



**HAL**  
open science

# LapNet: Automatic balanced loss and optimal assignment for real-time dense object detection

Florian Chabot, Quoc-Cuong Pham, Mohamed Chaouch

► **To cite this version:**

Florian Chabot, Quoc-Cuong Pham, Mohamed Chaouch. LapNet: Automatic balanced loss and optimal assignment for real-time dense object detection. 2019, 10.48550/arXiv.1911.01149 . cea-04324810

**HAL Id: cea-04324810**

**<https://cea.hal.science/cea-04324810>**

Submitted on 5 Dec 2023

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# LapNet : Automatic Balanced Loss and Optimal Assignment for Real-Time Dense Object Detection

Florian Chabot, Quoc-Cuong Pham, and Mohamed Chaouch

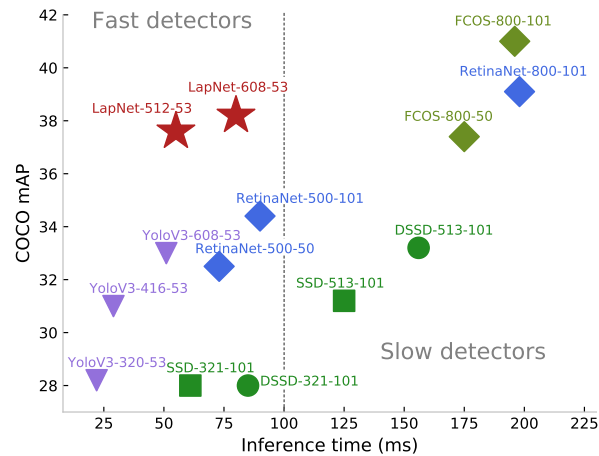
CEA, LIST, Vision and Learning Lab for Scene Analysis, France  
florian.chabot@cea.fr

**Abstract.** Real-time single-stage object detectors based on deep learning [18,19,20,21] still remain less accurate than more complex ones [15,27]. The trade-off between model performance and computational speed is a major challenge. In this paper, we propose a new way to efficiently learn a single-shot detector which offers a very good compromise between these two objectives (Figure 1). To this end, we introduce LapNet, an anchor based detector, trained end-to-end without any sampling strategy. Our approach aims to overcome two important problems encountered in training an anchor based detector: (1) ambiguity in the assignment of anchor to ground truth and (2) class and object size imbalance. To address the first limitation, we propose a soft positive/negative anchor assignment procedure based on a new overlapping function called Per-Object Normalized Overlap (PONO). This soft assignment can be self-corrected by the network itself to avoid ambiguity between close objects. To cope with the second limitation, we propose to learn additional weights, that are not used at inference, to efficiently manage sample imbalance. These two contributions make the detector learning more generic whatever the training dataset. Various experiments show the effectiveness of the proposed approach.

**Keywords:** Real Time Object Detection, Sample Imbalance

## 1 Introduction

Object detection has been widely studied and it still remains a research field of interest for scene understanding applications. It needs to face several challenges: small object detection, heavily occluded objects and real time processing. Modern detectors are based on deep convolutional neural networks (CNN) which have shown their effectiveness in several computer vision tasks (classification, segmentation, detection...). These object detectors can be divided into two main categories: two-stage and single-stage detectors. Two-stage detectors such as [10,22] first propose a set of regions of interest in which deep feature maps are cropped and classified. Although these methods achieve good performance, they are time consuming because of their cascaded pipeline. In contrast, single-stage detectors [19,20,18,15,27] directly predict detection boxes without any region



**Fig. 1.** Speed/accuracy trade-off for single-shot detectors on COCO *test-dev* 2017 using TITAN X GPU. We only plot detectors with a computing time less than 200 ms. Fast detectors correspond to approaches with a processing time per image less than 100 ms. Methods are denoted by "name-resolution-backbone" (50: ResNet-50, 101: ResNet-101, 53: DarkNet-53). LapNet gives significantly better results than YoloV3 [21], using the same backbone network while keeping the same computing time. With an input image resolution of 608x608, LapNet has similar accuracy than RetinaNet [15] but a smaller inference time.

extraction and refinement step. Some of them [19,20,21,18] are fast and thus suitable for real-time detection, at the cost of a lower accuracy.

Recent object detection algorithms generally use a set of discrete pre-defined boxes called anchors. The network predicts if an anchor contains an object (positive anchor) or not (negative anchor) as well as offsets applied to anchors to fit the objects. The strategy for finding positive and negative anchors during training is based on the absolute overlap criterion using Intersection over Union function (IoU): an anchor is labeled as positive if a ground truth box overlaps it above a fixed threshold. Thresholding the overlap is a way to define what an object is and what it is not. However, this strategy could be discussed. For instance, due to the discrete property of the anchor set, there is some chance that no anchor overlaps a ground truth box above the threshold, especially for small objects. In addition, an anchor highly overlapping two objects is assigned to only one of them. This ambiguous case perturbs the model training.

Another general issue when training object detectors is class imbalance and object size imbalance. Best represented classes (large number of training samples) and large objects are generally best trained. Positive anchors are more easily associated to large objects and dominate the loss function. A standard way to address imbalance is positive/negative sampling strategy [22,18,8] based on heuristics. Another way to solve it is to design a specific loss function such as the focal loss [15] which explicitly manages the balance. The objective of the

focal loss is to minimize the influence of easy samples (*i.e.* background samples) in the total loss. It balances foreground and background samples in an automatic hard data mining way, but it does not provide a global weighting function which explicitly takes into account class representativeness and object sizes.

In this paper, we introduce LapNet, a new single-stage object detector combining real time processing and efficient detection. LapNet is based on a CNN encoder-decoder architecture which takes as input an image and a set of predefined anchors. It outputs a dense set of bounding boxes with associated per-class probabilities. In this work, we present two main contributions.

First, we propose a new method to choose positive/negative anchors. As stated above, the assignment based on standard absolute overlap (AO) criterion greatly depends of the number of anchors. To overcome this limitation, we introduce Per-Object Normalized Overlap (PONO), presented in section 3.2. PONO is the overlap between an anchor and a given ground truth box, normalized by the maximum overlap between this ground truth and all its assigned anchors. In other words, each ground truth box has at least one anchor which has a PONO value equal to one. It has two advantages compared to the absolute overlap criterion. The first advantage is an increased robustness when applying a threshold to select positive and negative anchors. When a ground truth box has too small AO values, it is discarded. PONO solves this issue by design. The second advantage is that PONO normalizes the overlap score which makes its more insensitive to the anchor discretization. LapNet takes its name from this idea. In addition, a strategy allowing the network to self-correct the positive and negative anchor assignment during training is proposed. The intuition is that a positive anchor could highly overlap almost equally several objects. When such an ambiguity happens, the network will not be able to fit the right object. To solve this issue, we propose to consider an ambiguous anchor as negative using an online Ambiguity Management Strategy (AMS) described in section 3.3.

The second contribution is related to the loss function. Dense optimization for object detection is still difficult because of large imbalance of classes and object sizes. The authors of [13] proposed the homoscedastic loss for automatic loss balancing in the context of multi-task learning. In our work, we generalize this concept to address general sample imbalance by automatically learn weights which balance classes and object sizes (explained in section 3.3). It is a generic alternative to multi-scale detection used in recent litterature [16,15,27,21]. These methods use Feature Pyramid Network (FPN [16]), which consists in predicting boxes for several feature levels (multi-scale prediction). During training, FPN based approaches balance object sizes in the network architecture itself: large objects are optimized in the lower pyramid level and small objects in the higher pyramid level. It forces to define which anchors correspond to each level. Differently, LapNet’s generic weight learning avoids the use of multi-scale detection.

The two contributions of this work lead LapNet to be comparable to best performing object detectors with a significant reduction of the processing time. This is detailed in section 4.

## 2 Related work

Given an input image, an object detector returns a list of bounding boxes and associated confidence score for each class. An object detector classifies image regions to predict those containing an object. A standard way to analyse all regions in the image is the sliding window scheme which has been widely used by reference work [29,7,4]. With the advent of Deep Learning in computer vision community, sliding window and hand-crafted features for object detection have been surpassed by CNN based methods.

**Anchor based detectors.** Most CNN based methods [22,18,20,15] use anchors to detect objects. Anchors are pre-defined boxes that are supposed to be representative of all objects in terms of scale and ratio. Each 2D position in the CNN output map corresponds to one anchor at this position. The objective is to predict if an anchor contains an object or not, as well as the offsets to apply to the anchor to fit the object. During training, positive and negative labels are assigned to anchors by using the absolute overlap criterion. Classification loss (cross entropy, focal loss) and regression loss (smoothL1) functions are used to train the model. One limitation of these methods is related to the positive/negative anchor assignment. Because of the absolute overlap criterion, some ground truth boxes can be left out. In this paper, we propose to solve this issue using a soft overlap criterion and an Ambiguity Management Strategy allowing to be less sensitivity to the number of pre-defined anchors.

**Two-stage vs single-stage detectors.** The first two-stage based detector was proposed by the authors of [11]. They introduced the object proposal concept: a multi-scale segmentation algorithm [28] is executed to propose several candidate boxes. These regions are then extracted in the image and forwarded into a CNN that predicts class scores and regression offsets. This work was extended in [10] where deep feature maps are extracted in regions of interest (using ROI Pooling) to be more efficient and faster. In 2015, Faster R-CNN [22] was introduced: it combines both anchor based object proposal generation and region classification in the same deep network. Since, Faster R-CNN has been well studied and improved [16,3,2] making it the reference method for two-stage detection. However, cascaded methods such as Faster R-CNN are time consuming in inference phase. For this reason, single-stage methods were first introduced with the SSD algorithm [18]. The network directly outputs a dense set of boxes and associated per class scores. Since, many single-shot approaches have been developed [19,20,8,15,27]. The main advantage of these methods is shorter processing time: with a reasonable input resolution and a light backbone network, some of these detectors give access to real time processing [18,19,20,21]. However, there still exists an important performance gap between fast single-shot detector and heavier ones such as RetinaNet [15] or the recent FCOS [27]. With LapNet, we provide high mean Average Precision (mAP) while keeping real time processing.

**Sample imbalance.** Class imbalance is a well known issue which is not specific

to object detection. For instance, Dice loss [25] is often used in medical image segmentation to address large class imbalance. In object detection, object scales and class representativeness are two important things to take into account. To not focus on large objects, some approaches use sampling strategy [22,18,8] or bootstrapping [23]. In [15,27], the focal loss automatically balances foreground and background samples by giving more weight to hard samples. In our work, a generic method for both classes and object scales imbalance is proposed. To our knowledge, no method modeling sample balance with trainable parameters exists. Our approach generalizes the homoscedastic loss proposed by [13] in the context on multi-task learning.

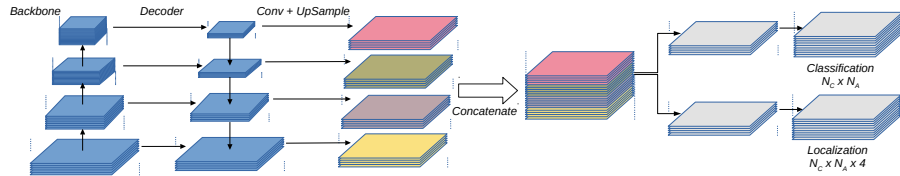
### 3 LapNet

#### 3.1 Overview

Given an input image, LapNet predicts a dense set of bounding boxes for each class and the associated confidence scores. These boxes are then filtered using non-maximum suppression to provide final detections. As in [22,20,16,18], LapNet uses anchors corresponding to pre-defined boxes. More specifically, the network returns for each anchor (1) the probability that it contains an object of a given class and (2) the offsets to apply to fit the object.

**Per-class anchors.** Anchors are computed on the training dataset with the K-means clustering method proposed in [20]. Contrary to related work, anchors are computed for each class independently. In this way, anchors are more specific and better fit the distribution of size and aspect ratio of objects for a given class. This is especially true in several contexts as in traffic scene analysis where the classes "car" and "person" do not share the same box distribution. We define  $N_C$  as the number of classes and  $N_A$  the number of anchors for each class. To simplify,  $N_A$  is the same for each class. With this formulation, LapNet returns  $N_C \times N_A$  score maps and  $N_C \times N_A \times 4$  box offsets (corresponding to the four offsets values  $dx, dy, dw, dh$  to apply to an anchor to fit its associated object as in [10,22,15]).

**Architecture.** LapNet is a fully convolutional network based on an encoder-decoder architecture with skip connections. This architecture allows to keep information of objects at multiple resolutions and contextual information. Unlike FPN based approaches [16,15,21], detection is not performed on each pyramid level. Instead, each pyramid feature map is passed through four convolutional layers and resized to the size of the highest resolution level. These feature maps are then concatenated and forwarded into a classification head and an offset regression head to predict the network outputs. In addition to its simplicity, the advantage of this architecture resides in its genericity compared to multi-scale detection based methods. Indeed, it does not require an explicit assignment between anchors and pyramid levels during training. Figure 2 illustrates the LapNet architecture.



**Fig. 2.** LapNet encoder-decoder architecture overview. The input image is passed through a backbone network. A decoder with skip connections is then used to compute feature maps at several resolutions. Each of these feature maps is then processed by four 3x3 convolutional layers and resized to the size of the highest resolution feature map. After multi-scale feature concatenation, we use two convolutional heads to provide the two outputs. Each head is composed by four 3x3 convolutional layers and one final convolutional layer returning the output.

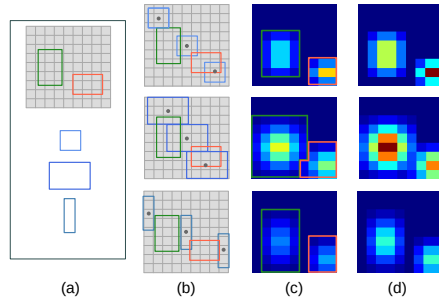
### 3.2 Dense association between anchors and ground truth boxes

LapNet requires to define labels on the entire anchor set. In previous work [22,15,18], anchors with positive or negative label are defined according to a threshold applied to the absolute overlap (AO) criterion: if the overlap value of an anchor with a ground truth box is above the threshold, the anchor is considered as positive. The main drawback of absolute overlap measure is that some ground truth boxes may be missed by the label assignment because of the discrete property of anchor set. This usually happens in the case of small and strongly occluded objects.

To solve this issue, we introduce a new overlap measure called Per-Object Normalized Overlap function (PONO). It is a relative measure that increases the value of AO, especially since the AO value is low. Additionally, it always gives an overlap value equal to one for at least one anchor associated to a given object. Consequently, no object in the ground truth is missed.

**Anchor to ground truth assignment.** A grid of anchors is computed using pre-defined anchors (Figure 3b). It consists in a grid of size  $H_f \times W_f \times N_C \times N_A \times 4$  where  $H_f$  and  $W_f$  are the height and width of the output feature map.  $N_A$  and  $N_C$  are respectively the number of anchors per class and the number of classes. Each element of the anchor grid is assigned to a ground truth box using the AO criterion. This assignment forms clusters where each anchor of the cluster corresponds to the same object in the ground truth (Figure 3c).

**Per-Object Normalized Overlap.** Assignment clusters are then used to compute the PONO map  $O$ . For each cluster, the highest overlap is selected. Each element of the cluster is divided by its corresponding maximum. Let  $A_{c,a,i,j}$  be the anchor of class  $c$ , of index  $a$  at position  $(i, j)$  in the feature map. For a given ground truth box  $B_n$ , we denote  $\mathcal{C}_{B_n}$  the set of anchors associated to  $B_n$ . The PONO function is then defined by:



**Fig. 3.** Per-Object Normalized Overlap (PONO) computation. (a) An input image with two ground truth objects (top) and three pre-defined anchors (bottom). (b) Dense anchor grid representation. (c) Anchor to ground truth assignment using absolute overlap, cluster colors (green and orange) correspond to the ground truth object assigned to each anchor of the grid. (d) Per-Object Normalization Overlap. The two ground truth boxes have at least one anchor with a PONO value equal to one (in red).

$$O(A_{c,a,i,j}, B_n) = \frac{IoU(A_{c,a,i,j}, B_n)}{\max_{A_{c,a',i',j'} \in \mathcal{C}_{B_n}} IoU(A_{c,a',i',j'}, B_n)} \quad \text{with} \quad A_{c,a,i,j} \in \mathcal{C}_{B_n} \quad (1)$$

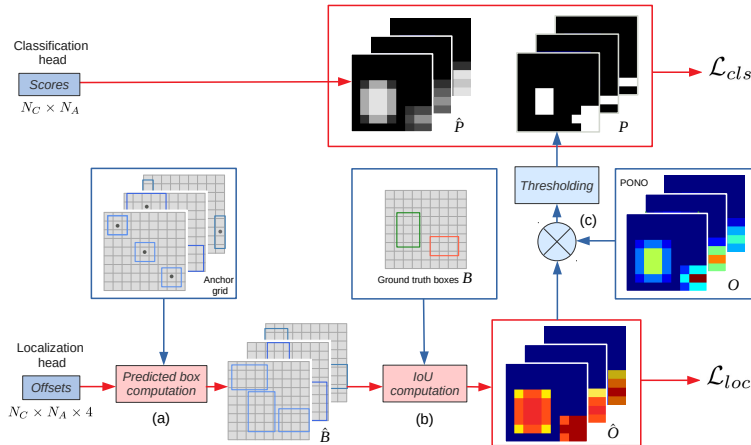
In the following, we consider the resulting PONO map  $O = \{O_{c,a,i,j} = O(A_{c,a,i,j}, B_n)\}$ . By definition,  $O$  has at least one element of value 1 for each object (Figure 3d). It will be used as input in the training process described in the next part.

### 3.3 LapNet training process

To train LapNet, two loss functions are used: a localization loss  $\mathcal{L}_{loc}$ , and a classification loss  $\mathcal{L}_{cls}$ . They use the pre-computed PONO map  $O$  described in the previous section. The localization loss is used to regress offsets applied to an anchor to fit its associated ground truth box. The classification loss is used to predict if an anchor contains an object of a specific class. In this work, we propose a weighted variant of these losses to avoid class and object size imbalance. Manually finding weights or standard hyper-parameters search to balance samples or losses for better training is time consuming. To deal with this issue, we propose an efficient optimization strategy to find these weights automatically in the training loop, without extra cost in computing time. The overall training process is illustrated in Figure 4.

**Pixel-wise localization loss.** In LapNet, the localization loss uses the Intersection over Union function to train the box offset regression task. Contrary to previous anchor based approaches [22,20,19,15] which directly learn to regress





**Fig. 4.** LapNet training process. Given an image, the network outputs scores and offsets. (a) Offsets are applied to the anchor grid to get predicted boxes. (b) IoU map  $\hat{O}$  is then computed using predicted boxes and associated ground truth boxes.  $\hat{O}$  is used to compute the localization loss  $\mathcal{L}_{loc}$ . (c) The thresholded product between PONO map  $O$  and predicted IoU map  $\hat{O}$  determines classification labels to optimize the classification loss  $\mathcal{L}_{cls}$ . Back-propagation is performed on the inverse path of flows colored in red.

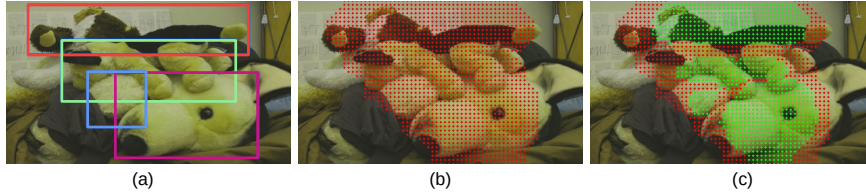
box offsets using SmoothL1 loss, we propose to train the model to predict box offsets in a latent way, by optimizing the IoU function. It is a bounded function that is more stable when predicted offsets have high values. This is particularly true when using PONO because it recovers positive anchors associated to objects that are difficult to detect.

First, by applying predicted offsets to the anchors, predicted boxes are generated (Figure 4a). Then, the IoU  $\hat{O}_{c,a,i,j}$  between a predicted box and its assigned ground truth box is computed.  $\hat{O}$  is the resulting IoU map (Figure 4b). Finally, using the PONO map  $O$ , the localization loss function is defined as follows:

$$\mathcal{L}_{loc}(c, a, i, j) = \begin{cases} \|1 - \hat{O}_{c,a,i,j}\|^2, & \text{if } O_{c,a,i,j} > 0.5 \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

This loss function allows to learn latent offsets so that the overlap between a predicted box and its associated ground truth box is equal to one. The IoU loss has already been used for pixel-to-box regression [27,31] but not for anchor based detector.

**Pixel-wise classification loss with label ambiguity management.** A standard way to define labels for classification loss would be thresholding the PONO map as it is done in the localization loss. Although PONO copes with the problem of unassigned ground truth objects, there still exists the issue of ambiguity between objects that are spatially close in the image. Indeed, an anchor that



**Fig. 5.** Illustration of ambiguity management (best viewed in color). (a) An image with four overlapping ground truth bounding boxes. (b) Selected positive anchors by thresholding PONO values. For readability, anchors are represented by red points. (c) Selected positive anchors (green points) by thresholding PONO map  $O$  weighted by geometric overlap  $\hat{O}$ . We observe that green points form four clusters corresponding to the four objects. Remaining red points correspond to ambiguous anchors (on object borders) labeled as negative anchors.

overlaps almost equally several objects is likely to fit any of these objects. Such an anchor is ambiguous and should not be considered as a positive anchor during training because it is generally unable to fit its assigned object correctly. Figure 5 illustrates the ambiguity situation.

To handle such cases, we propose a strategy for determining anchor labels by using the product of the PONO map  $O$  by the predicted geometric overlap  $\hat{O}$  (Figure 4c). Using this strategy, positive and negative labels are defined as follows:

$$P_{c,a,i,j} = \begin{cases} 1, & \text{if } O_{c,a,i,j} \times \hat{O}_{c,a,i,j} > 0.5 \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

In case of ambiguity, if the network cannot predict offsets correctly for a given anchor, the value of  $\hat{O}_{c,a,i,j}$  will be low and therefore the product will return a low score potentially below the threshold. Such an anchor will then be considered as a negative one. Finally, the pixel-wise classification loss is defined by:

$$\mathcal{L}_{cls}(c, a, i, j) = CE(P_{c,a,i,j}, \hat{P}_{c,a,i,j}) \quad (4)$$

where  $CE$  is the standard binary cross-entropy and  $\hat{P}_{c,a,i,j}$  is the probability that the anchor  $A_{c,a,i,j}$  contains an object of the class  $c$ . The sigmoid activation function is applied on logits to get  $\hat{P}$ .

**Balanced losses.** One important challenge in training multi-class models for object detection or semantic segmentation is class and object scale imbalance. Underrepresented classes are much less well learned than well represented classes. Additionally, large objects dominate the loss function leading to poor performance on small objects. To solve it, several methods use sampling strategies [22,16], multi-scale detection head [16,15,27,21], hand-crafted weights [32,1] or specific loss [15].

In our work, we propose to automatically learn balance weights. Inspired by [13], that addresses loss balancing for multi-task training, we learn two kinds of weights: loss weights and anchor weights. Loss weights are used to balance the different loss functions in the total loss. Our main contribution here, is the anchor weights intended for balancing classes and object sizes. More formally, we propose the following weighted losses:

$$\mathcal{L}_{loc} = \lambda_{loc} \frac{1}{N^+} \sum_c \sum_a \lambda_{loc}^{c,a} \sum_{i,j} \mathcal{L}_{loc}(c, a, i, j) \quad (5)$$

$$\mathcal{L}_{cls} = \lambda_{cls} \frac{1}{N} \sum_c \sum_a \lambda_{cls}^{c,a} \sum_{i,j} \mathcal{L}_{cls}(c, a, i, j) \quad (6)$$

where  $\lambda_{cls}$ ,  $\lambda_{loc}$  are the loss weights and  $\lambda_{cls}^{c,a}$ ,  $\lambda_{loc}^{c,a}$  are the anchor weights, defined for a given anchor  $a$  and a class  $c$ .  $N^+$  is the number of positive anchors for the localization task and  $N$  is the total number of pixels. In this formulation, all  $\lambda$  weights are trainable variables. A regularization loss  $\mathcal{L}_{reg}$  is added to avoid  $\lambda = 0$ :

$$\begin{aligned} \mathcal{L}_{reg} = & \log\left(\frac{1}{\lambda_{cls}}\right) + \log\left(\frac{1}{\lambda_{loc}}\right) \\ & + \frac{1}{N_C N_A} \sum_c \sum_a \log\left(\frac{1}{\lambda_{cls}^{c,a}}\right) + \log\left(\frac{1}{\lambda_{loc}^{c,a}}\right) \end{aligned} \quad (7)$$

In practice, instead of directly optimizing  $\lambda$ , we use  $s = \log(\frac{1}{\lambda})$  as trainable variables for better numerical stability and to avoid weights to be negative. In addition, if an anchor grid does not have a least one positive label, corresponding anchor weights  $\lambda_{cls}^{c,a}$  and  $\lambda_{loc}^{c,a}$  are not updated. This is important to avoid huge values of weights which could make the network diverge. In fact, the objective of anchor weights is to balance the loss of each anchor grid. If an anchor grid has only negative samples, it is an easy case for the network and thus its loss will be very small. In this case, the anchor weight increases too much compared to the other weights of anchor grids containing positive samples. Finally, the total loss for training LapNet is the sum of the localization, classification and regularization losses.

## 4 Experiments

In this section, the effectiveness of LapNet is demonstrated through various experiments. In particular, we provide results on two challenging datasets (PASCAL VOC [6], MSCOCO [17]). Ablation studies are conducted to show the relevance of each contribution.

**Training details.** In all experiments, SGD with momentum of 0.9, initial learning rate of 0.005 and polynomial decay policy with power of 0.9 are used. For data augmentation, random scale and horizontal random flipping are applied.

We arbitrarily fix the number of anchors per object to  $N_A = 10$ . The trainable variables  $s = \log(\lambda)$  are all initialized to 1 (we observed that the initial value does not influence final performances). Two pre-trained backbone networks, DarkNet-53 [21] and Inception-Resnet-V2 [26] are used, both pre-trained on ImageNet [5]. The training is performed on 4 GPUs with a global batch size of 28 for DarkNet-53 and 16 for Inception-Resnet-V2. For DarkNet-53, the decoder structure proposed in YoloV3 [21] is used and for Inception-Resnet-v2, the decoder structure proposed by [15] is plugged into the backbone. In both cases, the spatial output size of the network is eight times smaller than the input image (feat stride = 8).

#### 4.1 PASCAL VOC

This dataset contains 20 classes of objects. LapNet is trained using the union of 2007 and 2012 `trainval`, and tested on 2007 `test`. 120K iterations are runned to get the final result shown in Table 1. We can see that LapNet improves detection results compared to previous work for both backbone networks. In particular, LapNet, trained end-to-end, offers better performance than the best previous method DSSD [8] which uses multiple training stages to get the final detector. In the following, an ablation study based on PASCAL VOC is presented.

**PONO and ambiguity management strategy.** In this part, we evaluate the relevance of Per-Object Normalized Overlap and ambiguity management strategy. The analysis is presented in Table 2a. All models are trained using loss weights, anchor weights and the DarkNet-53 backbone [21]. In the first column of Table 2a, we compare the performance of LapNet models trained with PONO and the standard absolute overlap (AO) label assignement criteria. We observe better detection results are obtained with PONO than with absolute overlap (81.1 vs 80.3), which demonstrates that PONO is a better criterion for positive/negative label assignment. The last column of Table 2a provides results when ambiguity management strategy, described in 3.3, is added to choose positive and negative anchors. Combining ambiguity management with AO decreases the mAP (78.4 vs 80.3). This could be explained by the fact that AO already misses some ground truth objects. Ambiguity management further deteriorates the result because it makes the overlap value lower. Combining PONO and ambiguity management gives the best result (81.7 vs 81.1).

**Automatic balancing.** We also study the influence of automatic weight learning for loss and anchor balance. Table 2b summarizes obtained results. All models are trained using the PONO map and the ambiguity management strategy. The first row is a model where weights are constant and equal to one. We observe that the model completely failed as it was expected. The main reason comes from the normalizer  $N$  in the equation of the classification loss (5) which is an important parameter. As a matter of fact,  $N$  is simply the number of pixels (anchors) in the outputs. In other words, we just take the average over all pixels without any

method	backbone	mAP	areo	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Faster[22]	VGG-16	73.2	76.5	79.0	70.9	65.5	52.1	83.1	84.7	86.4	52.0	81.9	65.7	84.8	84.6	77.5	76.7	38.8	73.6	73.9	83.0	72.6
Faster[22]	ResNet-101	76.4	79.8	80.7	76.2	68.3	55.9	85.1	85.3	89.8	56.7	87.8	69.4	88.3	88.9	80.9	78.4	41.7	78.6	79.8	85.3	72.0
MR-CNN[9]	VGG-16	78.2	80.3	84.1	78.5	70.8	68.5	88.0	85.9	87.8	60.3	85.2	73.7	87.2	86.5	85.0	76.4	48.5	76.3	75.5	85.0	81.0
R-FCN[3]	ResNet-101	80.5	79.9	87.2	81.5	72.0	<b>69.8</b>	86.8	88.5	<b>89.8</b>	<b>67.0</b>	88.1	74.5	89.8	90.6	79.9	81.2	53.7	81.8	81.5	85.9	79.9
YOLOv2[20]	DarkNet-19	78.6	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
SSD300[18]	VGG-16	77.5	79.5	83.9	76.0	69.6	50.5	87.0	85.7	88.1	60.3	81.5	77.0	86.1	87.5	83.97	79.4	52.3	77.9	79.5	87.6	76.8
SSD512[18]	VGG-16	79.5	84.8	85.1	81.5	73.0	57.8	87.8	88.3	87.4	63.5	85.4	73.2	86.2	86.7	83.9	82.5	55.6	81.7	79.0	86.6	80.0
SSD513[8]	ResNet-101	80.6	84.3	87.6	82.6	71.6	59.0	88.2	88.1	89.3	64.4	85.6	76.2	88.5	88.9	87.5	83.0	53.6	83.9	82.2	87.2	81.3
DSSD513[8]	ResNet-101	81.5	86.6	86.2	82.6	74.9	62.5	89.0	88.7	88.8	65.2	87.0	78.7	88.2	89.0	87.5	83.7	51.1	86.3	81.6	85.7	83.7
LapNet512	DarkNet-53	81.7	88.1	88.7	82.0	75.0	65.8	88.1	<b>91.7</b>	90.0	65.1	85.9	77.4	88.5	90.8	86.1	<b>86.2</b>	51.5	82.7	81.8	89.2	79.4
LapNet512	IncResV2	<b>83.2</b>	<b>89.8</b>	<b>89.8</b>	<b>83.8</b>	<b>76.1</b>	65.2	<b>89.8</b>	90.7	92.0	64.6	<b>89.8</b>	<b>79.0</b>	<b>91.8</b>	<b>91.8</b>	<b>89.5</b>	84.9	<b>53.5</b>	<b>86.3</b>	<b>82.4</b>	<b>89.6</b>	<b>84.7</b>

**Table 1.** Results on the VOC 2007 test set. Two-stage detectors are on the top of the table and single-stage at the end. Compared to both detector families, LapNet gives the best results.

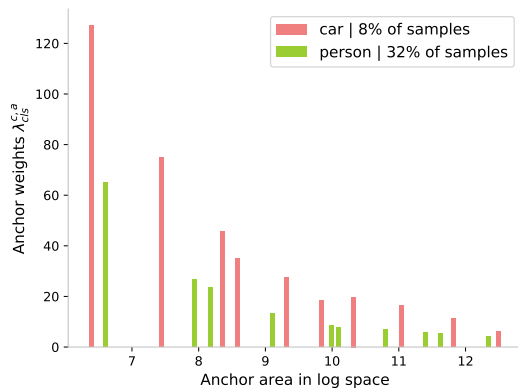
			$\lambda_{loc}$	$\lambda_{cls}$	$\lambda_{loc}^{c,a}$	$\lambda_{cls}^{c,a}$	Loss	mAP
AMS	no	yes	1	1	1	1	CE	16.4
AO	80.3	78.4	1	$R$	1	1	CE	79.0
PONO	81.1	<b>81.7</b>	1	$R$	1	1	FL	71.4
			learned	learned	1	1	CE	81.3
			learned	learned	learned	learned	CE	<b>81.7</b>

(a)

(b)

**Table 2.** (a) Influence of PONO and ambiguity management strategy (AMS). AO is the standard absolute overlap. The bottom right result corresponds to the complete LapNet model which outperforms other kind of positive/negative assignment. (b) Influence of automatic weights learning in the loss equation (5) and (6). We provide results for different weighing strategies (hand-crafted fixed weights or learned weights).  $R = \frac{N}{N^+}$  represents the normalization used for classification loss in RetinaNet [15], see text for detail. CE is the cross entropy and FL the focal loss. The second line is considered as the baseline. The last line corresponds to the proposed full automatic weighting strategy.

focus on positive anchors unlike RetinaNet [15]. The resulting classification loss value is therefore very low because of the large number of easy negative samples leading the network to not be trained correctly. In the second row, weights are also constant but we use the normalization method used in [15] where the classification loss is divided by the number of positive anchors  $N^+$ . With this formulation and equation (5), we give the constant value  $\lambda_{cls} = R = \frac{N}{N^+}$ . This experiment is the baseline for our study. In the third line, we repeat the same experiment using focal loss [15] (with default parameters  $\alpha = 0.25$  and  $\gamma = 2$ ) instead of cross entropy. Surprisingly, the focal loss decreases results significantly. It may be necessary to tune focal loss hyper-parameters to achieve better performance but it is a time consuming process which is specific to each training dataset. The fourth line of Table 2b shows that when using only automatic loss weights the model outperforms the baseline model of the second line. That proves that during training, even if classification loss values are low because of dividing by  $N$ , automatic loss weights naturally counterbalance that phenomenon. Finally the best result is obtained (Table 2b, last line) when using our proposed full automatic loss and anchor balancing. Figure 6 plots the values of some of the learned anchor weights. This plot shows that the larger is the area of an



**Fig. 6.** Analysis of weights automatically learned on VOC2007-2012 `trainval`. Anchor weights of two classes (car and person) are represented ( $N_A = 10$ ). The proportion of samples over the entire training dataset is reported in the top-right corner. This plot shows that weight values decrease with the anchor size and decrease with the number of samples of the class.

anchor, the smaller is its learned weight. Besides, the more the class is under-represented, the larger are the weights. This is the expected behavior which would be hardly designed manually. Thanks to its automatic balancing method, LapNet provides state-of-the-art results on PASCAL VOC.

## 4.2 MSCOCO

We train LapNet on the 80 object categories using the 2017 `trainval` split (115K images) and test on 2017 `test-dev` (20K images) by uploading results on the evaluation server. Results presented in Table 3 are obtained using 500K training iterations. The performance of LapNet is compared to several other state-of-the-art object detectors among which fast single-shot detectors (*i.e* running in less than 100 ms) [21,20,18], slower single-shot detectors [15,27,8,14] and two-stage detectors [24,16,12].

The main result is that LapNet widely outperforms the best of fast detectors, YoloV3 [21], using the same backbone network and the same decoder (37.6 of mAP vs 33.0). This important improvement comes from our proposed learning strategy showing that with almost the same computing time (Figure 1), LapNet gives better detection results.

Compared to stronger but more time consuming methods such as RetinaNet [15] and FCOS [27] taking input images of 800 pixels for the smaller dimension (denoted RetinaNet-800 and FCOS-800), and using ResNet-101-FPN backbone, LapNet is less accurate (-0.9 to -2.8 in mAP) but is 2.5x faster (Figure 1). In previous work, the best results of RetinaNet and FCOS were obtained with the heavier ResNeXt-101 [30] backbone, outperforming LapNet by 2.6 and 3.9 points in mAP respectively. They are mentioned in Table 3 for information. However,

	method	backbone	$AP$	$AP_{50}$	$AP_{75}$	$AP_S$	$AP_M$	$AP_L$
Two-stage	Faster R-CNN w/ TDM [24]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	52.1
	Faster R-CNN w/ FPN [16]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
	Faster R-CNN by G-RMI [12]	Inception-ResNet-v2	34.7	55.5	36.7	13.5	38.1	52.0
	Faster R-CNN w/ TDM [24]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	52.1
Slow Single-stage	SSD-513 [18]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
	DSSD-513 [8]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
	RetinaNet-800 [15]	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
	RetinaNet-800 [15]	ResNeXt-32x8d-101-FPN	40.8	61.1	44.1	24.1	44.2	51.2
	CornerNet [14]	Hourglass-104	40.5	56.5	43.1	19.4	42.7	<b>53.9</b>
	FCOS-800 [27]	ResNet-101-FPN	41.0	60.7	44.1	24.0	44.1	51.0
	FCOS-800 [27]	ResNeXt-32x8d-101-FPN	<b>42.1</b>	<b>62.1</b>	<b>45.2</b>	<b>25.6</b>	<b>44.9</b>	52.0
Fast Single-stage	SSD-321 [18]	ResNet-101-SSD	28.0	45.4	29.3	6.2	28.3	49.3
	DSSD-321 [8]	ResNet-101-DSSD	28.0	46.1	29.2	7.4	28.1	47.6
	RetinaNet-500 [15]	ResNet-50-FPN	32.5	50.9	34.8	13.9	35.8	46.7
	RetinaNet-500 [15]	ResNet-101-FPN	34.4	53.1	36.8	14.7	38.5	49.1
	YoloV2-544 [20]	DarkNet-19	21.6	44.0	19.2	5.0	22.4	35.5
	YoloV3-608 [21]	DarkNet-53	33.0	<b>57.9</b>	34.4	18.3	35.4	41.9
	Lapnet-512	DarkNet-53	37.6	55.5	40.4	17.6	40.5	<b>49.9</b>
	Lapnet-608	DarkNet-53	<b>38.2</b>	56.6	<b>41.2</b>	<b>20.3</b>	<b>41.6</b>	47.5

**Table 3.** LapNet results on MSCOCO `test-dev2017` with two different input resolutions (512x512 and 608x608). They are compared to results from several previous methods. LapNet clearly outperforms previous fast single-stage detectors. Figure 1 presents these results in the form of speed/accuracy graph.

we do not consider them in the speed/accuracy trade-off analysis because they are not suitable for real-time applications. For the same reason, we did not implement LapNet with ResNeXt-101 backbone. ResNet-101 was not implemented either because it was shown in [21] that DarkNet-53 has equivalent performance.

Input image resolution is also an important parameter. LapNet is implemented with the same input image resolution as YoloV3 (512x512 or 608x608) to be fast. Compared to the version of RetinaNet using similar input image resolution (500 pixels for the smallest dimension) denoted RetinaNet-500, LapNet clearly achieves better performance (+3.2 in mAP at least, see Figure 1 and Table 3).

## 5 Conclusion

We have introduced LapNet, a real time single-shot detector which gives a very good trade-off between speed and accuracy. The training process is based on a new positive/negative label assignment method using Per-Object Normalized Overlap and Ambiguity Management Strategy. An automatic weighting method is also proposed to balance losses, classes and object sizes for efficient learning without any extra cost in computation time. Experiments on two reference datasets demonstrate the relevance of our contributions and position LapNet as the most accurate detector among real-time methods. We believe that the proposed generic automatic balancing could be applied to many other computer vision tasks.

## References

1. Badrinarayanan, V., Kendall, A., Cipolla, R.: Segnet: A deep convolutional encoder-decoder architecture for image segmentation. PAMI (2017)
2. Chabot, F., Chaouch, M., Rabarisoa, J., Teuliere, C., Chateau, T.: Deep manta: A coarse-to-fine many-task network for joint 2d and 3d vehicle analysis from monocular image. CVPR (2017)
3. Dai, J., Li, Y., He, K., Sun, J.: R-fcn: Object detection via region-based fully convolutional networks. NIPS (2016)
4. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. CVPR (2005)
5. Deng, J., Dong, W., Socher, R., jia Li, L., Li, K., Fei-fei, L.: Imagenet: A large-scale hierarchical image database. CVPR (2009)
6. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. IJCV (2010)
7. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part based models. PAMI (2010)
8. Fu, C.Y., Liu, W., Ranga, A., Tyagi, A., Berg, A.C.: Dssd: Deconvolutional single shot detector. arXivpreprint arXiv:1701.06659 (2017)
9. Gidaris, S., Komodakis, N.: Object detection via a multi-region and semantic segmentation-aware cnn model. ICCV (2015)
10. Girshick, R.: Fast r-cnn. ICCV (2015)
11. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. CVPR (2014)
12. Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S., Murphy, K.: Speed/accuracy trade-offs for modern convolutional object detectors. CVPR (2017)
13. Kendall, A., Gal, Y., Cipolla, R.: Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. CVPR (2018)
14. Law, H., Deng, J.: Cornernet: Detecting objects as paired keypoints. ECCV (2018)
15. Lin, T., Goyal, P., Girshick, R., He, K., Dollr, P.: Focal loss for dense object detection. ICCV (2018)
16. Lin, T., Dollár, P., Girshick, R.B., He, K., Hariharan, B., Belongie, S.J.: Feature pyramid networks for object detection. CVPR (2017)
17. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollr, P., Zitnick, C.L.: Microsoft coco: Common objects in context (2014)
18. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. ECCV (2016)
19. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. CVPR (2016)
20. Redmon, J., Farhadi, A.: Yolo9000: Better, faster, stronger. CVPR (2017)
21. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement. arXivpreprint arXiv:1804.02767 (2018)
22. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. NIPS (2015)
23. Shrivastava, A., Mulan, H.: Training region-based object detectors with online hard example mining. CVPR (2016)
24. Shrivastava, A., Sukthankar, R., Malik, J., Gupta, A.: Beyond Skip Connections: Top-Down Modulation for Object Detection. arXivpreprint arXiv:1612.06851 (2016)



25. Sudre, C., Li, W., Vercauteren, T., Ourselin, S., Cardoso, M.: Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. *DLMIA* (2017)
26. Szegedy, C., Ioffe, S., Vanhouck, V., Alemi, A.: Inception-v4, inception-resnet and the impact of residual connections on learning. *ICLR WS* (2016)
27. Tian, Z., Shen, C., Chen, H., He, T.: FCOS: Fully convolutional one-stage object detection. *ICCV* (2019)
28. Uijlings, J., van de Sande, K., Gevers, T., Smeulders, A.: Selective search for object recognition. *IJCV* (2013)
29. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. *CVPR* (2001)
30. Xie, S., Girshick, R., Dollr, P., Tu, Z., He, K.: Aggregated residual transformations for deep neural networks. *CVPR* (2017)
31. Yu, J., Jiang, Y., Cao, Z., Huang, T.: Unitbox: An advanced object detection network. *ACM* (2016)
32. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. *CVPR* (2017)