six versions of the model ranging from the namesake YOLOv7 (fastest, smallest, and least accurate) to the beefy YOLOv7-E6E (slowest, largest, and most accurate).
The differences between the different sizes of the model are:

- The image input resolution
- The number of **anchors**
- The number of parameters
- The number of layers

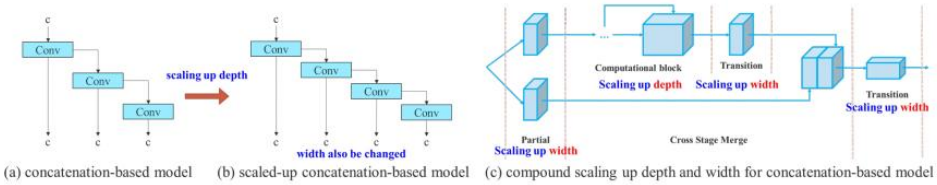


The evaluation of YOLOv7 models show that they infer faster (x-axis) and with greater accuracy (y-axis) than comparable real time object detection models. YOLOv7 evaluates in the upper left - faster and more accurate than its peer networks.



Scaling Techniques

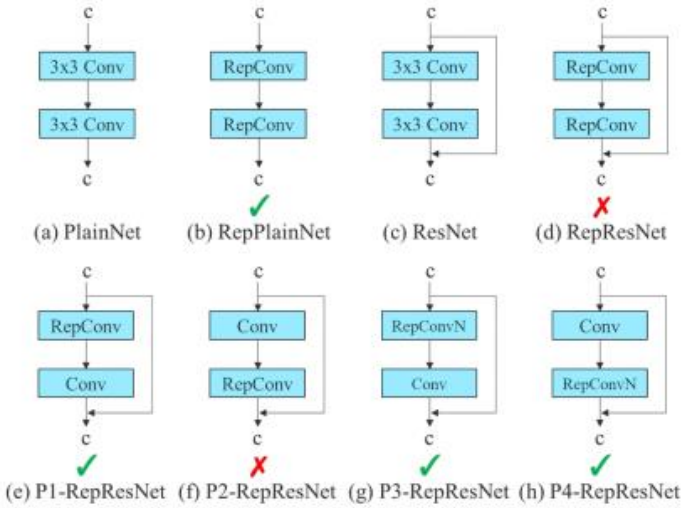
In YOLOv7 the authors scale the network depth and width in concert while concatenating layers together.



Compound scaling in YOLOv7 model sizes

Re-parameterization

Re-parameterization techniques involve averaging a set of model weights to create a model that is more robust to general patterns that it is trying to model.



Conclusions

This paper presents a significant challenge for object detection networks due to its ultra-low pixel objects with less than 20 pixels, long distance, large background, and dense population. To address these challenges, this paper improved upon the most advanced object detector, YOLOv7. The new YOLOv7 shows the best speed-to-accuracy balance compared to state-of-the-art object detectors. In general,

YOLOv7 surpasses all previous object detectors in terms of both speed and accuracy, ranging from 5 FPS to as much as 160 FPS.

References

1. La Route Automatisée. 2019. Traffic Lights Recognition (TLR) public benchmarks. Retrieved from http://www.lara.prd.fr/benchmarks/trafficlightsrecognition
2. Martin Bach, Daniel Stumper, and Klaus Dietmayer. 2018. Deep convolutional traffic light recognition for automated driving. In Proceedings of the 21st International Conference on Intelligent Transportation System (ITSC'18). IEEE, 851-858.
3. Karsten Behrendt and Libor Novak. [n.d.]. A deep learning approach to traffic lights: Detection, tracking, and classification. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'17). IEEE