RWU
HOCHSCHULE
RAVENSBURG-WEINGARTEN
UNIVERSITY
OF APPLIED SCIENCES

Computer Vision

Project Report

“Evaluation of YOLOv3 object detector on KITTI Dataset”

Bhuvan Harlapur

Matriculation Nr- 32960

Abstract

Object detection is one of the most vital components in the Automotive sector. It can be used for various applications like Automotive safety, Autonomous driving and so on. Hence, various object detecting models have been developed over the years. One such model is the YOLOv3 which has been developed using the concepts of neural network and deep learning. But once this model is developed, it cannot be directly used in real life applications. The model needs to be evaluated by feeding it datasets containing images with known objects and check the accuracy of detection.

In this project we are trying to evaluate the convolutional neural network YOLOv3 object detector pre trained on COCO dataset. We are trying calculate the precision with which the pre trained YOLOv3 model can detect objects when a completely new dataset is provided to it. Therefore, this evaluation is done using KITTI dataset for three classes of objects namely Cars, Pedestrians and Trucks. The evaluation is carried out on PyCharm using Python and OpenCV library.

The project has been slightly deferred from the proposal where the earlier plan was to evaluate just one class of object of KITTI dataset on YOLOv3 model. Since the model can detect more no. of classes and data for those classes are also available in KITTI dataset, I decided to evaluate the YOLOv3 model for more no. of classes than just one class.

Table of Contents

List of Figures

1. Introduction

1.1 KITTI DATASET

"The KITTI dataset is a recording taken from a moving platform (Car fitted with 4 video cameras, a rotating 3D laser scanner and a combined GPS/IMU inertial navigation system) while driving in and around Karlsruhe, Germany. It includes camera images, laser scans, high-precision GPS measurements and IMU accelerations from a combined GPS/IMU system." (Andreas Geiger, 2013). It includes camera images of various classes like Cars, Pedestrians, Truck etc. which are 8 in total and also the label files containing ground truth values.

1.2 YOLOv3

YOLO stands for You Only Live Once, it is a convolutional neural network which is used to detect objects in an image. YOLO network has 24 convolutional layers and 2 fully connected layers. It is much faster than a Fast R-CNN (Joseph Redmon, 2016). YOLOv3 is the third improvised version of YOLO neural network which is pre-trained on COCO Dataset and has more convolutional layers for faster detection (Farhadi, 2018).

2. KITTI DATASET Contents.

From KITTI dataset, 1500 training images and the corresponding label files have been used.

- KITTI dataset contains 8 different classes which are Car, Van, Truck, Pedestrian, Person sitting, Cyclist, Tram, Misc. and DontCare.
- Each of these images contains its corresponding label files (.txt format) which are nothing but ground truth values.
- Each label file consists information of different objects that are present in the image. Class, Truncation, Occlusion, Observation angle, 2D bounding box boundary, 3D bounding box dimension, 3D bounding box coordinates and the Rotation value of the box are information given of each object.

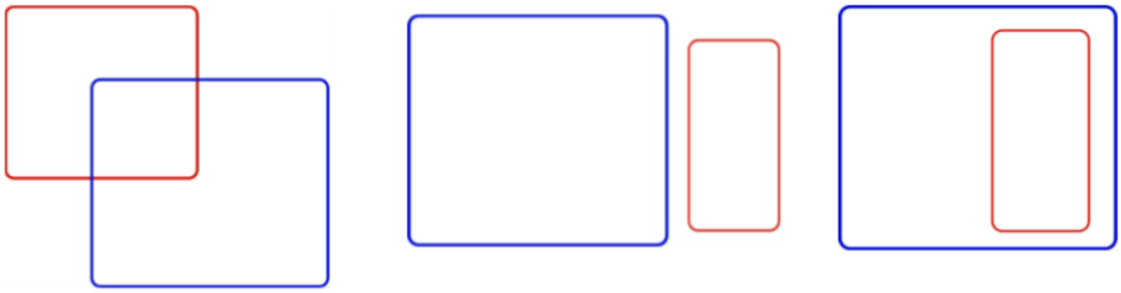
3. Evaluation Method.

We have to first get the predicted boxes as the output after feeding the images to the YOLOv3 object detector, once we get the predicted boxes, we have to compare these with the ground truth values available in KITTI dataset. The evaluation is done using Average Precision which takes into consideration both Precision and Recall.

Average precision can be calculated using the following method.

- Intersection over Union (IoU).

Intersection over union is based on Jaccard Index. Here, we require both predicted bounding box and ground truth box, intersection over union of the area of both these boxes are calculated. This ratio is nothing but Intersection over Union or IoU. Once we calculate the IoU we proceed to understand whether the detection is valid or not by calculating/assigning True Positives or False Positives with respect to the IoU. If IoU is greater than a certain set threshold then the detection is True Positive else it is False Positive. In the first figure given below there is a small amount of intersection between the Red Box (Ground truth) and Blue Box (Predicted Box) hence the IoU is greater than 0. In the send figure the IoU is equal to 0 as there is no area of intersection. In the third figure there the ground truth box is completely intersected in the predicted box but as we take the ratio, IoU is greater than 0 but less than 1. Hence, even if there is an intersection as given in third figure, the IoU still needs to be greater than the threshold and such intersection may still not be considered true positive.



0<IoU<1 (Berens, 2019)

IoU=0 (Berens, 2019)

0<IoU<1 (Berens, 2019)

Figure 1: Intersection over Union (IoU).

Once the no. of True positive's and False positives are found out, Precision and Recall have to be calculated.

$$\text{Precision - } \frac{TP}{TP+FP} \text{ or } \frac{TP}{Number\ of\ Predictions} \qquad (1)$$

$$\text{Recall- } \frac{TP}{TP+FN} \text{ or } \frac{TP}{No.of\ Ground\ truths} \qquad (2)$$

(Padilla, 2020)

After calculating the above values, we need to plot the curve for Precision and Recall and the area under this curve is Average Precision and then we take mean of all Average Precision's per class which is Mean Average Precision or mAP

To find out the area under the curve used the following method has been used.

- All Point Interpolation:

$$AP = \sum_{r=0}^{N} (r_{n+1} - r_n) * \rho(r_{n+1}) \qquad (3)$$

(Padilla, 2020)

4. Implementation

The implementation of the evaluation of object detector is done in the sequence as shown below.

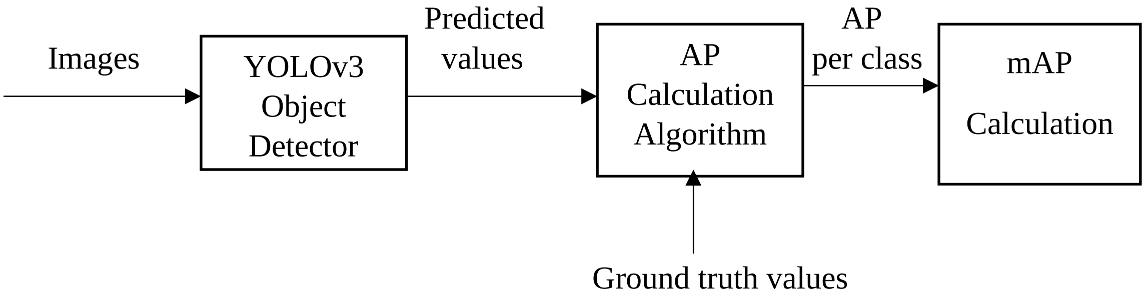


Figure 2: YOLOv3 object detection and evaluation implementation flow chart.

- The object detector takes the images files of the KITTI dataset as an input and the detects the objects giving out the predicted values.
- The predicted values are given out as a text file containing the class name confidence value and the co-ordinates x_{min}, y_{min}, x_{max} , y_{max}. An example for one class is shown below

Class Name	Confidence Score	x_{min}	y_{min}	x_{max}	y_{max}
Pedestrian	0.9980242252349854	723	146	807	301

- The predicted values for each object's detected is only taken if and only if the confidence values are greater than 0.3 or 30 percent.
- In the next step we move towards extraction of ground truth values for the label files which are in KITTI format. We extract data in the format as given below.

Class Name	x_{min}	y_{min}	x_{max}	y_{max}
Pedestrian	723	146	807	301

- Here the label files for each image contains Class, Truncation, Occlusion, Observation angle, 2D bounding box boundary, 3D bounding box dimension, 3D bounding box coordinates and the Rotation value of the box of each object of which we only need Class and 2D bounding box boundary.
- With all the above values we can draw bounding box with both predicted values and ground truth values for an image.

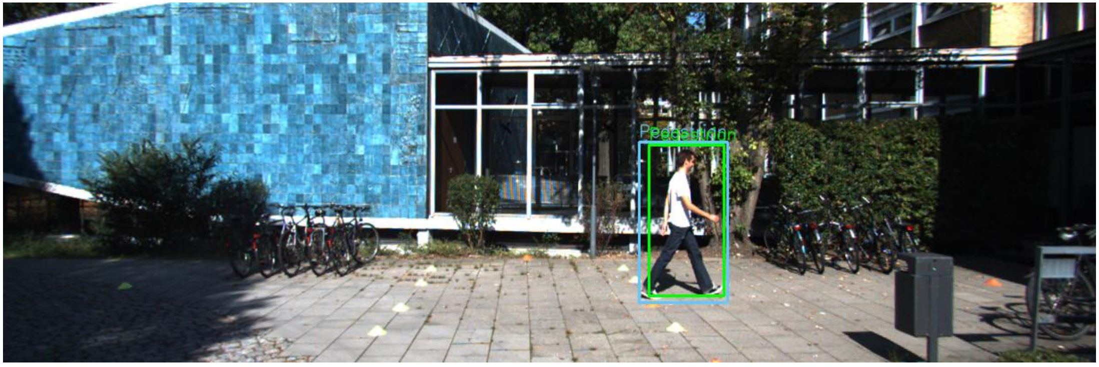


Figure 3: YOLOv3 Predictions and Ground truth predictions.

- In the above image the blue bounding box represents ground truth box and the green one is the predicted values box. In this image in spite of good detection by the object detector we need to calculate IoU which must be greater than certain threshold which in my case is 0.5.
- If the IoU value is greater than 0.5 we can consider the detection to be True Positive or else it is False positive.
- Likewise, we compute no. of True positives, False positives, Accumulated True Positives and Accumulated False Positives for all the predicted values for each class (Padilla, 2020).
- Using the precision and recall formulae, we calculate Precision and Recall for all the detection.
- This Precision v/s Recall curve is plotted for each class with precision monotonically decreasing. We then proceed with Average Precision and Mean Average Precision calculation.
- Average Precision Calculation
  - The Average precision calculation can be done with two methods namely, 11-point interpolation and All point interpolation method.
  - In this evaluation, All point interpolation method for calculation of average precision has been used as it is easier to program and also the results obtained between the two methods do not vary a lot.
  - Average precision using All point interpolation is calculated by find the area under the curve of precision v/s recall curve obtained for each class.
  - We take maximum precision between the present recall value and the last recall value
  - We then find the area under the curve using all recall values and maximum precision values by applying Average precision formula programically as given below

    Average_ Precision += ((recall[i]-recall[i-1])*max_precision[i]).

    where i = Index value

    recall = Recall value.

    max_precision = Maximum precision between present and the corresponding value.
- After calculating the Average precision on for all classes we take the ratio of sum of all average precision values and the no. of classes which gives us the Mean Average Precision or mAP.

4.1 Finding the Average Precision using the IoU method per class

Pseudo Code

GTB- Ground truth boxes and PB- Predicted Boxes

```
Initialize a 2-D array of 7000*9 for calculating precision and recall to zero
FOR the length of no. of Images (1500 in my case)
    IF GTB for an image is not 0
        For the length of no. of GTB
            For the length of no. of PB
                Calculate area of Intersection
                Calculate area of the union of two boxes
                Calculate Intersection over union (IoU)
                Initialize the IoU to another 2D array
        For the length of the PB (column)
            Find the maximum ratio across the column and also the row value at which it is maximum.
                IF ratio >0.5 (IoU)
                    Find the maximum across the row value
                    IF the maximum in the row value==maximum in column
                        TP=1
                        FP=0
                        Initialize relevant information like Image Name, Class Name, Confidence Score, TP and FP to the 2D array
                Else
                    TP=0
                    FP=1
                    Initialize relevant information like Image Name, Class Name, Confidence Score TP and FP to the 2D array
            IF the ratio<0.5 (IoU)
                TP=0
                FP=1
                Initialize relevant information like Image Name, Class Name, Confidence Score TP and FP to the 2D array
```

```
Sort the 2D array in descending w.r.t confidence score.
FOR the length of no. of rows of the sorted 2D array
        Calculate Accumulated TP
        Calculate Accumulated FP
        Calculate Precision
        Initialize the Accumulated TP, Accumulated FP and Precision to the 2D array
FOR the length of no. of rows of the sorted 2D array
        Calculate Recall
        Initialize the Accumulated TP, Accumulated FP and Precision to the 2D array
```

Initialize all precision and recall values from the 2D array to two different 1-Darray. With these Precision and Recall values we calculate Average Precision using All point interpolation method. Repeat the same logic for all the selected classes

5. Results and Disscusion

Using the above algorithm, the code was run for ***1500*** images, with IoU threshold being ***0.5***, no. of True Positives and False Positives, Precision v/s Recall curve and the Average precision was calculated for all the selected classes namely Cars, Pedestrians and Trucks. Using Average Precision per class we calculated Mean Average Precision(mAP) to be ***32.59%***

Class Name	Average Precision	No. of TP	No. of FP	Ground truth boxes	Predicted boxes
Car	60.21	4035	3056	5583	7091
Pedestrian	27.68	338	443	893	781
Truck	9.87	68	608	215	676

5.1 Precision v/s Recall Curves

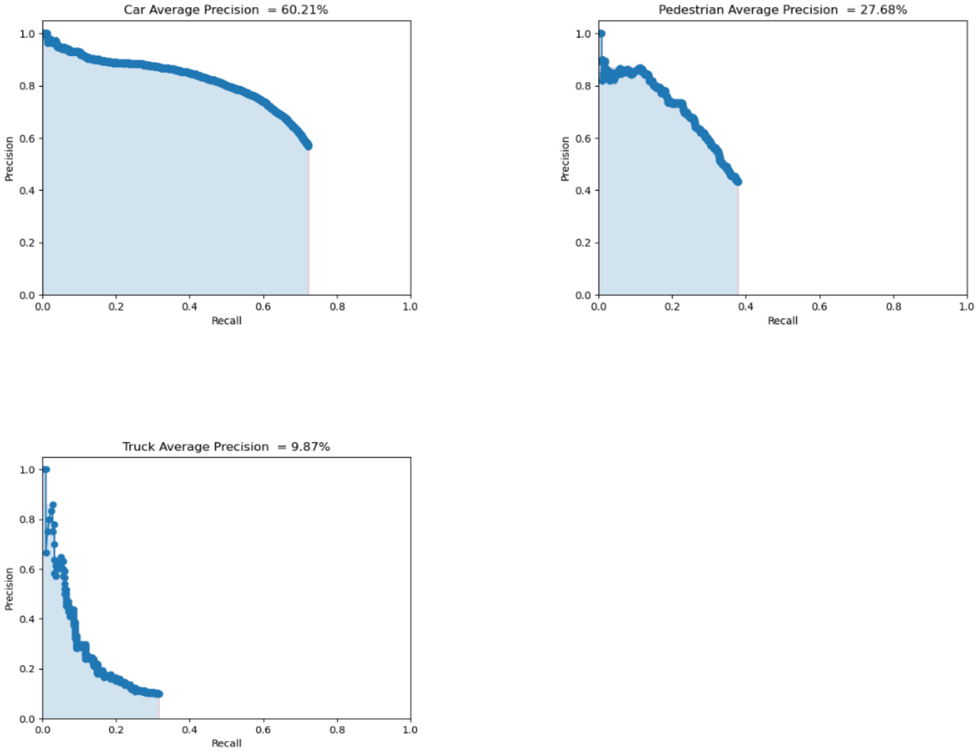


Figure 4: Precision v/s Recall curves for Cars, Pedestrians and Trucks.

6. Conclusion

If we look at the results closely the Average Precision values apart from that of the Car class are very low for both Truck and Pedestrians. Hence, this evaluation was conducted for only three classes as the Average Precision values for the rest of the classes of KITTI dataset were quite negligible. The YOLOv3 model trained on COCO dataset but when chosen to evaluate on KITTI dataset does well in recognising Car objects but falters with other class of objects. We have evaluated the model on IoU method, but there are many different methods like Centre point Comparison method which may have given different results. Looking at the KITTI leader board we see that there are better models which detect Cars, Pedestrians and Trucks with better results than YOLOv3 (Urtasun, 2012). Hence, we can say the object detector needs to be probably trained well and also the evaluation algorithm could be improvised for better evaluation results. Because in the current situation using this model in real life applications for detection of objects like Cars, Pedestrians, Truck etc would not be the best idea.

7. Future Scope of Work

Due to the lack of powerful GPU, I could only evaluate YOLOv3 model which was pretrained on COCO dataset. In the future the YOLOv3 could be completely trained on KITTI dataset on all classes of object with a powerful GPU which can train the model better and faster with a good training algorithm. We could also use a better evaluation algorithm which helps to improve the accuracy of the object detector which in turn will help in better detection of objects like Car, Pedestrians, Trucks etc.

References

Andreas Geiger, P. L. (2013). *Vision meets Robotics: The KITTI Dataset.* Karlsruhe, Deutschland: International Journal of Robotics Research. Retrieved June 21, 2020, from http://www.cvlibs.net/publications/Geiger2013IJRR.pdf

Berens, F. (2019, December). INTRODUCTION TO LIDAR AND KITTI. *(Course Material of LIDAR AND RADAR SYSTEMS)*. Retrieved May 2020

Farhadi, J. R. (2018). *YOLOv3: An Incremental Improvement.* arXiv. Retrieved from https://pjreddie.com/darknet/yolo/

Joseph Redmon, S. D. (2016). *You Only Look Once:Unified, Real-Time Object Detection.* arXiv:1506.02640v5 [cs.CV]. Retrieved from https://arxiv.org/pdf/1506.02640.pdf

Padilla, R. (2020). *Object-Detection-Metrics*. Retrieved June 2020, from https://github.com: https://github.com/rafaelpadilla/Object-Detection-Metrics

Urtasun, A. G. (2012). *"Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite"*. Retrieved June 2020, from Conference on Computer Vision and Pattern Recognition (CVPR): http://www.cvlibs.net/datasets/kitti/eval_object.php?obj_benchmark=2d