# A Survey on Performance Metrics for Object-Detection Algorithms

**Conference Paper** · July 2020

DOI: 10.1109/IWSSIP48289.2020

CITATIONS
113

READS
14,976

**3 authors:**

Rafael Padilla
Federal University of Rio de Janeiro
**11** PUBLICATIONS **476** CITATIONS
SEE PROFILE

Sergio Lima Netto
Federal University of Rio de Janeiro
**158** PUBLICATIONS **1,661** CITATIONS
SEE PROFILE

Eduardo A. B. da Silva
Federal University of Rio de Janeiro
**295** PUBLICATIONS **3,089** CITATIONS
SEE PROFILE

# A Survey on Performance Metrics for Object-Detection Algorithms

Rafael Padilla[1], Sergio L. Netto[2], Eduardo A. B. da Silva[3]
[1,2,3]PEE, COPPE, Federal University of Rio de Janeiro, P.O. Box 68504, RJ, 21945-970, Brazil
{*rafael.padilla, sergioln,eduardo*}*@smt.ufrj.br*

***Abstract*—This work explores and compares the plethora of metrics for the performance evaluation of object-detection algorithms. Average precision (AP), for instance, is a popular metric for evaluating the accuracy of object detectors by estimating the area under the curve (AUC) of the precision $\times$ recall relationship. Depending on the point interpolation used in the plot, two different AP variants can be defined and, therefore, different results are generated. AP has six additional variants increasing the possibilities of benchmarking. The lack of consensus in different works and AP implementations is a problem faced by the academic and scientific communities. Metric implementations written in different computational languages and platforms are usually distributed with corresponding datasets sharing a given bounding-box description. Such projects indeed help the community with evaluation tools, but demand extra work to be adapted for other datasets and bounding-box formats. This work reviews the most used metrics for object detection detaching their differences, applications, and main concepts. It also proposes a standard implementation that can be used as a benchmark among different datasets with minimum adaptation on the annotation files.**

***Keywords*—object-detection metrics, average precision, object-detection challenges, bounding boxes.**

## I. Introduction

Object detection is an extensively studied topic in the field of computer vision. Different approaches have been employed to solve the growing need for accurate object detection models [1]. The Viola-Jones framework [2], for instance, became popular due to its successful application in the face-detection problem [3], and was later applied to different subtasks such as pedestrian [4] and car [5] detections. More recently, with the popularization of the convolutional neural networks (CNN) [6]–[9] and GPU-accelerated deep-learning frameworks, object-detection algorithms started being developed from a new perspective [10], [11]. Works as Overfeat [12], R-CNN [13], Fast R-CNN [14], Faster R-CNN [15], R-FCN [16], SSD [17] and YOLO [18]–[20] highly increased the performance stantards on the field. World famous competitions such as VOC PASCAL Challenge [21], COCO [22], ImageNet Object Detection Challenge [23], and Google Open Images Challenge [24] have as their top object-detection algorithms methods inspired on the aforementioned works. Differently from algorithms such as the Viola-Jones, CNN-based detectors are flexible enough to be trained with several (hundreds or even a few thousands) classes.

A detector outcome is commonly composed of a list of bounding boxes, confidence levels and classes, as seen in Figure 1. However, the standard output-file format varies a lot for different detection algorithms. Bounding-box detections

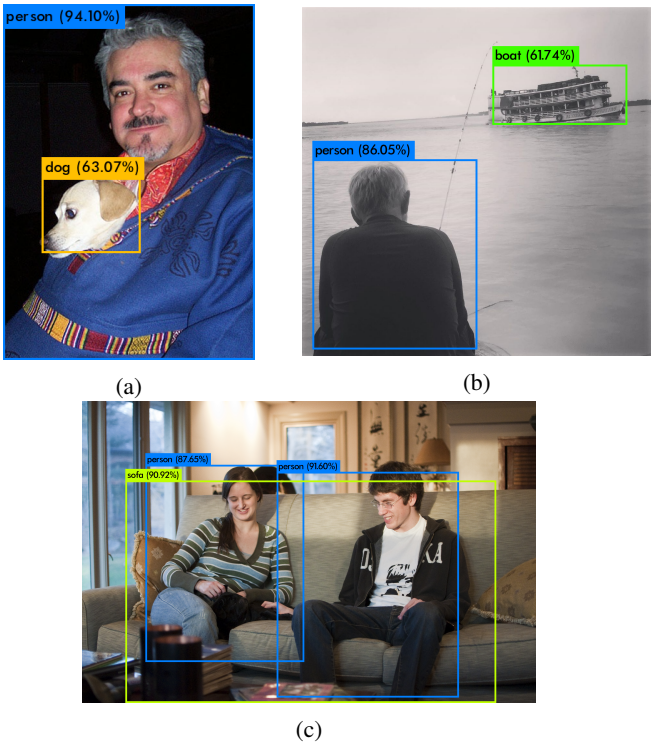


Fig. 1: Examples of detections performed by YOLO [20] in different datasets. (a) PASCAL VOC; (b) personal dataset; (c) COCO. Besides the bounding box coordinates of a detected object, the output also includes the confidence level and its class.

are mostly represented by their top-left and bottom-right coordinates $(x_{\text{ini}}, y_{\text{ini}}, x_{\text{end}}, y_{\text{end}})$, with a notable exception being the YOLO [18]–[20] algorithm, that differs from the others by outlining the bounding boxes by their center coordinates, width, and height $\left(\frac{x_{\text{center}}}{\text{image width}}, \frac{y_{\text{center}}}{\text{image height}}, \frac{\text{box width}}{\text{image width}}, \frac{\text{box height}}{\text{image height}}\right)$.

Different challenges, competitions, and hackathons [21], [23]–[27] attempt to assess the performance of object detections in specific scenarios by using real-world annotated images [28]–[30]. In these events, participants are given a testing nonannotated image set in which objects have to be detected by their proposed works. Some competitions provide their own (or 3rd-party) source code, allowing the participants to evaluate their algorithms in an annotated validation image set before submitting their testing-set detections. In the end,

each team sends a list of bounding-boxes coordinates with their respective classes and (sometimes) their confidence levels to be evaluated.

In most competitions, the average precision (AP) and its derivations are the metrics adopted to assess the detections and thus rank the teams. The PASCAL VOC dataset [31] and challenge [21] provide their own source code to measure the AP and the mean AP (mAP) over all object classes. The City Intelligence Hackathon [27] uses the source code distributed in [32] to rank the participants also on AP and mAP. The ImageNet Object Localization challenge [23] does not recommend any code to compute their evaluation metric, but provides a pseudo-code explaining it. The Open Images 2019 [24] and Google AI Open Images [26] challenges use mAP, referencing a tool to evaluate the results [33], [34]. The Lyft 3D Object Detection for Autonomous Vehicles challenge [25] does not reference any external tool, but uses the AP averaged over 10 different thresholds, the so-called AP@50:5:95 metric.

This work reviews the most popular metrics used to evaluate object-detection algorithms, including their main concepts, pointing out their differences, and establishing a comparison between different implementations. In order to introduce its main contributions, this work is divided into the following topics: Section II explains the main performance metrics employed in the field of object detection and how the AP metric can produce ambiguous results; Section III describes some of the most known object detection challenges and their employed performance metrics, whereas Section IV presents a project implementing the AP metric to be used with any annotation format.

## II. Main Performance Metrics

Among different annotated datasets used by object detection challenges and the scientific community, the most common metric used to measure the accuracy of the detections is the AP. Before examining the variations of the AP, we should review some concepts that are shared among them. The most basic are the ones defined below:

- True positive (TP): A correct detection of a ground-truth bounding box;
- False positive (FP): An incorrect detection of a nonexistent object or a misplaced detection of an existing object;
- False negative (FN): An undetected ground-truth bounding box;

It is important to note that, in the object detection context, a true negative (TN) result does not apply, as there are infinite number of bounding boxes that should not be detected within any given image.

The above definitions require the establishment of what a "correct detection" and an "incorrect detection" are. A common way to do so is using the intersection over union (IOU). It is a measurement based on the Jaccard Index, a coefficient of similarity for two sets of data [35]. In the object detection scope, the IOU measures the overlapping area between the predicted bounding box $B_p$ and the ground-truth bounding box $B_{gt}$ divided by the area of union between them, that is

$$J(B_p, B_{gt}) = \text{IOU} = \frac{\text{area}(B_p \cap B_{gt})}{\text{area}(B_p \cup B_{gt})}, \tag{1}$$

as illustrated in Figure 2.

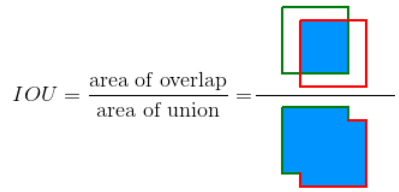


Fig. 2: Intersection Over Union (IOU).

By comparing the IOU with a given threshold $t$, we can classify a detection as being correct or incorrect. If IOU $\geq t$ then the detection is considered as correct. If IOU $< t$ the detection is considered as incorrect.

Since, as stated above, the true negatives (TN) are not used in object detection frameworks, one refrains to use any metric that is based on the TN, such as the TPR, FPR and ROC curves [36]. Instead, the assessment of object detection methods is mostly based on the precision $P$ and recall $R$ concepts, respectively defined as

$$P = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{\text{TP}}{\text{all detections}}, \tag{2}$$

$$R = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{\text{TP}}{\text{all ground truths}}. \tag{3}$$

Precision is the ability of a model to identify only relevant objects. It is the percentage of correct positive predictions. Recall is the ability of a model to find all relevant cases (all ground-truth bounding boxes). It is the percentage of correct positive predictions among all given ground truths.

The precision $\times$ recall curve can be seen as a trade-off between precision and recall for different confidence values associated to the bounding boxes generated by a detector. If the confidence of a detector is such that its FP is low, the precision will be high. However, in this case, many positives may be missed, yielding a high FN, and thus a low recall. Conversely, if one accepts more positives, the recall will increase, but the FP may also increase, decreasing the precision. However, a good object detector should find all ground-truth objects ($FN = 0 \equiv$ high recall) while identifying only relevant objects ($FP = 0 \equiv$ high precision). Therefore, a particular object detector can be considered good if its precision stays high as its recall increases, which means that if the confidence threshold varies, the precision and recall will still be high. Hence, a high area under the curve (AUC) tends to indicate both high precision and high recall. Unfortunately, in practical cases, the precision $\times$ recall plot is often a zigzag-like curve, posing challenges to an accurate measurement of its AUC. This is circumvented by processing the precision $\times$ recall curve in order to remove the zigzag behavior prior to AUC estimation. There are basically

two approaches to do so: the 11-point interpolation and all-point interpolation.

In the 11-point interpolation, the shape of the precision $\times$ recall curve is summarized by averaging the maximum precision values at a set of 11 equally spaced recall levels [0, 0.1, 0.2, ... , 1], as given by

$$\text{AP}_{11} = \frac{1}{11} \sum_{R \in \{0, 0.1, \ldots, 0.9, 1\}} P_{\text{interp}}(R), \tag{4}$$

where

$$P_{\text{interp}}(R) = \max_{\tilde{R}:\tilde{R} \geq R} P(\tilde{R}). \tag{5}$$

In this definition of AP, instead of using the precision $P(R)$ observed at each recall level $R$, the AP is obtained by considering the maximum precision $P_{\text{interp}}(R)$ whose recall value is greater than $R$.

In the all-point interpolation, instead of interpolating only 11 equally spaced points, one may interpolate through all points in such way that:

$$\text{AP}_{\text{all}} = \sum_{n} (R_{n+1} - R_n) P_{\text{interp}}(R_{n+1}), \tag{6}$$

where

$$P_{\text{interp}}(R_{n+1}) = \max_{\tilde{R}:\tilde{R} \geq R_{n+1}} P(\tilde{R}). \tag{7}$$

In this case, instead of using the precision observed at only few points, the AP is now obtained by interpolating the precision at each level, taking the maximum precision whose recall value is greater or equal than $R_{n+1}$.

The mean AP (mAP) is a metric used to measure the accuracy of object detectors over all classes in a specific database. The mAP is simply the average AP over all classes [15], [17], that is

$$\text{mAP} = \frac{1}{N} \sum_{i=1}^{N} \text{AP}_i, \tag{8}$$

with $AP_i$ being the AP in the $i$th class and $N$ is the total number of classes being evaluated.

### A. A Practical Example

As stated previously, the AP is calculated individually for each class. In the example shown in Figure 3, the boxes represent detections (red boxes identified by a letter - *A*, *B*, ..., *Y*) and the ground truth (green boxes) of a given class. The percentage value drawn next to each red box represents the detection confidence for this object class. In order to evaluate the precision and recall of the 24 detections among the 15 ground-truth boxes distributed in seven images, an IOU threshold $t$ needs to be established. In this example, let us consider as a TP detection box one having IOU $\geq 30\%$. Note that each value of IOU threshold provides a different AP metric, and thus the threshold used must always be indicated.

Table I presents each detection ordered by their confidence level. For each detection, if its area overlaps 30% or more of a ground truth (IOU $\geq$ 30%), the TP column is identified as 1; otherwise it is set to 0 and it is considered as FP. Some detectors can output multiple detections overlapping a single ground truth (e.g. detections D and E in Image 2; G, H and I in Image 3). For those cases the detection with the highest confidence is considered a TP and the others are considered as FP, as applied by the PASCAL VOC 2012 challenge. The columns *Acc TP* and *Acc FP* accumulate the total amount of TP and FP along all the detections above the corresponding confidence level. Figure 4 depicts the calculated precision and recall values for this case.

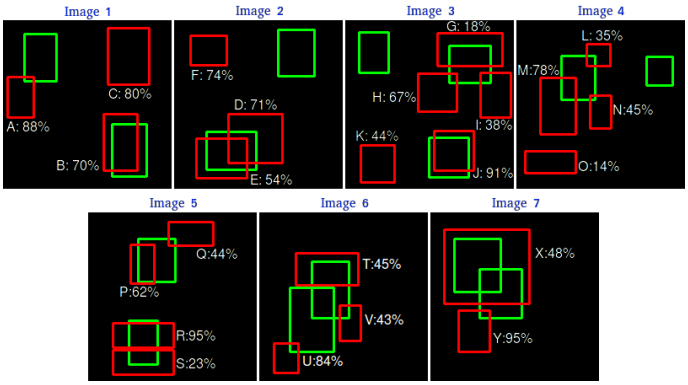


Fig. 3: Example of 24 detections (red boxes) performed by an object detector aiming to detect 15 ground-truth objects (green boxes) belonging to the same class.

TABLE I: Computation of Precision and Recall Values for IOU threshold = 30%

| detection | confidence | TP | FP | acc TP | acc FP | precision | recall |
|---|---|---|---|---|---|---|---|
| R | 95% | 1 | 0 | 1 | 0 | 1 | 0.0666 |
| Y | 95% | 0 | 1 | 1 | 1 | 0.5 | 0.0666 |
| J | 91% | 1 | 0 | 2 | 1 | 0.6666 | 0.1333 |
| A | 88% | 0 | 1 | 2 | 2 | 0.5 | 0.1333 |
| U | 84% | 0 | 1 | 2 | 3 | 0.4 | 0.1333 |
| C | 80% | 0 | 1 | 2 | 4 | 0.3333 | 0.1333 |
| M | 78% | 0 | 1 | 2 | 5 | 0.2857 | 0.1333 |
| F | 74% | 0 | 1 | 2 | 6 | 0.25 | 0.1333 |
| D | 71% | 0 | 1 | 2 | 7 | 0.2222 | 0.1333 |
| B | 70% | 1 | 0 | 3 | 7 | 0.3 | 0.2 |
| H | 67% | 0 | 1 | 3 | 8 | 0.2727 | 0.2 |
| P | 62% | 1 | 0 | 4 | 8 | 0.3333 | 0.2666 |
| E | 54% | 1 | 0 | 5 | 8 | 0.3846 | 0.3333 |
| X | 48% | 1 | 0 | 6 | 8 | 0.4285 | 0.4 |
| N | 45% | 0 | 1 | 6 | 9 | 0.4 | 0.4 |
| T | 45% | 0 | 1 | 6 | 10 | 0.375 | 0.4 |
| K | 44% | 0 | 1 | 6 | 11 | 0.3529 | 0.4 |
| Q | 44% | 0 | 1 | 6 | 12 | 0.3333 | 0.4 |
| V | 43% | 0 | 1 | 6 | 13 | 0.3157 | 0.4 |
| I | 38% | 0 | 1 | 6 | 14 | 0.3 | 0.4 |
| L | 35% | 0 | 1 | 6 | 15 | 0.2857 | 0.4 |
| S | 23% | 0 | 1 | 6 | 16 | 0.2727 | 0.4 |
| G | 18% | 1 | 0 | 7 | 16 | 0.3043 | 0.4666 |
| O | 14% | 0 | 1 | 7 | 17 | 0.2916 | 0.4666 |

As mentioned above, each interpolation method yields a different AP result, as given by (Figure 5):

$$\text{AP}_{11} = \frac{1}{11}(1 + 0.6666 + 0.4285 + 0.4285 + 0.4285)$$
$$\text{AP}_{11} = 26.84\%,$$

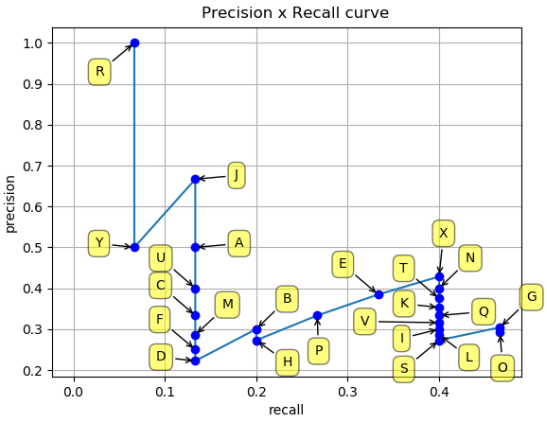


Fig. 4: Precision x Recall curve with values calculated for each detection in Table I.

and (Figure 6):

$$\begin{aligned} \text{AP}_{\text{all}} &= 1 * (0.0666 - 0) + 0.6666 * (0.1333 - 0.0666) \\ &\quad + 0.4285 * (0.4 - 0.1333) + 0.3043 * (0.4666 - 0.4) \\ \text{AP}_{\text{all}} &= 24.56\%. \end{aligned}$$

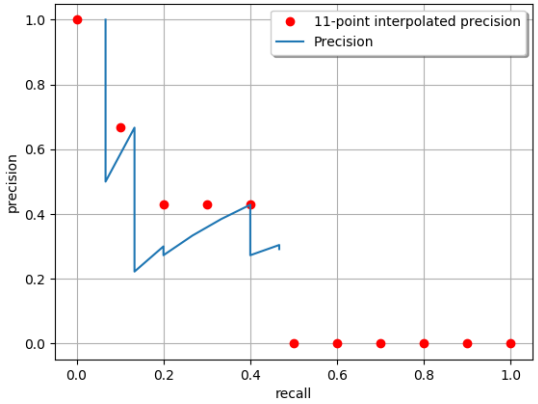


Fig. 5: Precision $\times$ Recall curves of points from Table I using the 11-point interpolation approach.

From what we have seen so far, benchmarks are not truly comparable if the method used to calculate the AP is not reported. Works found in the literature [1], [9], [12]–[20], [37] usually neither mention the method used nor reference the adopted tool to evaluate their results. This problem does not occur much often in challenges, as it is a common practice to have a reference software tool included in order for the participants to evaluate their results. Also, it is not rare to occur cases where a detector sets the same confidence level for different detections. Table I, for example, illustrates that detections R and Y obtained the same confidence level (95%). Depending on the criterion used by a certain implementation, one or other detection can be sorted as the first detection in the table, directly affecting the final result of an object-detection algorithm. Some implementations may consider the order that

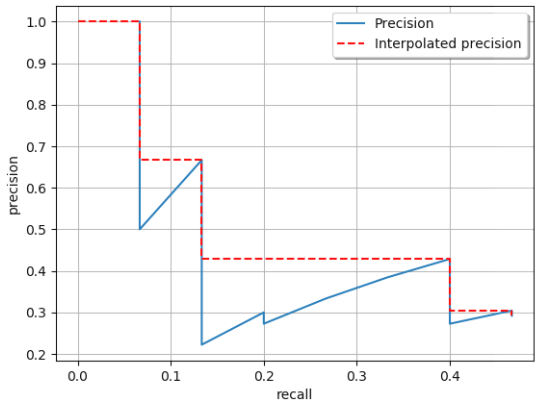


Fig. 6: Precision $\times$ Recall curves of points from Table I applying interpolation with all points.

each detection was reported as the tiebreaker (usually one or more evaluation files contain the detections to be evaluated), but in general there is no common consensus by the evaluation tools.

## III. Object-Detection Challenges and Their AP Variants

Constantly, new techniques are being developed and new different state-of-the-art object-detection algorithms are arising. Comparing their results with different works is not an easy task. Sometimes the applied metrics vary or the implementation used by the different authors may not be the same, generating dissimilar results. This section covers the main challenges and their most popular AP variants found in the literature.

The PASCAL VOC [31] is an object-detection challenge released in 2005. From 2005 to 2012, a new version of the Pascal VOC was released with increased numbers of images and classes, starting at four classes, reaching 20 classes in its last update. The PASCAL VOC competition still accepts submissions, revealing state-of-the-art algorithms for object detections ever since. In this trail, the challenge applies the 11-point interpolated precision (see Section II) and uses the mean AP over all of its classes to rank the submission performances, as implemented by the provided development kit.

The Open Images 2019 challenge [24] in its object-detection track uses the Open Images Dataset [29] containing 12.2 M annotated bounding boxes across 500 object categories on 1.7 M images. Due to its hierarchical annotations, the same object can belong to a main class and multiple sub-classes (e.g. 'helmet' and 'football helmet'). Because of that, the users should report the class and subclasses of a given detection. If somehow only the main class is correctly reported for a detected bounding box, the unreported subclasses affect negatively the score, as it is counted as a false negative. The metric employed by the aforementioned challenge is the mean AP over all classes using the Tensorflow Object Detection API [33].

The COCO detection challenge (bounding box) [22] is a competition which provides bounding-box coordinates of more than 200,000 images comprising 80 object categories. The

submitted works are ranked according to metrics gathered into four main groups.

- AP: The AP is evaluated with different IOUs. It can be calculated for 10 IOUs varying in a range of 50% to 95% with steps of 5%, usually reported as AP@50:5:95. It also can be evaluated with single values of IOU, where the most common values are 50% and 75%, reported as AP50 and AP75 respectively;
- AP Across Scales: The AP is determined for objects in three different sizes: small (with area $< 32^2$ pixels), medium (with $32^2 <$ area $< 96^2$ pixels), and large (with area $> 96^2$ pixels);
- Average Recall (AR): The AR is estimated by the maximum recall values given a fixed number of detections per image (1, 10 or 100) averaged over IOUs and classes;
- AR Across Scales: The AR is determined for objects in the same three different sizes as in the AP Across Scales, usually reported as AR-S, AR-M, and AR-L, respectively;

Tables II and III present results obtained by different object detectors for the COCO and PASCAL VOC challenges, as given in [20], [38]. Due to different bounding-box annotation formats, researchers tend to report only the metrics supported by the source code distributed with each dataset. Besides that, works that use datasets with other annotation formats [39] are forced to convert their annotations to PASCAL VOC's and COCO's formats before using their evaluation codes.

TABLE II: Results using AP variants obtained by different methods on COCO dataset [40].

| methods | AP@50:5:95 | AP50 | AP75 | AP-S | AP-M | AP-L |
|---|---|---|---|---|---|---|
| Faster R-CNN with ResNet-101 [9], [15] | 34.9 | 55.7 | 37.4 | 15.6 | 38.7 | 50.9 |
| Faster R-CNN with FPN [15], [41] | 36.2 | 59.1 | 39.0 | 18.2 | 39.0 | 48.2 |
| Faster R-CNN by G-RMI [15], [42] | 34.7 | 55.5 | 36.7 | 13.5 | 38.1 | 52.0 |
| Faster R-CNN with TDM [15], [43] | 36.8 | 57.7 | 39.2 | 16.2 | 39.8 | 52.1 |
| YOLO v2 [19] | 21.6 | 44.0 | 19.2 | 5.0 | 22.4 | 35.5 |
| YOLO v3 [20] | 33.0 | 57.9 | 34.4 | 18.3 | 35.4 | 41.9 |
| SSD513 with ResNet-101 [9], [17] | 31.2 | 50.4 | 33.3 | 10.2 | 34.5 | 49.8 |
| DSSD513 with ResNet-101 [9], [44] | 33.2 | 53.3 | 35.2 | 13.0 | 35.4 | 51.1 |
| RetinaNet [40] | 39.1 | 59.1 | 42.3 | 21.8 | 42.7 | 50.2 |

TABLE III: Results using AP variant (mAP) obtained by different methods on PASCAL VOC 2012 dataset [38].

| methods | mAP |
|---|---|
| Faster R-CNN * [15] | 70.4 |
| YOLO v1 [18] | 57.9 |
| YOLO v2 ** [19] | 78.2 |
| SSD300 ** [17] | 79.3 |
| SSD512 ** [17] | 82.2 |

(*) trained with PASCAL VOC dataset images only, while (**) trained with COCO dataset images.

The metric AP50 in Table II is calculated in the same way as the metric mAP in Table III, but as the methods were trained and tested in different datasets, one obtains different results in both evaluations. Due to the need of conversions between the bounding-box annotations among different datasets, researchers in general do not evaluate all methods with all possible metrics. In practice, it would be more meaningful if methods trained and tested with one dataset (PASCAL VOC, for instance) could also be evaluated by the metrics employed in other datasets (COCO, for instance).

## IV. An Open-Source Performance Metric Repository

In order to help other researchers and the academic community to obtain trustworthy results that can be comparable regardless the detector, the database, or the format of the ground-truth annotations, a library was developed in Python with the AP metric that can be extended to its variations. Easy-to-use functions implement the same metrics used as benchmark by the most popular competitions and object-detection researches. The proposed implementation does not require modifications of the detection model to match complicated input formats, avoiding conversions to XML or JSON files. To assure the accuracy of the results, the implementation followed to the letter the definitions and our results were carefully compared against the official implementations and the results are precisely the same. The variations of the AP metric such as mAP, AP50, AP75 and AP@50:5:95 using the 11-point or the all-point interpolations can be obtained with the proposed library.

The input data (ground-truth bounding boxes and detected bounding boxes) format was simplified requiring a single format to compute all AP variation metrics. The format required is straightforward and can support the most popular detectors. For the ground-truth bounding boxes, a single text file for each image should be created with each line in one of the following formats:

*<class> <left> <top> <right> <bottom>*
*<class> <left> <top> <width> <height>*

For the detections, a text file for each image should include a line for each bounding box in one of the following formats:

*<class> <confidence> <left> <top> <right> <bottom>*
*<class> <confidence> <left> <top> <width> <height>*

The second options support YOLO's output bounding-box formats. Besides specifying the input formats of the bounding boxes, one can also set the IOU threshold used to consider a TP (useful to calculate the metrics AP@50:5:95, AP50 and AP75) and the interpolation method (11-point interpolation or interpolation with all points). The tool will output the plots as in Figures 5 and 6, the final mAP and the AP for each class, giving a better view of the results for each class. The tool also provides an option to generate the output images with the bounding boxes drawn on it as shown in Figure 1.

The project distributed with this paper can be accessed at: *https://github.com/rafaelpadilla/Object-Detection-Metrics*. So far, our framework has helped researchers to obtain AP metrics and its variations in a simple way, supporting the most popular formats used by datasets, avoiding conversions to XML or JSON files. The proposed tool has been used as the official tool in the competition [27], adopted in 3rd-party libraries such as [45] and used by many other works as in [46]–[48].

## REFERENCES


[1] W. Hu, T. Tan, L. Wang, and S. Maybank, "A survey on visual surveillance of object motion and behaviors," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 34, no. 3, pp. 334–352, Aug 2004.

[2] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, Dec 2001, p. 511518.

[3] R. Padilla, C. Costa Filho, and M. Costa, "Evaluation of haar cascade classifiers designed for face detection," *World Academy of Science, Engineering and Technology*, vol. 64, pp. 362–365, 2012.

[4] E. Ohn-Bar and M. M. Trivedi, "To boost or not to boost? on the limits of boosted trees for object detection," in *IEEE International Conference on Pattern Recognition*, Dec 2016, pp. 3350–3355.

[5] Z. Sun, G. Bebis, and R. Miller, "On-road vehicle detection: A review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 5, pp. 694–711, May 2006.

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *International Conference on Neural Information Processing Systems*, 2012, pp. 1097–1105.

[7] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *IEEE Conference on Computer Vision and Pattern Recognition*, June 2015, pp. 1–9.

[8] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, 1998, pp. 2278–2324.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition*, Jun 2016, pp. 770–778.

[10] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[11] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, Jul. 2006.

[12] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," *CoRR*, 2013.

[13] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition*, Jun 2014.

[14] R. Girshick, "Fast r-cnn," in *IEEE International Conference on Computer Vision*, Dec 2015.

[15] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems 28*, 2015, pp. 91–99.

[16] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: object detection via region-based fully convolutional networks," *CoRR*, 2016.

[17] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, and A. C. Berg, "SSD: single shot multibox detector," *CoRR*, 2015.

[18] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779–788.

[19] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7263–7271.

[20] ——, "Yolov3: An incremental improvement," *Technical Report*, 2018.

[21] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *International Journal of Computer Vision*, vol. 111, no. 1, pp. 98–136, Jan. 2015.

[22] Coco detection challenge (bounding box). [Online]. Available: https://competitions.codalab.org/competitions/20794

[23] ImageNet. Imagenet object localization challenge. [Online]. Available: https://www.kaggle.com/c/imagenet-object-localization-challenge/

[24] G. Research. Open images 2019 - object detection challenge. [Online]. Available: https://www.kaggle.com/c/open-images-2019-object-detection/

[25] Lyft. Lyft 3d object detection for autonomous vehicles. [Online]. Available: https://www.kaggle.com/c/3d-object-detection-for-autonomous-vehicles/

[26] G. Research. Google ai open images - object detection track. [Online]. Available: https://www.kaggle.com/c/google-ai-open-images-object-detection-track/

[27] City intelligence hackathon. [Online]. Available: https://belvisionhack.ru

[28] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[29] I. Krasin, T. Duerig, N. Alldrin, V. Ferrari, S. Abu-El-Haija, A. Kuznetsova, H. Rom, J. Uijlings, S. Popov, S. Kamali, M. Malloci, J. Pont-Tuset, A. Veit, S. Belongie, V. Gomes, A. Gupta, C. Sun, G. Chechik, D. Cai, Z. Feng, D. Narayanan, and K. Murphy, "Openimages: A public dataset for large-scale multi-label and multi-class image classification," 2017.

[30] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: common objects in context," *CoRR*, 2014.

[31] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, Jun. 2010.

[32] R. Padilla. Metrics for object detection. [Online]. Available: https://github.com/rafaelpadilla/Object-Detection-Metrics

[33] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015.

[34] TensorFlow. Detection evaluation protocols. [Online]. Available: https://github.com/tensorflow

[35] P. Jaccard, "Étude comparative de la distribution florale dans une portion des alpes et des jura," *Bulletin de la Societe Vaudoise des Sciences Naturelles*, vol. 37, pp. 547–579, 1901.

[36] J. A. Hanley and B. J. McNeil, "The meaning and use of the area under a receiver operating characteristic (roc) curve." *Radiology*, vol. 143, no. 1, pp. 29–36, 1982.

[37] D. Yoo, S. Park, J.-Y. Lee, A. S. Paek, and I. So Kweon, "Attentionnet: Aggregating weak directions for accurate object detection," in *IEEE International Conference on Computer Vision*, 2015, pp. 2659–2667.

[38] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3212–3232, 2019.

[39] "An annotated video database for abandoned-object detection in a cluttered environment," in *International Telecommunications Symposium*, 2014, pp. 1–5.

[40] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *IEEE International Conference on Computer Vision*, 2017, pp. 2980–2988.

[41] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2117–2125.

[42] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama *et al.*, "Speed/accuracy trade-offs for modern convolutional object detectors," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7310–7311.

[43] A. Shrivastava, R. Sukthankar, J. Malik, and A. Gupta, "Beyond skip connections: Top-down modulation for object detection," *arXiv*, 2016.

[44] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, "Dssd: Deconvolutional single shot detector," *arXiv*, 2017.

[45] C. R. I. of Montreal (CRIM). thelper package. [Online]. Available: https://thelper.readthedocs.io/en/latest/thelper.optim.html

[46] C. Adleson and D. C. Conner, "Comparison of classical and cnn-based detection techniques for state estimation in 2d," *Journal of Computing Sciences in Colleges*, vol. 35, no. 3, pp. 122–133, 2019.

[47] A. Borji and S. M. Iranmanesh, "Empirical upper-bound in object detection and more," *arXiv*, 2019.

[48] D. Caschili, M. Poncino, and T. Italia, "Optimization of cnn-based object detection algorithms for embedded systems," Masters dissertation, Politecnico di Torino, 2019.