Turkish Online Journal of Qualitative Inquiry (TOJQI)
Volume 12, Issue 7, July 2021: 8268 - 8275

Research Article

Object Regognition and Detection using Yolo V3

J.Srilatha [a], Dr. T.S.Subhashini[b], Dr. K.Vaidehi [c]

[a] Research Scholar, [b]Professor, [c]Associate Professor
[a,b,c] Dept. of CSE, Stanley College of Enginering and Technology for Women,Hyderabad.

[a]jsrilatha@stanley.edu.in,[b]rtramsuba@gmail.com,[c]kvaidehi@stanley.edu.in

Abstract

Object detection and identification is an entrancing field, and as such, apart from research applications automatic object detection applications are being witnessed in real-time domains such as non-stop surveillance and in different industries and businesses. The improved capabilities in both hardware and software have led to fast and pivotal discoveries in this arena. Convolutional neural network (CNN) is the most representative model of deep learning and in this paper, a variant of CNN namely YOLO v3 is experimented with for localizing twenty different object classes that include aeroplane, person, car etc. The work is done using darknet-53 pre-trained model as the backbone network using open CV python, and TensorFlow 2.0. Images taken from two different datasets namely COCO and Pascal VOC datasets are given as input to the model and theoutput in the form of bounding boxes, accompanied by objectness score and class label. The results indicate that YOLO V3 is very efficient in object detection.

Keywords: Object Detection and Recognition, YOLO, YOLO v1, YOLO v2, YOLO v3

1. Introduction

Availability of accurate and fast object detection methods is the prerequisite for advanced visual applications viz., content-based image and video recognition [1]. Earlier for object detection researchers have experimented with traditional methods based on manual features. Though they were partially successful, these traditional methods due to their low accuracy and speed and poor environmental adaptability have been gradually replaced by neural networks. [2-7].Though a wide range of approaches to object detection and recognition has been tried out by researchers, there still isn't a conspicuous or even "best" approach that could be considered a optimal solution for all object detection and recognition pitfalls, which means there's still a number of research gaps that could be addressed to arrive at an optimal object detector.

Today, deep learning has found itself a deep-rooted place into digital image processing and has proved to be definite game changer especially in field of computer vision. Deep learning has squashed the other old-style machine learning techniques as far as classification tasks are concerned and today it is also proving to be the cutting-edge technique for object detection applications. Though deep learning was known as a concept as early as 1960's the rise of datasets such as ImageNet [8] and rapid emergence of parallel computing systems, such as GPU clusters has helped to visualize its learning potential.

Training the network has become quite easy and at the same time efficient due to its advanced network structures, training strategies and batch normalization techniques adopted [9] in deep neural networks. CNN being the highly recognized model of deep learning research works based on CNN and its variants for object detection and recognition are portrayed in these review papers [11] [12] and [13]. In these papers a exhaustive analysis of the deep learning-based object detection frameworks, typical deep learning architecture, public datasets available and the challenges faced by the researchers are given. The reviews indicate that two

approaches to object detection is generally followed. The first approach generates region proposals which are then classified into different object categories. In this a coarse scan of the whole image is done firstly and then regions of interest are focused upon. Algorithms belonging to RCNN [14] family, i.e. Fast RCNN [15] and Faster RCNN [16], Mask R-CNN [13] uses this approach. The other approach considers object detection as a regression problem, to classify objects into different categories. MultiBox [17], AttentionNet [18], G-CNN [19], YOLO [20], SSD [21], YOLOv2 [22], DSSD [23] and DSOD [24] come under the second category.

The ultimate aim of any real-time object recognition system is fast and efficient recognition. Though R-CNN, FRCNN, faster R-CNN was successful with better recognition speed, they are still considered slower for real-time object detection applications such as navigation of self-driving vehicles, quality control of parts in manufacturing, among many, many other things [10]. This paper exhibits one of the outstanding variants of CNN viz., You Only Look Once (YOLO), as a tool for effective object detection. The rest of the paper is organized as follows. Section 2 examines YOLO and the variants of YOLO. Section 3 gives the experimental details and the object detection results of YOLO V3 on COCCO and Pascal VOC datasets. Conclusion of the paper is done in Section 4.

2. YOLO

Two-stage object detection methods based on Region Proposal Networks (RPN) rely heavily on powerful GPU computing power. Two stage detectors such as R-CNN [25], Fast R-CNN [14], Faster R-CNN [15] and Mask R-CNN [20] produce accurate object detection results but the computational cost is very high. Real time applications cannot afford high computational capabilities and as such improvement in accuracy with heavy computational cost may not be helpful in such situations. The ability of models to perform object detection in real time has become the big necessity especially with the emergence technologies such as autonomous vehicles [16]. One of the pre-requisites for the success of technologies such as autonomous vehicles, augmented reality [26], etc is the availability of a reliable, fast and efficient object detection and recognition system.

To overcome the drawbacks of two stage detectors, some fast single-stage object detectors, such as SSD [27], DSSD [28], YOLO series models [29][30][31], RetinaNet [32], etc have been proposed by the researchers in recent times. Unlike two stage detectors, single stage object detectors takes a gander at the entire image at test time, so its predictions are based on the image as a whole. Further, predictions are done with a single network assessment unlike two stage detectors such as R-CNN which require thousands of assessments for a single image. This results in improved speed and is more than 1000x faster than R-CNN and 100x faster than Fast R-CNN. However, one stage detectors fails to attain better detection results and continuous changes are under process by researchers. YOLO V3 is one among the one stage detector which well balances the detection accuracy and the speed. Though YOLOv3 demands heavy computational cost and large run-time memory it is one of the popular one stage detectors because of its speed and reliability. To suit real world applications which can afford only very little computational power the YOLO series, have lightweight versions called YOLO-tiny [33] and YOLO-Lite [10].

YOLO V1

For training the model YOLO v1 employs the Darknet framework and images for training are taken from the ImageNet- dataset. First, YOLO v1 divides the image into MxM grid and each grid cell predicts a number of boundary boxes for an object [35]. The bounding-box parameters are *(x, y, w, h, confidence score) where x* and *y* are the object coordinates, *w* and *h* are the width and height of the object respectively. *Confidence score* is the probability that box contains an object and how accurate is the bounding box. One of the drawbacks of YOLO V1 is that if the objects are close together, it fails to detect the small objects. Further, if dimension of the objects in test images are different from those that were in the dataset, YOLO V1 was not that successful in detecting objects [14][16].

2.2. YOLO V2

Yolo v2 uses darknet -19 framework and to improve accuracy, batch normalization and anchor boxes are employed in YoloV2. This has considerably improved performance and also helps to arrive at a striking balance between processing time and accuracy. Unlike YOLO V1, YOLO v2 uses a custom deep architecture with 30 layers. It appended the darknet-19-layer network with 11 more layers to make it a 30 layer deep network. YOLO v2 was not efficient in detecting small objects and this is because the image got down sampled as it passes through the 30 layers resulting in the loss of fine-grained features. To cater to this problem YOLO v2 captures low level features using an identity mapping, concatenating feature maps from the previous layer.

2.3 YOLO V3

YOLOv3 algorithm has become a current research hotspot as it is more accurate and fast compared to earlier

versions namely YOLO V1 and YOLO V2. The YOLOv3 algorithm strikes a balance between speed and detection accuracy by employing the idea of Feature Pyramid Networks (FPN) to predict objects at multi-scales [15] and deep residual network (ResNet)[36] to extract features [20]. Since object predictions are done at various scales, YOLO V3 is able to detect small objects too. The network structure of the YOLOv3 algorithm is shown in Fig. 1. The usage of residual network helps in improving the feature extraction process, and the basic backbone network used is Darknet-53 which aids in extracting deeper feature information when compared to Darknet-19 employed in YOLO V2.

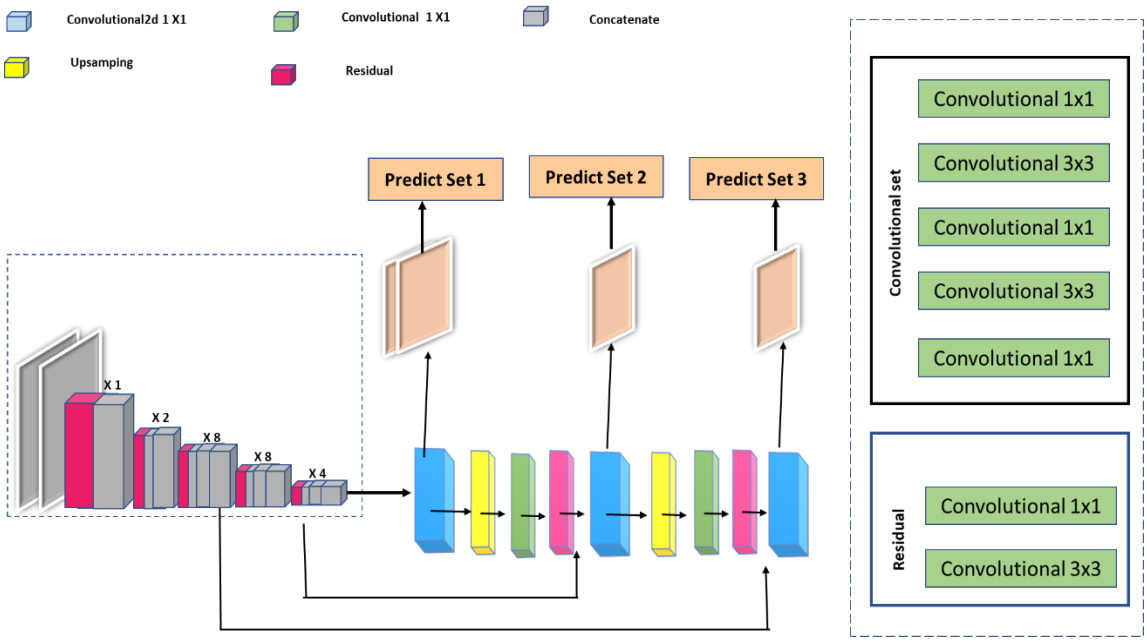

Fig.1. Network Structure of YOLO V3

YOLO works independent of the size of the input image. However, to process images in batch we need to have all images of fixed height and width and here the images are resized to 416x416. The Darknet-53 network in YOLOv3 uses alternating 1 × 1 and 3 × 3 convolutional kernels and after each layer of convolution[37][38], the BN layer is used for normalization. Darknet uses Leaky Relu activation function and to reduce the size of the feature map the convolution kernel size is increased. The network stride is 32 which down samples the 416x416 input image to 13x13 output image.

3. Experiments and Results

Detecting an object is easy for humans but for computers, it is not that simple. Of late, more and more accurate object detection has been witnessed with the advent of deep learning techniques. With the application of deep learning models, today, we are able to achieve good results to an extent, that it could be used in real-time scenarios, which was not the case earlier. In this paper, the implementation of YOLOv3 in Python is done. COCO and Pascal VOC datasets are used in this study to evaluate the performance of YOLO V3. Common Objects in Context (COCO) dataset as the name spells out, refers to a collection of images captured from real time scenarios. The COCO dataset contains 80 different object categories of labeled and segmented images and can be download from [39]. Yet another popular dataset is the Pascal VOC which is widely used in evaluating algorithms for image classification, object detection, and segmentation [4][40][41].

The input to YOLO v3 is a batch of colour images of dimension 416x416. The output is a set of bounding boxes with recognized class probabilities. The prediction of the bounding box is done by using a convolutional layer which uses 1 x 1 and 3x3 convolutions. Prediction across three different features maps of scales 13x13, 26 x26 and 52x52 is done in YOLO v3. That is, the detection layer makes detection on feature maps of three different sizes, having strides 32, 16, 8 respectively. Further with three different anchors, YOLO predicts three bounding boxes per cell. Anchors are nothing but bounding box priors, that are calculated using K-means clustering on COCO dataset. The anchors are different for different scales. At each scale, each cell predicts 3 bounding boxes using 3 anchors, making the total number of anchors used 9. The attributes of the bounding box predicted is represented as 5 + C which indicate the center coordinates, the dimensions, the objectness score and

C class confidences for each bounding box. The bounding box dimension namely bb_x, bb_y, bb_w, bb_h are predicted from the network output and anchors as per the equations given below.

$$bb_x = \sigma(t_x) + c_x$$

$$bb_y = \sigma(t_y) + c_y$$

$$bb_w = A_w e^{t_w}$$

$$bb_h = eA_h{}^{t_h}$$

Here t_x, t_y, t_w, t_h is the network outputs, c_x and c_y are the top-left coordinates of the cell and A_w and A_h are anchors dimensions. Yolo V3 predicts bounding boxes at three different scales and so for an image of size 416 x 416, YOLO predicts ((52 x 52) + (26 x 26) + 13 x 13)) x 3 = 10647 bounding boxes. However, we are interested in one object only. Hence to reduce the detections from 10647 to 1, thresholding and NMS is carried out as a post processing step. Boxes whose objectness core is below some threshold are first discarded. Next, Non-maximum Suppression (NMS) which uses t "Intersection over Union", or IoU. Function is employed to get rid of the problem of multiple detections of the same image. The "Intersection over Union" function is depicted in Fig. 2 The steps to implement non-max suppression, are 1. The box with the highest objectness score is selected. 2. Identifying those overlapping boxes and remove those boxes that overlap it more than a IOU threshold. 3. Steps 1 and 2 are repeated till all the overlapping boxes whose score s less than the current selected box.

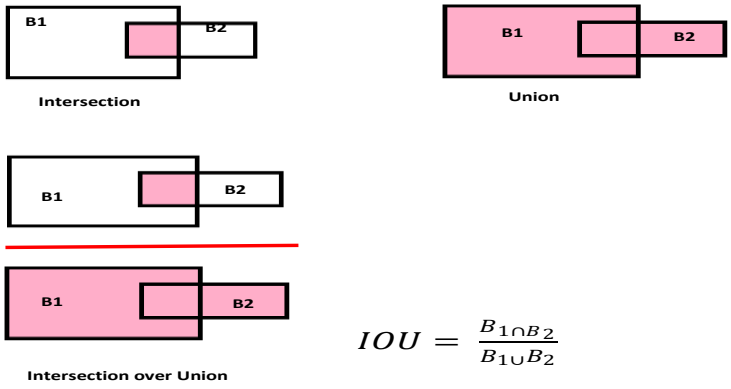

Fig.2. Intersection over Union Function

Fig.3. and Fig.4. Shows the object detection outputs of YOLO V3 on COCO and Pascal VOC dataset.

Fig. 3. Object detection using YOLO V3: COCO dataset

Fig. 4. Object detection using YOLO V3: PASCAL VOC dataset

In this work, 20 different objects were considered for detection and Table 1 gives the prediction probability of YOLV3 on COCO and Pascal VOC datasets. A graphical comparison is depicted in Fig. 5 which indicates that, overall YOLO V3 is very good at object detection both when COCCO and Pascal VOC dataset is used. However, the results are somewhat better with Pascal VOC dataset.

Table 1 Test analysis of coco and pascal VOC data set

Object detected using deep learning in YOLO V3 Model		
Classes	COCO	Pascal VOC
Airplane	0.9995	0.9975
Bus	0.8852	0.9954
Bowl	0.8106	0.9477
Bed	0.9754	0.9993
Car	0.99	0.9987
Motorcycle	0.95	0.9998
Bicycle	0.9541	0.9895
Person	0.99	0.9999
Tie	0.99	0.7486
Dog	0.9863	0.9798
Train	0.994	0.9993
Truck	0.9976	0.9614
Cat	0.9698	0.9814
Bottle	0.869	0.9863
Dining table	0.7825	0.9821
Bird	0.6757	0.9990
Sheep	0.9945	0.8293
Hourse	0.9922	0.9732
Cow	0.7527	0.623

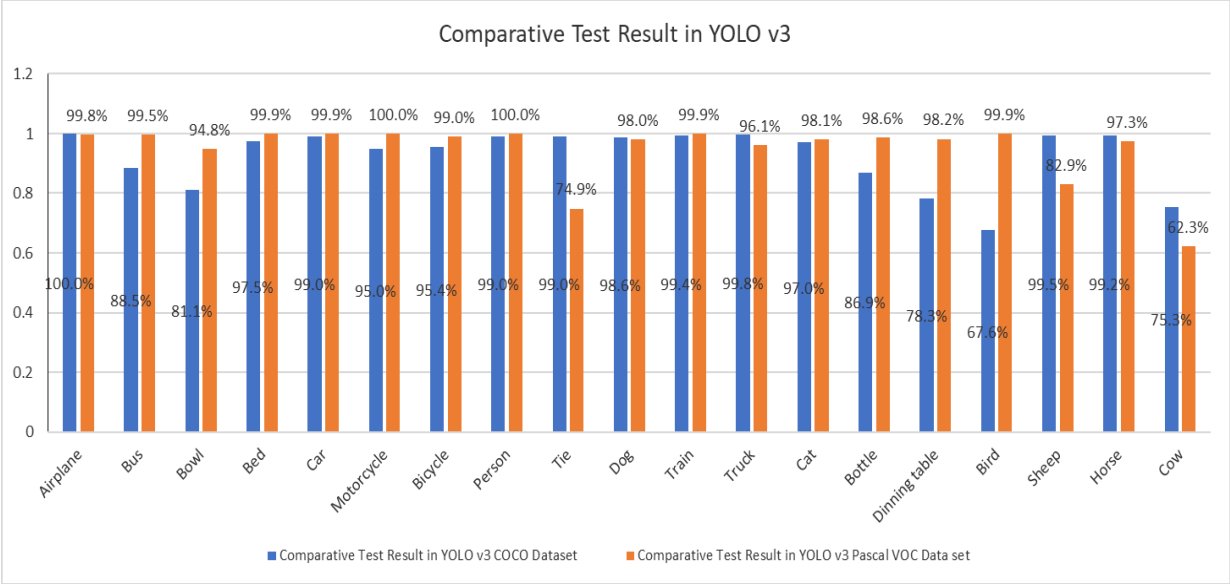

Fig. 5. Comparison of Object Prediction probabilities of YOLO V3 on COCO and PASCAL VOC datasets

4. Conclusion

This paper aims at detecting objects with the YOLO V3 system using a pre-trained darknet 53 model as the backbone network. Preprocessed images are given as input to the detector and the detector outputs bounding boxes along with objectness score and class label. YOLO v3 uses logistic regression to compute the objectness score and it provides the objectness scores for all object classes in each bounding box predicted. As YOLO V3 uses logistic classifier for each class in place of the softmax layer used in YOLO v2, it is capable of providing multi-class classification. COCO and Pascal VOC datasets are used in this study to evaluate the performance of YOLO V3, and the results indicate that YOLO V3 is very good at object detection.

References

[1] C. Wojek, P. Dollar, B. Schiele, and P. Perona, “Pedestrian detection: An evaluation of the state of the

[2] art,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 34, no. 4, p. 743, 2012.

[3] H. Kobatake and Y. Yoshinaga, “Detection of spicules on mammogram based on skeleton analysis.”

[4] IEEE Trans. Med. Imag., vol. 15, no. 3, pp. 235–245, 1996.

[5] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell,

[6] “Caffe: Convolutional architecture for fast feature embedding,” in ACM MM, 2014.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in NIPS, 2012.

[8] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, “Realtime multi-person 2d pose estimation using part affinity fields,” in CVPR, 2017.

[9] Z. Yang and R. Nevatia, “A multi-scale cascade fully convolutional network face detector,” in ICPR, 2016.

[10] C. Chen, A. Seff, A. L. Kornhauser, and J. Xiao, “Deepdriving: Learning affordance for direct

[11] perception in autonomous driving,” in ICCV, 2015

[12] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale

[13] hierarchical image database,” in CVPR, 2009.

[14] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in ICML, 2015.

[15] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, vol. 521, no. 7553, pp. 436–444, 2015.

[16] Y. Li, K. He, J. Sun et al., "R-fcn: Object detection via region-based fully convolutional networks," in NIPS, 2016, pp. 379–387.

[17] T.-Y. Lin, P. Dollar, R. B. Girshick, K. He, B. Hariharan, and S. J. ´ Belongie, "Feature pyramid networks for object detection," in CVPR, 2017.

[18] K. He, G. Gkioxari, P. Dollar, and R. B. Girshick, "Mask r-cnn," in ´ ICCV, 2017

[19] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object

[20] detection and semantic segmentation," in CVPR, 2014.

[21] R. Girshick, "Fast r-cnn," in ICCV, 2015

[22] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards realtime object detection with region

[23] proposal networks," in NIPS, 2015, pp. 91–99.

[24] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov, "Scalable object detection using deep neural

[25] networks," in CVPR, 2014.

[26] D. Yoo, S. Park, J.-Y. Lee, A. S. Paek, and I. So Kweon, "Attentionnet: Aggregating weak directions

[27] for accurate object detection," in CVPR, 2015.

[28] M. Najibi, M. Rastegari, and L. S. Davis, "G-cnn: an iterative grid based object detector," in CVPR, 2016

[29] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in CVPR, 2016.

[30] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in ECCV, 2016.

[31] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," arXiv:1612.08242, 2016.

[32] C. Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, "Dssd: Deconvolutional single shot detector," arXiv:1701.06659, 2017.

[33] Z. Shen, Z. Liu, J. Li, Y. G. Jiang, Y. Chen, and X. Xue, "Dsod: Learning deeply supervised object

[34] detectors from scratch," in ICCV, 2017

[35] A. Stuhlsatz, J. Lippel, and T. Zielke, "Feature extraction with deep neural networks by a generalized

[36] discriminant analysis." IEEE Trans. Neural Netw. & Learning Syst., vol. 23, no. 4, pp. 596–608, 2012.

[37] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," Int. J. of Comput. Vision,

[38] vol. 60, no. 2, pp. 91–110, 2004.

[39] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in CVPR, 2005.

[40] R. Lienhart and J. Maydt, "An extended set of haar-like features for rapid object detection," in ICIP,

[41] 2002.

[42] C. Cortes and V. Vapnik, "Support vector machine," Machine Learning, vol. 20, no. 3, pp. 273–297,

[43] 1995.

[44] Y. Freund and R. E. Schapire, "A desicion-theoretic generalization of on-line learning and an

[45] application to boosting," J. of Comput. & Sys. Sci., vol. 13, no. 5, pp. 663–671, 1997.

[46] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with

[47] discriminatively trained part-based models," IEEE Trans. Pattern Anal. Mach. Intell., vol. 32, pp. 1627–1645, 2010.

[48] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object

[49] classes challenge 2007 (voc 2007) results (2007)," 2008

[50] H. Jiang and E. Learned-Miller, "Face detection with the faster r-cnn," in FG, 2017.

[51] N. Liu, J. Han, D. Zhang, S. Wen, and T. Liu, “Predicting eye fixations using convolutional neural
[52] networks,” in CVPR, 2015.
[53] Z. Yang and R. Nevatia, “A multi-scale cascade fully convolutional network face detector,” in ICPR,
[54] 2016.
[55] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in CVPR, 2016.
[56] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, “Overfeat: Integrated
[57] recognition, localization and detection using convolutional networks,” arXiv:1312.6229, 2013.
[58] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A.
[59] Rabinovich, “Going deeper with convolutions,” in CVPR, 2015.
[60] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick,
[61] “Microsoft coco: Common objects in ´ context,” in ECCV, 2014.
[62] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, “The pascal visual object
[63] classes challenge 2007 (voc 2007) results (2007),” 2008
[64] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman, “The pascal visual object
[65] classes challenge 2012 (voc2012) results (2012),” in http://www.pascal- network.org/ challenges/VOC/
[66] voc2011 / workshop/index.html,2011