

Imbalance Problems in Object Detection: A Review

Kemal Oksuz^{ID}, Baris Can Cam^{ID}, Sinan Kalkan^{ID}, and Emre Akbas^{ID}

Abstract—In this paper, we present a comprehensive review of the imbalance problems in object detection. To analyze the problems in a systematic manner, we introduce a problem-based taxonomy. Following this taxonomy, we discuss each problem in depth and present a unifying yet critical perspective on the solutions in the literature. In addition, we identify major open issues regarding the existing imbalance problems as well as imbalance problems that have not been discussed before. Moreover, in order to keep our review up to date, we provide an accompanying webpage which catalogs papers addressing imbalance problems, according to our problem-based taxonomy. Researchers can track newer studies on this webpage available at: <https://github.com/kemaloksuz/ObjectDetectionImbalance>

Index Terms—Object detection, imbalance, class imbalance, scale imbalance, spatial imbalance, objective imbalance

1 INTRODUCTION

OBJECT detection is the simultaneous estimation of categories and locations of object instances in a given image. It is a fundamental problem in computer vision with many important applications in e.g. surveillance [1], [2], autonomous driving [3], [4], medical decision making [5], [6], and many problems in robotics [7], [8], [9], [10], [11], [12].

Since the time when object detection (OD) was cast as a machine learning problem, the first generation OD methods relied on hand-crafted features and linear, max-margin classifiers. The most successful and representative method in this generation was the Deformable Parts Model (DPM) [13]. After the extremely influential work by Krizhevsky et al. in 2012 [14], deep learning (or deep neural networks) has started to dominate various problems in computer vision and OD was no exception. The current generation OD methods are all based on deep learning where both the hand-crafted features and linear classifiers of the first generation methods have been replaced by deep neural networks. This replacement has brought significant improvements in performance: On a widely used OD benchmark dataset (PASCAL VOC), while the DPM [13] achieved 0.34 mean average-precision (mAP), current deep learning based OD models achieve around 0.80 mAP [15].

In the last five years, although the major driving force of progress in OD has been the incorporation of deep neural networks [16], [17], [18], [19], [20], [21], [22], [23], imbalance problems in OD at several levels have also received significant attention [24], [25], [26], [27], [28], [29], [30]. An *imbalance problem* with respect to an input property occurs when the

distribution regarding that property affects the performance. When not addressed, an imbalance problem has adverse effects on the final detection performance. For example, the most commonly known imbalance problem in OD is the foreground-to-background imbalance which manifests itself in the extreme inequality between the number of positive examples versus the number of negatives. In a given image, while there are typically a few positive examples, one can extract millions of negative examples. If not addressed, this imbalance greatly impairs detection accuracy.

In this paper, we review the deep-learning-era object detection literature and identify eight different imbalance problems. We group these problems in a taxonomy with four main types: class imbalance, scale imbalance, spatial imbalance and objective imbalance (Table 1). *Class imbalance* occurs when there is significant inequality among the number of examples pertaining to different classes. While the classical example of this is the foreground-to-background imbalance, there is also imbalance among the foreground (positive) classes. *Scale imbalance* occurs when the objects have various scales and different numbers of examples pertaining to different scales. *Spatial imbalance* refers to a set of factors related to spatial properties of the bounding boxes such as regression penalty, location and IoU. Finally, *objective imbalance* occurs when there are multiple loss functions to minimize, as is often the case in OD (e.g. classification and regression losses).

1.1 Scope and Aim

Imbalance problems in general have a large scope in machine learning, computer vision and pattern recognition. We limit the focus of this paper to imbalance problems in object detection. Since the current state-of-the-art is shaped by deep learning based approaches, the problems and approaches that we discuss in this paper are related to deep object detectors. Although we restrict our attention to object detection in still images, we provide brief discussions on similarities and differences of imbalance problems in other domains. We

- The authors are with the Department of Computer Engineering, Middle East Technical University (METU), Ankara 06800, Turkey. E-mail: {kemal.oksuz, can.cam, skalkan}@metu.edu.tr, emre@ceng.metu.edu.tr.

Manuscript received 30 Aug. 2019; revised 17 Jan. 2020; accepted 11 Mar. 2020. Date of publication 19 Mar. 2020; date of current version 2 Sept. 2021.

(Corresponding author: Kemal Oksuz).

Recommended for acceptance by J. Verbeek.

Digital Object Identifier no. 10.1109/TPAMI.2020.2981890

TABLE 1
Imbalance Problems Reviewed in This Paper

Type	Imbalance Problem	Related Input Property
Class	Foreground-Background Class Imbalance (Section 4.1)	The numbers of input bounding boxes pertaining to different classes
	Foreground-Foreground Class Imbalance (Section 4.2)	
Scale	Object/box-level Scale Imbalance (Section 5.1)	The scales of input and ground-truth bounding boxes Contribution of the feature layer from different abstraction levels of the backbone network (i.e. high and low level)
	Feature-level Imbalance (Section 5.2)	
Spatial	Imbalance in Regression Loss (Section 6.1)	Contribution of the individual examples to the regression loss
	IoU Distribution Imbalance (Section 6.2)	IoU distribution of positive input bounding boxes
	Object Location Imbalance (Section 6.3)	Locations of the objects throughout the image
Objective	Objective Imbalance (Section 7)	Contribution of different tasks (i.e. classification, regression) to the overall loss

We state that an imbalance problem with respect to an input property occurs when the distribution regarding that property affects the performance. The first column shows the major imbalance categories. For each Imbalance problem given in the middle column, the last column shows the associated input property concerning the definition of the imbalance problem.

believe that these discussions would provide insights on future research directions for object detection researchers.

Presenting a comprehensive background for object detection is not among the goals of this paper; however, some background knowledge on object detection is required to make the most out of this paper. For a thorough background on the subject, we refer the readers to the recent, comprehensive object detection surveys [31], [32], [33]. We provide only a brief background on state-of-the-art object detection in Section 2.1.

Our main aim in this paper is to present and discuss imbalance problems in object detection comprehensively. In order to do that,

- 1) We identify and define imbalance problems and propose a taxonomy for studying the problems and their solutions.
- 2) We present a critical literature review for the existing studies with a motivation to unify them in a systematic manner. The general outline of our review includes a definition of the problems, a summary of the main approaches, an in-depth coverage of the specific solutions, and comparative summaries of the solutions.
- 3) We present and discuss open issues at the problem-level and in general.
- 4) We also reserved a section (in the Supplementary Material, which can be found on the Computer Society Digital Library at <http://doi.ieeecomputersociety.org/10.1109/TPAMI.2020.2981890>) for imbalance problems found in domains other than object detection. This section is generated with meticulous examination of methods considering their adaptability to the object detection pipeline.
- 5) Finally, we provide an accompanying webpage¹ as a living repository of papers addressing imbalance problems, organized based on our problem-based taxonomy. This webpage will be continuously updated with new studies.

1.2 Comparison With Previous Reviews

Recent object detection surveys [31], [32], [33] aim to present advances in deep learning based generic object detection. To

this end, these surveys propose a taxonomy for object detection methods, and present a detailed analysis of some cornerstone methods that have had high impact. They also provide discussions on popular datasets and evaluation metrics. From the imbalance point of view, these surveys only consider the class imbalance problem with a limited provision. Additionally, Zou *et al.* [32] provide a review for methods that handle scale imbalance. Unlike these surveys, here we focus on a classification of imbalance problems related to object detection and present a comprehensive review of methods that handle these imbalance problems.

There are also surveys on category specific object detection (e.g. pedestrian detection, vehicle detection, face detection) [34], [35], [36], [37]. Although Zehang Sun *et al.* [34] and Dollar *et al.* [35] cover the methods proposed before the current deep learning era, they are beneficial from the imbalance point of view since they present a comprehensive analysis of feature extraction methods that handle scale imbalance. Zafeiriou *et al.* [36] and Yin *et al.* [38] propose comparative analyses of non-deep and deep methods. Litjens *et al.* [39] discuss applications of various deep neural network based methods *i.e.* classification, detection, segmentation to medical image analysis. They present challenges with their possible solutions which include a limited exploration of the class imbalance problem. These category specific object detector reviews focus on a single class and do not consider the imbalance problems in a comprehensive manner from the generic object detection perspective.

Another set of relevant work includes the studies specifically for imbalance problems in machine learning [40], [41], [42], [43]. These studies are limited to the foreground class imbalance problem in our context (i.e. there is no background class). Generally, they cover dataset-level methods such as undersampling and oversampling, and algorithm-level methods including feature selection, kernel modifications and weighted approaches. We identify three main differences of our work compared to such studies. Firstly, the main scope of such work is the classification problem, which is still relevant for object detection; however, object detection also has a “search” aspect, in addition to the recognition aspect, which brings in the background (i.e. negative) class into the picture.

1. <https://github.com/kemaloksuz/ObjectDetectionImbalance>

Secondly, except Johnson *et al.* [43], they consider machine learning approaches in general without any special focus on deep learning based methods. Finally, and more importantly, these works only consider foreground class imbalance problem, which is only one of eight different imbalance problems that we present and discuss here (Table 1).

1.3 A Guide to Reading This Review

The paper is organized as follows. Section 2 provides a brief background on object detection, and the list of frequently-used terms and notation used throughout the paper. Section 3 presents our taxonomy of imbalance problems. Sections 4-7 then cover each imbalance problem in detail, with a critical review of the proposed solutions and include open issues for each imbalance problem. Each section dedicated to a specific imbalance problem is designed to be self-readable, containing definitions and a review of the proposed methods. Section 8 discusses open issues that are relevant to all imbalance problems. Finally, Section 9 concludes the paper. In order to provide a more general perspective, in the Supplementary Material (available online) we present the solutions addressing imbalance in other but closely related domains. Readers who are familiar with the current state-of-the-art object detection methods can directly jump to Section 3 and use Fig. 1 to navigate both the imbalance problems and the sections dedicated to the different problems according to the taxonomy. For readers who lack a background in state-of-the-art object detection, we recommend starting with Section 2.1, and if this brief background is not sufficient, we refer the reader to the more in-depth reviews mentioned in Section 1.1.

2 BACKGROUND, DEFINITIONS, AND NOTATION

In the following, we first provide a brief background on state-of-the-art object detection methods, and then present the definitions and notations used throughout the paper.

2.1 State-of-the-Art in Object Detection

Today there are two major approaches to object detection: top-down and bottom-up. Although both the top-down and bottom-up approaches were popular prior to the deep learning era, today the majority of the object detection methods follow the top-down approach; the bottom-up methods have been proposed relatively recently. The main difference between the top-down and bottom-up approaches is that, in the top-down approach, holistic object hypotheses (i.e., anchors, regions-of-interests/proposals) are generated and evaluated early in the detection pipeline, whereas in the bottom-up approach, holistic objects emerge by grouping sub-object entities like keypoints or parts, later in the processing pipeline.

Methods following the top-down approach are categorized into two: two-stage and one-stage methods. Two-stage methods [16], [17], [18], [21] aim to decrease the large number of negative examples resulting from the predefined, dense sliding windows, called anchors, to a manageable size by using a proposal mechanism [21], [44], [45] which determines the regions where the objects most likely appear, called Region of Interests (RoIs). These RoIs are further processed by a detection network which outputs the object

detection results in the form of bounding boxes and associated object-category probabilities. Finally, the non-maxima suppression (NMS) method is applied on the object detection results to eliminate duplicate or highly-overlapping results. NMS is a universal post-processing step used by all state-of-the-art object detectors.

One-stage top-down methods, including SSD Variants [19], [46], YOLO variants [15], [20], [47] and RetinaNet [22], are designed to predict the detection results directly from anchors – without any proposal elimination stage – after extracting the features from the input image. We present a typical one-stage object detection pipeline in Fig. 1a. The pipeline starts with feeding the input image to the feature extraction network, which is usually a deep convolutional neural network. A dense set of object hypotheses (called anchors) are produced, which are then sampled and labeled by matching them to ground-truth boxes. Finally, labeled anchors (whose features are obtained from the output of the feature extraction network) are fed to the classification and regression networks for training. In a two-stage method, object proposals (or regions-of-interest) are first generated using anchors by a separate network (hence, the two stages).

On the other hand, bottom-up object detection methods [23], [48], [49] first predict important key-points (e.g. corners, centers, etc.) on objects and then group them to form whole object instances by using a grouping method such as associative embedding [50] and brute force search [49].

2.2 Frequently Used Terms and Notation

Table 2 presents the notation used throughout the paper, and below is a list of frequently used terms.

Feature Extraction Network/Backbone. This is the part of the object detection pipeline from the input image until the detection network.

Classification Network/Classifier. This is the part of the object detection pipeline from the features extracted by the backbone to the classification result, which is indicated by a confidence score.

Regression Network/Regressor. This is the part of the object detection pipeline from the features extracted by the backbone to the regression output, which is indicated by two bounding box coordinates each of which consisting of an x-axis and y-axis values.

Detection Network/Detector. It is the part of the object detection pipeline including both classifier and regressor.

Region Proposal Network (RPN). It is the part of the two stage object detection pipeline from the features extracted by the backbone to the generated proposals, which also have confidence scores and bounding box coordinates.

Bounding Box. A rectangle on the image limiting certain features. Formally, $[x_1, y_1, x_2, y_2]$ determine a bounding box with top-left corner (x_1, y_1) and bottom-right corner (x_2, y_2) satisfying $x_2 > x_1$ and $y_2 > y_1$.

Anchor. The set of pre defined bounding boxes on which the RPN in two stage object detectors and detection network in one stage detectors are applied.

Region of Interest (RoI)/Proposal. The set of bounding boxes generated by a proposal mechanism such as RPN on which the detection network is applied on two state object detectors.

Input Bounding Box. Sampled anchor or RoI the detection network or RPN is trained with.

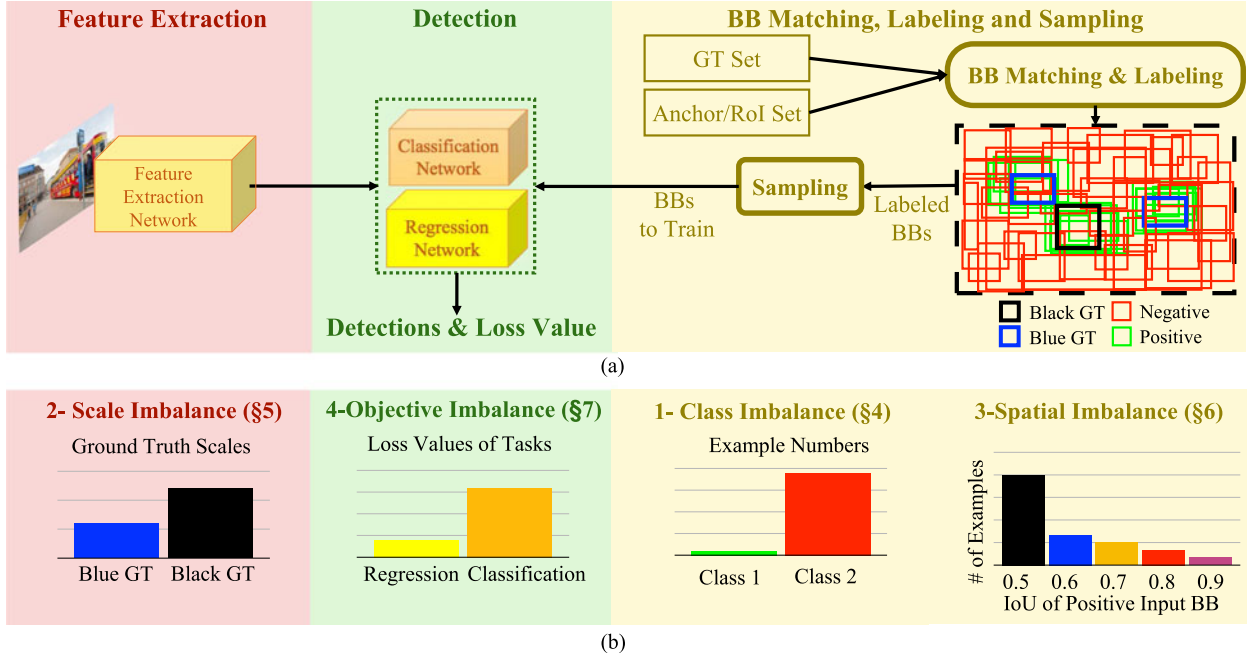


Fig. 1. (a) The common training pipeline of a generic detection network. The pipeline has 3 phases (i.e. feature extraction, detection and BB matching, labeling, and sampling) represented by different background colors. (b) Illustration of an example imbalance problem from each category for object detection through the training pipeline. Background colors specify at which phase an imbalance problem occurs.

Ground Truth. It is tuple (B, u) such that B is the bounding box and u is the *class label* where $u \in C$ and C is the enumeration of the classes in the dataset.

Detection. It is a tuple (\bar{B}, p) such that \bar{B} is the bounding box and p is the vector over the confidence scores for each class² and bounding box.

Intersection Over Union. For a ground truth box B and a detection box \bar{B} , we can formally define Intersection over Union (IoU) [51], [52], denoted by $\text{IoU}(B, \bar{B})$, as

$$\text{IoU}(B, \bar{B}) = \frac{A(B \cap \bar{B})}{A(B \cup \bar{B})}, \quad (1)$$

such that $A(B)$ is the area of a bounding box B .

Under-Represented Class. The class which has less samples in a dataset or mini batch during training in the context of class imbalance.

Over-Represented Class. The class which has more samples in a dataset or mini batch during training in the context of class imbalance.

Backbone Features. The set of features obtained during the application of the backbone network.

Pyramidal Features/Feature Pyramid. The set of features obtained by applying some transformations to the backbone features.

Regression Objective Input. Some methods make predictions in the log domain by applying some transformation which can also differ from method to method (compare transformation in Fast R-CNN [17] and in KL loss [53] for Smooth L1 Loss), while some methods directly predict the bounding box coordinates [23]. For the sake of clarity, we use \hat{x} to denote the regression loss input for any method.

3 A TAXONOMY OF THE IMBALANCE PROBLEMS AND THEIR SOLUTIONS IN OBJECT DETECTION

In Section 1, we defined the problem of imbalance as the occurrence of a distributional bias regarding an input property in the object detection training pipeline. Several different types of such imbalance can be observed at various stages of the common object detection pipeline (Fig. 1). To study these problems in a systematic manner, we propose a taxonomy based on the related input property.

We identify eight different imbalance problems, which we group into four main categories: class imbalance, scale imbalance, spatial imbalance and objective imbalance. Table 1 presents the complete taxonomy along with a brief definition for each problem. In Fig. 2, we present the same taxonomy along with a list of proposed solutions for each problem. Finally, in Fig. 1, we illustrate a generic object detection

TABLE 2
Frequently Used Notations in the Paper

Symbol	Domain	Denotes
B	See. Def.	A Bounding box
C	A set of integers	Set of Class labels in a dataset
$ C_i $	$ C_i \in \mathbb{Z}^+$	Number of examples for the i th class in a dataset
C_i	$i \in \mathbb{Z}^+$	Backbone feature layer at depth i
$\mathbb{I}(P)$	$\mathbb{I}(P) \in \{0, 1\}$	Indicator function. 1 if predicate P is true, else 0
P_i	$i \in \mathbb{Z}^+$	Pyramidal feature layer corresponding to i th backbone feature layer
p_i	$p_i \in [0, 1]$	Confidence score of i th class (i.e. output of classifier)
p_s	$p_s \in [0, 1]$	Confidence score of the ground truth class
p_0	$p_0 \in [0, 1]$	Confidence score of background class
u	$u \in C$	Class label of a ground truth
\hat{x}	$\hat{x} \in \mathbb{R}$	Input of the regression loss

2. We use class and category interchangeably in this paper.

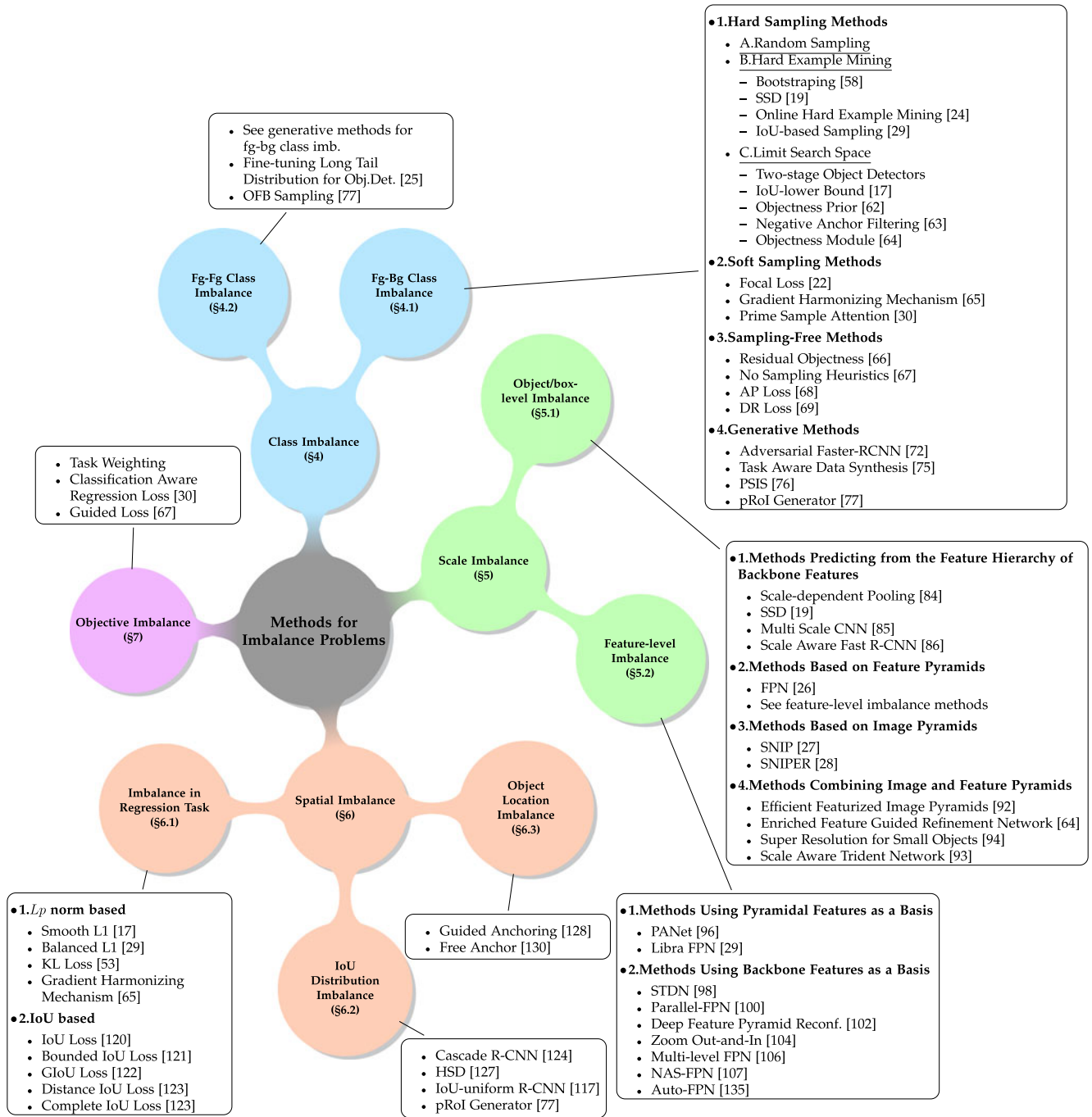


Fig. 2. Problem based categorization of the methods used for imbalance problems. Note that a work may appear at multiple locations if it addresses multiple imbalance problems – e.g. Libra R-CNN [29]

pipeline where each phase is annotated with their typically observed imbalance problems. In the following, we elaborate on the brief definitions provided earlier, and illustrate the typical phases where each imbalance problem occurs.

Class Imbalance (Section 4; blue branch in Fig. 2) occurs when a certain class is over-represented. This problem can manifest itself as either “foreground-background imbalance,” where the background instances significantly outnumber the positive ones; or “foreground-foreground imbalance,” where typically a small fraction of classes dominate the dataset (as observed from the long-tail distribution in Fig. 5). The class imbalance is usually handled at the “sampling” stage in the object detection pipeline (Fig. 1). **Scale imbalance** (Section 5; green branch in Fig. 2) is observed when object instances have various scales and different number of examples pertaining to different scales. This problem is a natural outcome of the fact that objects have different dimensions in nature. Scale also could cause imbalance at the feature-level (typically handled in the “feature extraction” phase in Fig. 1), where contribution from different abstraction layers (i.e. high and low levels) are not balanced. Scale imbalance problem suggests that a single scale of visual processing is not sufficient for detecting objects at different scales. However, as we will see in Section 5, proposed methods fall short in addressing scale imbalance, especially for small objects, even when small objects are surprisingly abundant in a dataset.

Authorized licensed use limited to: UNIVERSIDAD CARLOS III MADRID. Downloaded on August 11, 2024 at 18:22:01 UTC from IEEE Xplore. Restrictions apply.

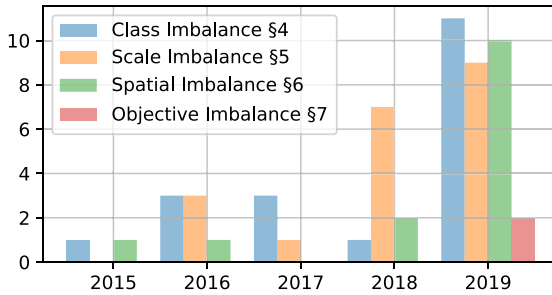


Fig. 3. Number of papers per imbalance problem category through years.

Spatial Imbalance (Section 6; orange branch in Fig. 2) refers to a set of factors related to spatial properties of the bounding boxes. Owing to these spatial properties, we identify three sub-types of spatial imbalance: (i) “imbalance in regression loss” is about the contributions of individual examples to the regression loss and naturally the problem is related to loss function design (ii) “IoU distribution imbalance” is related to the biases in the distribution of IoUs (among ground-truth boxes vs. anchors or detected boxes), and typically observed at the bounding-box matching and labeling phase in the object detection pipeline (Fig. 1) (iii) “object location imbalance” is about the location distribution of object instances in an image, which is related to both the design of the anchors and the sampled subset to train the detection network.

Finally, *objective imbalance* (Section 7; purple branch in Fig. 2) occurs when there are multiple objectives (loss functions) to minimize (each for a specific task, e.g. classification and box-regression). Since different objectives might be incompatible in terms of their ranges as well as their optimum solutions, it is essential to develop a balanced strategy that can find a solution acceptable in terms of all objectives.

Fig. 2 gives an overall picture of the attention that different types of imbalance problems have received from the research community. For example, while there are numerous methods devised for the foreground-background class imbalance problem, the objective imbalance and object location imbalance problems are examples of the problems that received relatively little attention. However, recently there have been a rapidly increasing interest (Fig. 3) in these imbalance problems as well, which necessitates a structured view and perspective on the problems as well as the solutions, as proposed in this paper.

Note that some imbalance problems are directly owing to the data whereas some are the by-products of the specific methods used. For example, class imbalance, object location imbalance etc. are the natural outcomes of the distribution of classes in the real world. On the other hand, objective imbalance, feature-level imbalance and imbalance in regression loss are owing to the selected methods and possibly avoidable with a different set of methods; for example, it is possible to avoid IoU distribution imbalance altogether by following a bottom-up approach where, typically, IoU is not a labeling criterion (e.g. [23], [49]).

4 IMBALANCE 1: CLASS IMBALANCE

Class imbalance is observed when a class is over-represented, having more examples than others in the dataset. This can occur in two different ways from the object detection

perspective: foreground-background imbalance and foreground-foreground imbalance.

Fig. 4 illustrates the presence of class imbalance. To generate the figure, we apply the default set of anchors from RetinaNet [22] on the MS-COCO dataset³ [54] and calculated the frequencies for the cases where the IoU of an anchor with a ground truth bounding box exceeds 0.5 and where it is less than 0.4 (i.e. it is a background box) following the labeling rule of RetinaNet [22]. When an anchor overlaps with a foreground class (with IoU > 0.5), we kept a count for each class separately and normalized the resulting frequencies with the number of images in the dataset.

These two types of class imbalance have different characteristics and have been addressed using different types of solutions. Therefore, in the following, we will cover them separately. However, some solutions (e.g. generative modeling) could be employed for both problem types.

4.1 Foreground-Background Class Imbalance

Definition. In foreground-background class imbalance, the over-represented and under-represented classes are background and foreground classes respectively. This type of problem is inevitable because most bounding boxes are labeled as background (i.e. negative) class by the bounding box matching and labeling module as illustrated in Fig. 4a. Foreground-background imbalance problem occurs during training and it does not depend on the number of examples per class in the dataset since they do not include any annotation on background.

Solutions. We can group the solutions for the foreground-background class imbalance into four: (i) hard sampling methods, (ii) soft sampling methods, (iii) sampling-free methods and (iv) generative methods. Each set of methods are explained in detail in the subsections below.

In sampling methods, the contribution (w_i) of a bounding box (BB_i) to the loss function is adjusted as follows:

$$w_i \text{CE}(p_s), \quad (2)$$

where $\text{CE}()$ is the cross-entropy loss. Hard and soft sampling approaches differ on the possible values of w_i . For the hard sampling approaches, $w_i \in \{0, 1\}$, thus a BB is either selected or discarded. For soft sampling approaches, $w_i \in [0, 1]$, i.e. the contribution of a sample is adjusted with a weight and each BB is somehow included in training.

4.1.1 Hard Sampling Methods

Hard sampling is a commonly-used method for addressing imbalance in object detection. It restricts w_i to be binary; i.e., 0 or 1. In other words, it addresses imbalance by selecting a subset of positive and negative examples (with desired quantities) from a given set of labeled BBs. This selection is performed using heuristic methods and the non-selected examples are ignored for the current iteration. Therefore, each sampled example contributes equally to the loss (i.e. $w_i = 1$) and the non-selected examples ($w_i = 0$) have no contribution

³ Throughout the text, unless otherwise specified, “COCO”, “PASCAL”, and “Open Images” respectively correspond to the Pascal VOC 2012 trainval [51], MS-COCO 2017 train [54], Open Images v5 training (subset with bounding boxes) [55] and Objects365 train [56] dataset partitions. And, when unspecified, the backbone is ResNet-50 [57].

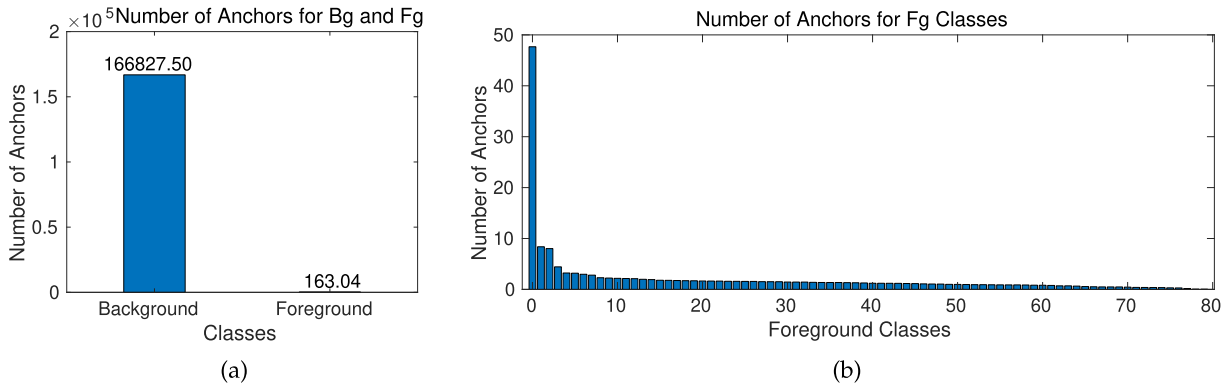


Fig. 4. Illustration of the class imbalance problems. The numbers of RetinaNet [22] anchors on MS-COCO [54] are plotted for foreground-background (a), and foreground classes (b). The values are normalized with the total number of images in the dataset. The figures depict severe imbalance towards some classes.

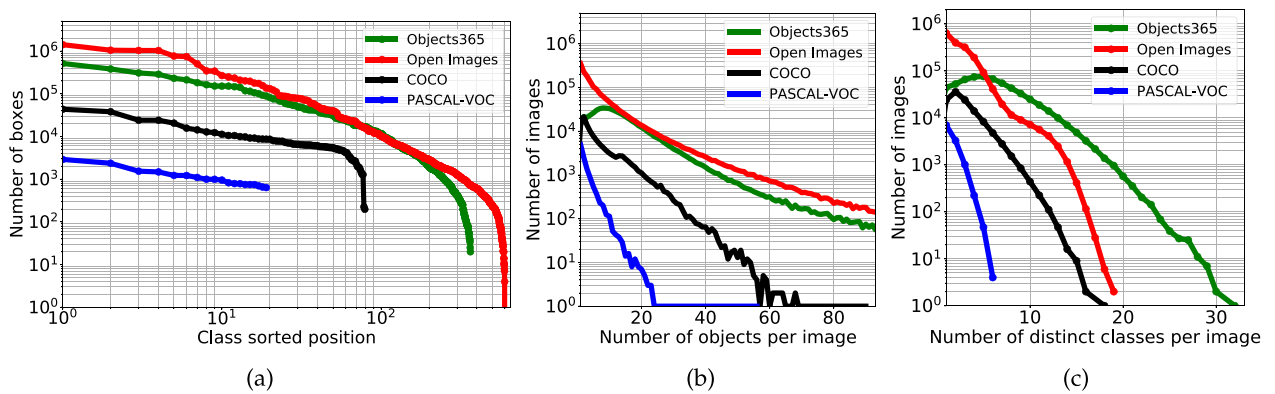


Fig. 5. Some statistics of common datasets (training sets). For readability, the y axes are in logarithmic scale. (a) The total number of examples from each class. (b) The number of images vs. the number of examples. (c) The number of images vs. the number of classes. (This figure is inspired from Kuznetsova *et al.* [55] but calculated and drawn from scratch).

to the training for the current iteration. See Table 3 for a summary of the main approaches.

A straightforward hard-sampling method is *random sampling*. Despite its simplicity, it is employed in R-CNN family of detectors [16], [21] where, for training RPN, 128 positive examples are sampled uniformly at random (out of all positive examples) and 128 negative anchors are sampled in a similar fashion; and 16 positive examples and 48 negative RoIs are sampled uniformly from each image in the batch at random from within their respective sets, for training the detection network [17]. In any case, if the number of positive input bounding boxes is less than the desired values, the mini-batch is padded with randomly sampled negatives. On the other hand, it has been reported that other sampling strategies may perform better when a property of an input box such as its loss value or IoU is taken into account [24], [29], [30].

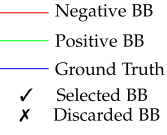






The first set of approaches to consider a property of the sampled examples, rather than random sampling, is the *Hard-example mining methods*⁴. These methods rely on the hypothesis that training a detector more with hard examples (i.e. examples with high losses) leads to better performance. The origins of this hypothesis go back to the *bootstrapping* idea in the early works on face detection [58], [59], [60],

human detection [61] and object detection [13]. The idea is based on training an initial model using a subset of negative examples, then using the negative examples on which the classifier fails (i.e. hard examples), a new classifier is trained. Multiple classifiers are obtained by applying the same procedure iteratively. Currently deep-learning-based methods also adopt some versions of the hard example mining in order to provide more useful examples by using the loss values of the examples. The first deep object detector to use hard examples in the training was Single-Shot Detector [19], which chooses only the negative examples incurring the highest loss values. A more systematic approach considering the loss values of positive and negative samples is proposed in *Online Hard Example Mining* (OHEM) [24]. However, OHEM needs additional memory and causes the training speed to decrease. Considering the efficiency and memory problems of OHEM, *IoU-based sampling* [29] was proposed to associate the hardness of the examples with their IoUs and to use a sampling method again for only negative examples rather than computing the loss function for the entire set. In the IoU-based sampling, the IoU interval for the negative samples is divided into K bins and equal number of negative examples are sampled randomly within each bin to promote the samples with higher IoUs, which are expected to have higher loss values.

To improve mining performance, several studies proposed to *limit the search space* in order to make hard examples

4. In this paper, we adopt the boldface font whenever we introduce an approach involving a set of different methods, and the method names themselves are in *italic*.

TABLE 3
A Toy Example Depicting the Selection Methods of Common Hard and Soft Sampling Methods

	Legend&Method	Considered Property	Intuition	Addit. Params.	Positive Examples		Negative Examples			
Boxes		N/A	N/A	N/A	 $p_s=0.8$ Loss = 0.22 IoU = 0.95	 $p_s=0.2$ Loss = 1.61 IoU = 0.55	 $p_s=0.1$ Loss = 2.30 IoU = 0.35	 $p_s=0.3$ Loss = 1.20 IoU = 0.15	 $p_s=0.6$ Loss = 0.51 IoU = 0.06	 $p_s=1.0$ Loss = 0.00 IoU = 0.00
Hard Sampling	Random Sampling	—	Samples uniformly among boxes	—	Random Sample $\times 1$		Random Sample $\times 2$			
	SSD [19]	Loss	Chooses neg. with max loss	—	Random Sample $\times 1$		✓	✓	✗	✗
	OHEM [24]	Loss	Chooses pos.&neg. with max loss	—	✗	✓	✓	✓	✗	✗
	IoU-Based Sampling [29]	IoU	Samples equal number of negatives from IoU bins	K=2	Does not specify a criterion for positives		Random Sample $\times 1$	Random Sample $\times 1$		
	IoU Lower Bound [17]	IoU=0.05	Discards neg. having IoU less than lower bound	—	Does not specify a criterion for positives		Random Sample $\times 2$			✗
	Objectness Prior [56]	—	Learns to predict objectness priors for pos.	—	Chooses all with prior larger than threshold.		The negatives are sampled randomly by preserving 1:3 ratio with positives. So, fixed batch size is not considered.			
	Negative Anchor Filtering [57]	$p_s = 0.9$	Discards negs. less than threshold	—	Does not specify a criterion for positives		Uses OHEM for the remaining negatives. In this case, the first two boxes are selected			✗
Soft Sampling	Focal Loss [22]	p_s	Promotes examples with larger loss values	$\gamma = 1$	0.2	0.8	0.9	0.7	0.4	0.0
	Gradient Harmonizing Mechanism [59]	p_s	Promotes hard examples by suppressing effects of outliers	$\epsilon = 0.4$	0.66	0.66	0.60	0.66	1.2	0.60
	Prime Sample Attention [30]	IoU, p	Promotes examples with larger IoUs for positives and larger fg scores for negatives (i.e. $1-p_s$ here)	$\gamma = 1$ $\beta = 0$	1.00	0.75	1.00	0.75	0.50	0.25

One positive and two negative examples are to be chosen from six bounding boxes (drawn at top-right). The properties are the basis for the sampling methods. p_s is the predicted ground truth probability (i.e. positive class probability for positive BBs, and background probability for negative BBs). If we set a property or hyper-parameter for this example, it is shown in the table. For soft sampling methods, the numbers are the weights of each box (i.e. w_i). We assumed one foreground class for PISA.

easy to mine. Two-stage object detectors [18], [21] are among these methods since they aim to find the most probable bounding boxes (i.e. RoIs) given anchors, and then choose top N RoIs with the highest objectness scores, to which an additional sampling method is applied. Fast R-CNN [17] sets the lower bound of IoU of the negative RoIs to 0.1 rather than 0 for promoting hard negatives and then applies random sampling. Kong *et al.* [62] proposed a method that learns *objectness priors* in an end-to-end setting in order to have a guidance on where to search for the objects. All of the positive examples having an objectness prior larger than a threshold are used during training, while the negative examples are selected such that the desired balance (i.e. 1 : 3) is preserved between positive and negative classes. Zhang *et al.* [63] proposed determining the confidence scores of anchors with the anchor refinement module in a one-stage detection pipeline and again adopted a threshold to eliminate the easy negative anchors. The authors coined their approach as *negative anchor filtering*. Nie *et al.* [64] used a cascaded-detection scheme in the SSD pipeline which includes an *Objectness Module* before each prediction module. These objectness modules are binary classifiers to filter out the easy anchors.

4.1.2 Soft Sampling Methods

Soft sampling adjusts the contribution (w_i) of each example by its relative importance to the training process. This way,

unlike hard sampling, no sample is discarded and the whole dataset is utilized for updating the parameters. See Table 3 for a summary of the main approaches.

A straightforward approach is to use constant coefficients for both the foreground and background classes. YOLO [20], having less number of anchors compared to other one-stage methods such as SSD [19] and RetinaNet [22], is a straightforward example for soft sampling in which the loss values of the examples from the background class are halved (i.e. $w_i = 0.5$).

Focal Loss [22] is the pioneer example that dynamically assigns more weight to the hard examples as follows:

$$w_i = (1 - p_s)^\gamma, \quad (3)$$

where p_s is the estimated probability for the ground-truth class. Since lower p_s implies a larger error, Equation (3) promotes harder examples. Note that when $\gamma = 0$, focal loss degenerates to vanilla cross entropy loss and Lin *et al.* [22] showed that $\gamma = 2$ ensures a good trade-off between hard and easy examples for their architecture.

Similar to focal loss [22], *Gradient Harmonizing Mechanism* (GHM) [65] suppresses gradients originating from easy positives and negatives. The authors first observed that there are too many samples with small gradient norm, only limited number of samples with medium gradient norm and significantly large amount of samples with large gradient

norm. Unlike focal loss, GHM is a counting-based approach which counts the number of examples with similar gradient norm and penalizes the loss of a sample if there are many samples with similar gradients as follows:

$$w_i = \frac{1}{G(BB_i)/m}, \quad (4)$$

where $G(BB_i)$ is the count of samples whose gradient norm is close to the gradient norm of BB_i ; and m is the number of input bounding boxes in the batch. In this sense, the GHM method implicitly assumes that easy examples are those with too many similar gradients.

Different from other methods, GHM is shown to be useful not only for classification task but also for the regression task. In addition, since the purpose is to balance the gradients within each task, this method is also relevant to the “imbalance in regression loss” discussed in Section 6.1.

Different from the latter soft sampling methods, *PrIme Sample Attention* (PISA) [30] assigns weights to positive and negative examples based on different criteria. While the positive ones with higher IoUs are favored, the negatives with larger foreground classification scores are promoted. More specifically, PISA first ranks the examples for each class based on their desired property (IoU or classification score) and calculates a normalized rank, \bar{r}_i , for each example i as follows:

$$\bar{r}_i = \frac{N_{max} - r_i}{N_{max}}, \quad (5)$$

where r_i ($0 \leq r_i \leq N_j - 1$) is the rank of the i th example and N_{max} is the maximum number of examples over all classes in the batch. Based on the normalized rank, the weight of each example is defined as:

$$w_i = ((1 - \beta)\bar{r}_i + \beta)^\gamma, \quad (6)$$

where β adjusts the contribution of the normalized rank and hence, the minimum sample weight; and γ is the modulating factor again. These parameters are validated for positives and negatives independently. Note that the balancing strategy in Equations (5) and (6) increases the contribution of the samples with high IoUs for positives and high scores for negatives to the loss.

4.1.3 Sampling-Free Methods

Recently, alternative methods emerged in order to avoid the aforementioned hand-crafted sampling heuristics and therefore, decrease the number of hyperparameters during training. For this purpose, Chen *et al.* [66] added an objectness branch to the detection network in order to predict *Residual Objectness* scores. While this new branch tackles the foreground-background imbalance, the classification branch handles only the positive classes. During inference, classification scores are obtained by multiplying classification and objectness branch outputs. The authors showed that such a cascaded pipeline improves the performance. This architecture is trained using vanilla cross entropy loss.

Another recent approach [67] suggests that if the hyperparameters are set appropriately, not only the detector can be trained without any sampling mechanism but also the

performance can be improved. Accordingly, the authors propose methods for setting the initialization bias, loss weighting (see Section 7) and class-adaptive threshold, and in such a way they trained the network using vanilla cross entropy loss achieving better performance.

Finally, an alternative method is to directly model the final performance measure and weigh examples based on this model. This approach is adopted by *AP Loss* [68] which formulates the classification part of the loss as a ranking task (see also *DR Loss* [69] which also uses a ranking method to define a classification loss based on Hinge Loss) and uses average precision (AP) as the loss function for this task. In this method, the confidence scores are first transformed as $x_{ij} = -(p^i - p^j)$ such that p^i is the confidence score of the i th bounding box. Then, based on the transformed values, the primary terms of the AP loss are computed as (presented here in a simplified form):

$$U_{ij} = \frac{\mathbb{I}(x_{ij} > 0)}{1 + \sum_{k \in \mathcal{P} \cup \mathcal{N}} \mathbb{I}(x_{ik} > 0)}, \quad (7)$$

where \mathcal{P}, \mathcal{N} are the set of positive and negative labeled examples respectively. Note that U_{ij} is zero if $p_i < p_j$ and a positive value less than one otherwise. With this quantity, AP loss is defined as follows:

$$L_{AP} = \frac{1}{|\mathcal{P}|} \sum_{i,j} U_{ij} y_{ij}, \quad (8)$$

where y_{ij} is the ranking label, set to 1 only if the i th box is a foreground bounding box and j th box is a background bounding box.

To make this differentiable, the authors propose a novel error-driven update rule, which performs better than the previous works aiming to include average precision as a training objective [70], [71]. However, this optimization algorithm is beyond the scope of our work.

4.1.4 Generative Methods

Unlike sampling-based and sampling-free methods, generative methods address imbalance by directly producing and injecting artificial samples into the training dataset. Table S1 in the Supplementary Material (available online) presents a summary of the main approaches.

One approach is to use generative adversarial networks (GANs). A merit of GANs is that they adapt themselves to generate harder examples during training since the loss values of these networks are directly based on the classification accuracy of the generated examples in the final detection. An example is the *Adversarial-Fast-RCNN* model [72], which generates hard examples with occlusion and various deformations. In this method, the generative manipulation is directly performed at the feature-level, by taking the fixed size feature maps after RoI standardization layers (i.e. RoI pooling [17]). For this purpose, they proposed two networks: (i) adversarial spatial dropout network for occluded feature map generation, and (ii) adversarial spatial transformer network for deformed (transformed) feature map generation. These two networks are placed sequentially in the network design in order to provide harder examples and they are

integrated into the conventional object training pipeline in an end-to-end manner.

Alternatively, artificial images can be produced to augment the dataset [73], [74], [75], [76] by generating composite images in which multiple crops and/or images are blended. A straightforward approach is to randomly place cropped objects onto images as done by Dwibedi *et al.* [73]. However, the produced images may look unrealistic. This problem is alleviated by determining where to paste and the size of the pasted region according to the visual context [74]. In a similar vein, the objects can be swapped between images: *Progressive and Selective Instance-Switching* (PSIS) [76] swaps single objects belonging to the same class between a pair of images considering also the scales and shapes of the candidate instances. Producing images by swapping objects of low-performing classes improves detection quality. For this reason, they use the performance ranking of the classes while determining the objects to swap and the number of images to generate.

A more prominent approach is to use GANs to generate images rather than copying existing objects: an example is *Task Aware Data Synthesis* [75] which uses three competing networks to generate hard examples: a synthesizer, a discriminator and a target network where the synthesizer is expected to fool both the discriminator and the target network by yielding high quality synthetic hard images. Given an image and a foreground object mask, the synthesizer aims to place the foreground object mask onto the image to produce realistic hard examples. The discriminator is adopted in order to enforce the synthesizer towards realistic composite images. The target network is an object detector, initially pretrained to have a baseline performance.

Instead of generating images, the *Positive RoI (pRoI) Generator* [77] generates a set of positive RoIs with given IoU, BB relative spatial and foreground class distributions. The approach relies on a bounding box generator that is able to generate bounding boxes (i.e. positive example) with the desired IoU with a given bounding box (i.e. ground truth). Noting that the IoU of an input BB is related to its hardness [29], the pRoI generator is a basis for simulating, thereby analyzing, hard sampling methods (Section 4.1.1).

4.2 Foreground-Foreground Class Imbalance

Definition. In foreground-foreground class imbalance, the over-represented and the under-represented classes are both foreground classes. This imbalance among the foreground classes has not attracted as much interest as foreground-background imbalance. We discuss this problem in two categories according to their origin: (i) dataset-level, and (ii) batch-level.

4.2.1 Foreground-Foreground Imbalance Owing to the Dataset

Definition. Objects exist at different frequencies in nature, and therefore, there is naturally an imbalance among the object classes in datasets – see Fig. 5a which shows that the datasets suffer from significant gap in class examples. For this reason, overfitting in favor of the over-represented classes may be inevitable for naive approaches on such datasets.

Solutions. Owing to the fact that some of the generative methods are able to generate new images or bounding boxes

(see Section 4.1.4) that cannot be obtained by the conventional training pipeline, these methods can also be adopted to alleviate the foreground-foreground class imbalance problem.

Another method addressing this imbalance problem from the object detection perspective is proposed by Ouyang *et al.* [25]. Their method of *finetuning long-tail distribution for object detection* (here, “long tail” corresponds to under-represented classes) provides an analysis on the effects of this level to the training process and uses clustering based on visual similarity. In their analysis, two factors affecting the training are identified: (i) the accuracy of the prediction, and (ii) the number of examples. Based on this observation, they handcrafted a similarity measure among classes based on the inner product of the features of the last layer of the pretrained backbone network (i.e. GoogLe Net [78]), and grouped the classes hierarchically in order to compensate for dataset-level foreground class imbalance. For each node in the designed hierarchy tree, a classifier is learned based on the confidence scores of the classifier. The leaves of the tree are basically an SVM classifier that determines the final detection for the given input BB.

4.2.2 Foreground-Foreground Imbalance Owing to the Batch

Definition. The distribution of classes in a batch may be uneven, introducing a bias in learning. To illustrate the batch-level foreground imbalance, Fig. 4b provides the mean number of anchors per class on the MS COCO dataset [54]. A random sampling approach is expected to allocate an unbalanced number of positive examples in favor of the one with more anchors, which may lead the model to be biased in favor of the over-represented class during training. Also see Figs 5b and 5c, which display that the numbers of objects and classes in an image vary significantly.

Solutions. A solution for this problem, *Online Foreground Balanced (OFB) sampling* by Oksuz *et al.* [77], shows that the foreground-foreground class imbalance problem can be alleviated at the batch level by assigning probabilities to each bounding box to be sampled, so that the distribution of different classes within a batch is uniform. In other words, the approach aims to promote the classes with lower number of positive examples during sampling. While the method is efficient, the performance improvement is not significant. Moreover, the authors have not provided an analysis on whether or not batch-level imbalance causes a bias in learning, therefore, we discuss this as an open problem in Section 4.4.

4.3 Comparative Summary

In early deep object detectors, hard sampling methods were commonly used to ensure balanced training. OHEM [24] was one of the most effective methods with a relative improvement of 14.7 percent over the Fast R-CNN baseline (with VGG-16 backbone) on MS COCO 2015. On the other hand, recent methods aim to use all examples either by soft sampling or without sampling. Among the seven papers investigated under these categories, six of them have been published during the last year. The pioneering Focal Loss method [22] achieves 9.3 percent relative improvement compared to the baseline alpha-balanced cross entropy on

Retina-Net (i.e. from 31.1 to 34.0 mAP⁵). AP Loss [68], with a relative improvement of 3.9 percent over Focal Loss (i.e. from 33.9 to 35.0 mAP), is also promising.

Sampling methods for positive examples are not many. One approach, prime sample attention (PISA) [30], reported a relative improvement of 4.4 percent (from 36.4 to 38.0 mAP when applied to positives only) and showed that sampling also matters for positives. We notice several crucial points regarding prime sampling. First of all, contrary to the popular belief that hard examples are more preferable over easy examples, PISA shows that, if balanced properly, the positive examples with higher IoUs, which normally incur smaller loss values, are more useful for training compared to OHEM [24] applied to positives. Moreover, the results suggest that the major improvement of the method is on localization since there is no performance improvement in $AP@0.50$, and there is significant improvement for APs with higher IoUs (i.e. up to 2.6 percent in $AP@0.75$). As a result, the improvement can be due to the changing nature of the IoU distribution rather than presenting more descriptive samples to the classifier since the classifier performs worse but the regressor improves (see the discussion on IoU distribution imbalance in Section 6.2).

4.4 Open Issues

As we have highlighted before, class imbalance problem can be analyzed in two main categories: foreground-background class imbalance and foreground-foreground class imbalance. In the following, we identify issues to be addressed with a more focus on foreground-foreground imbalance since it has received less attention.

4.4.1 Sampling More Useful Examples

Open Issue. Many criteria to identify useful examples (e.g. hard example, example with higher IoUs etc.) for both positive and negative classes have been proposed. However, recent studies point out interesting phenomena that need further investigation: (i) For the foreground-background class imbalance, soft sampling methods have become more prominent and optimal weighting of examples needs further investigation. (ii) Hard example mining [24] for the positive examples is challenged by the idea that favors examples with higher IoUs, i.e. the easier ones [30]. (iii) The usefulness of the examples are not considered in terms of foreground-foreground class imbalance.

*Explanation*⁶. In terms of the usefulness of the examples, there are two criteria to be identified: (i) The usefulness of the background examples, and (ii) the usefulness of the foreground examples.

The existing approaches mostly concentrated around the first criterion using different properties (i.e. IoU, loss value, ground truth confidence score) to sample a useful example for the background. However, the debate is still open after Li *et al.* [65] showed that there are large number of outliers during sampling using these properties, which will result in higher loss values and lower confidence scores. Moreover,

the methods preferring a weighting over all the examples [22], [65] rather than discarding a large portion of samples have proven to yield more performance improvement. For this reason, currently, soft sampling approaches that assign weights to the examples are more popular, but which negative examples are more useful needs more investigation.

For the foreground examples, Cao *et al.* [30] apply a specific sampling methodology to the positives based on the IoU, which has proven useful. Note that, this idea conflicts with the hard example mining approach since while OHEM [24] offers to pick the difficult samples, prime samples concentrate on the positive samples with higher IoUs with the ground truth, namely the easier ones. For this reason, currently it seems that the usefulness of the examples is different for the positives and negatives. To sum up, further investigation is required for identifying the best set of examples, or figuring out how to weigh the positive examples during training.

Moreover, sampling methods have proven to be useful for foreground-background class imbalance, however, their effectiveness for the foreground-foreground class imbalance problem needs to be investigated.

4.4.2 Foreground-Foreground Class Imbalance Problem

Open Issue. Foreground-foreground class imbalance has not been addressed as thoroughly as foreground-background class imbalance. For instance, it is still not known why a class performs worse than others; a recent work [76] discredits the correlation between the number of examples in a class and its detection performance. Moreover, despite its differences, the rich literature for addressing imbalance in image classification has not been utilized in object detection.

Explanation. One important finding of a recent study [76] is that a class with the fewest examples in the dataset can yield one of the best detection performances and thus the total number of instances in the dataset is not the only issue to balance the performance of foreground classes. Such discrepancies presents the necessity of an in-depth analysis to identify the root cause and investigate for better sampling mechanisms to employ while balancing a dataset.

Moreover, we identify a similarity and a difference between the class imbalance from image classification perspective and the foreground-foreground class imbalance problem (see the Supplementary Material available online). The similarity is that neither has a background class. On the other hand, the input BBs are labeled and sampled in an online fashion in object detection, which makes the data that the detector is trained with not static.

Class imbalance problem is addressed in image classification from a larger scope by using not only classical over-sampling and under-sampling methods but also (i) transfer learning to transfer the information from over-represented to under-represented classes, (ii) data redundancy methods to be useful for under-sampling the over-represented classes, (iii) weak supervision in order to be used in favor of under-represented class and (iv) a specific loss function for balancing foreground classes. Such approaches can be adopted for addressing foreground-foreground imbalance in object detection as well.

5. Unless otherwise stated, MS COCO 2017 minival results are reported using the COCO-style mAP.

6. Issues that may not be immediately clear include a detailing explanation section



Fig. 6. Illustration of batch-level class imbalance. (a) An example that is consistent with the overall dataset (person class has more instances than parking meter). (b) An example that has a different distribution from the dataset. Images are from the MS COCO dataset.

4.4.3 Foreground-Foreground Imbalance Owing to the Batch

Open Issue. The distribution of the foreground classes in a batch might be very different than that of the overall dataset, especially when the batch size is small. An important question is whether this may have an influence on the overall performance.

Explanation. A specific example is provided in Fig. 6 from the MS COCO dataset [54]: An over-represented class ('person' class in this example) across the dataset may be under-represented in a batch or vice versa. Similar to the foreground-background class imbalance problem, over-representing a class in a batch will increase the probability of the corresponding class to dominate more.

Even when a batch has uniform foreground-foreground class distribution, an imbalance may occur during sampling due to the fact that sampling algorithms either select a subset or apply a weighting to the large number of boxes. If the sampling mechanism tends to choose the samples from specific classes in the image, balancing the dataset (or the batch) may not be sufficient to address the foreground-foreground class imbalance entirely.

4.4.4 Ranking-Based Loss Functions

Open Issue. AP Loss [68] sorts the confidence scores pertaining to all classes together to make a ranking between the detection

boxes. However, this conflicts with the observation that the optimal confidence scores vary from class to class [79].

Explanation. Chen *et al.* [68] use all the confidence scores all together without paying attention to the fact that the optimal threshold per class may vary [79]. Therefore, a method that sorts the confidence scores in a class-specific manner might achieve a better performance. Addressing this issue is an open problem.

5 IMBALANCE 2: SCALE IMBALANCE

We discuss scale the imbalance problem in two parts: the first part, Object/Box-Level Scale Imbalance, analyses the problems arising from the imbalanced distribution of the scales of the objects and input bounding boxes. The second part, Feature Imbalance, analyses the problems arising at the feature extraction level and concerns the methods using pyramidal features.

5.1 Object/Box-Level Scale Imbalance

Definition. Scale imbalance occurs when certain sizes of the objects or input bounding boxes are over-represented in the dataset. It has been shown that this affects the scales of the estimated RoIs and the overall detection performance [27]. Fig. 7 presents the relative width, height and the area of the objects in the MS-COCO dataset [54]; we observe a skewness in the distributions in favor of smaller objects.

Many of the deep object detectors rely on a backbone convolutional neural network (e.g. [57], [78], [80], [81], [82]), pre-trained on an image classification task, in order to extract visual features from the input image. Li *et al.* [83] discuss the cons of employing such networks designed for the classification task and propose a backbone specially designed for the object detection task where they limit the spatial downsampling rate for the high level features. Overall, these networks, also called the backbones, play an important role for the performance of the object detectors, but they alone are insufficient for handling the scale diversity of the input bounding boxes.

Solutions. First examples of deep object detectors made predictions from the final layer of the backbone network (see [16], [17] and Fig. 8a), and therefore, neglected the scale-diversity of BBs. The solutions to addressing scale imbalance can be grouped into four (Fig. 8): methods predicting from the hierarchy of the backbone features (Fig. 8b), methods

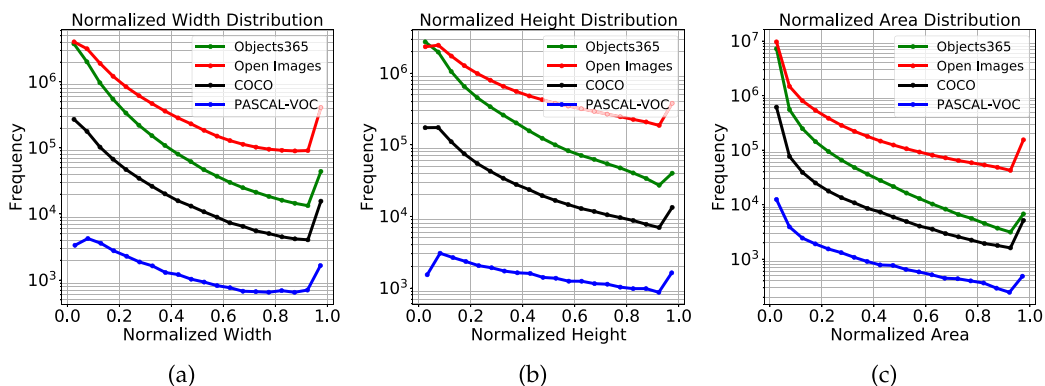


Fig. 7. Imbalance in scales of the objects in common datasets: the distributions of BB width (a), height (b), and area (c). Values are with respect to the normalized image (i.e. relative to the image). For readability, the y axes are in log-scale.

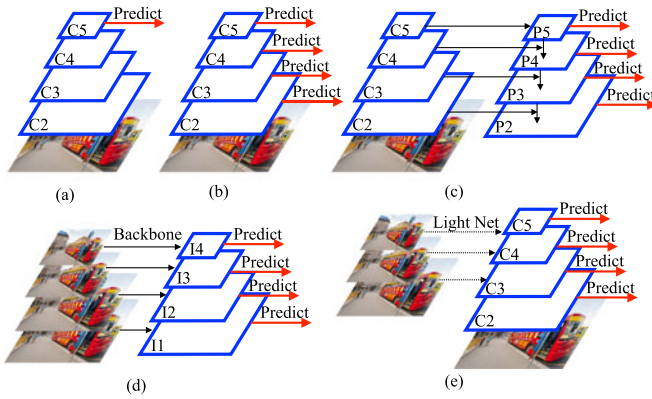


Fig. 8. An illustration and comparison of the solutions for scale imbalance. “Predict” refers to the prediction performed by a detection network. The layered boxes correspond to convolutional layers. (a) No scale balancing method is employed. (b) Prediction is performed from backbone features at different levels (i.e. scales) (e.g. SSD [19]). (c) The intermediate features from different scales are combined before making prediction at multiple scales (e.g. Feature Pyramid Networks [26]). (d) The input image is scaled first and then processed. Each I corresponds to an image pyramidal feature (e.g. Image Pyramids [27]). (e) Image and feature pyramids are combined. Rather than applying backbones, light networks are used in order to extract features from smaller images.

based on feature pyramids (Fig. 8c), methods based on image pyramids (Fig. 8d) and finally methods combining image and feature pyramids (Fig. 8e).

5.1.1 Methods Predicting from the Feature Hierarchy of Backbone Features

These methods make independent predictions from the features at different levels of the backbone network (Fig. 8b). This approach naturally considers object detection at multiple scales since the different levels encode information at different scales; e.g., if the input contains a small object, then earlier levels already contain strong indicators about the small object [84].

An illustrative example for one-stage detectors is the Single Shot Detector (SSD) [19], which makes predictions from features at different layers.

Two-stage detectors can exploit features at different scales while either estimating the regions (in the first stage) or extracting features from these regions (for the second stage). For example, the *Multi Scale CNN* (MSCNN) [85] uses different layers of the backbone network while estimating the regions in the first stage whereas Yang *et al.* [84] choose an appropriate layer to pool based on the scale of the estimated RoI; called *Scale Dependent Pooling* (SDP), the method, e.g., pools features from an earlier layer if the height of the RoI is small. Alternatively, the *Scale Aware Fast R-CNN* [86] learns an ensemble of two classifiers, one for the small scale and one for the large scale objects, and combines their predictions.

5.1.2 Methods Based on Feature Pyramids

Methods based on feature hierarchies use features from different levels independently without integrating low-level and high-level features. However, the abstractness (semantic content) of information varies among different layers, and thus it is not reliable to make predictions directly from a single layer (especially the lower layers) of the backbone network.

To address this issue, the *Feature Pyramid Networks* (FPN) [26] combine the features at different scales before making predictions. FPN exploits an additional top-down pathway along which the features from the higher level are supported by the features from a lower level using lateral connections in order to have a balanced mixed of these features (see Fig. 8c). The top-down pathway involves upsampling to ensure the sizes to be compatible and lateral connections are basically 1×1 convolutions. Similar to feature hierarchies, a RoI pooling step takes the scale of the RoI into account to choose which level to pool from. These improvements allow the predictor network to be applied at all levels which improves the performance especially for small and medium sized objects.

Although FPN was originally proposed for object detection, it quickly became popular and has been used for different (but related) tasks such as shadow detection [87], instance segmentation [88], [89] and panoptic segmentation [90].

Despite its benefits, FPN is known to suffer from a major shortcoming due to the straightforward combination of the features gathered from the backbone network – the *feature imbalance problem*. We discuss this problem in Section 5.2.

5.1.3 Methods Based on Image Pyramids

The idea of using multi-scale image pyramids, presented in Fig. 8d, for the image processing tasks goes back to the early work by Adelson *et al.* [91] and was popular before deep learning. In deep learning, these methods are not utilized as much due to their relatively high computational and memory costs. However, recently, Singh and Davis [27] presented a detailed analysis on the effect of the scale imbalance problem with important conclusions and proposed a method to alleviate the memory constraint for image pyramids. They investigated the conventional approach of training object detectors in smaller scales but testing them in larger scales due to memory constraints, and showed that this inconsistency between test and training time scales has an impact on the performance. In their controlled experiments, upsampling the image by two performed better than reducing the stride by two. Upon their analysis, the authors proposed a novel training method coined as *SNIP* based on image pyramids rather than feature pyramids. They argue that, while training scale-specific detectors by providing the input to the appropriate detector will lose a significant portion of the data, and that using multi-scale training on a single detector will increase the scale imbalance by preserving the variation in the data. Therefore, SNIP trains multiple proposal and detector networks with images at different sizes, however, for each network only the appropriate input bounding box scales are marked as valid, by which it ensures multi-scale training without any loss in the data. Another challenge, the limitation of the GPU memory, is overcome by an image cropping approach. The image cropping approach is made more efficient in a follow-up method, called *SNIPER* [28].

5.1.4 Methods Combining Image and Feature Pyramids

Image pyramid based methods are generally less efficient than feature pyramid based methods in terms of computational time and memory. However, image pyramid based methods are expected to perform better, since feature pyramid based methods are efficient approximations of such

methods. Therefore, to benefit from advantages of both approaches, they can be combined in a single model.

Even though there are different architectures, the generic approach is illustrated in Fig. 8e. For example, *Efficient Featureized Image Pyramids* [92] uses five images at different scales, four of which are provided to the light-weight featureized image pyramid network module (i.e. instead of additional backbone networks) and the original input is fed into the backbone network. This light-weight network consists of 4 consecutive convolutional layers designed specifically for each input. The features gathered from this module are integrated to the backbone network features at appropriate levels according to their sizes, such that the image features are combined with the features extracted from the backbone network using feature attention modules. Furthermore, after attention modules, the gathered features are integrated with the higher levels by means of forward fusion modules before the final predictions are obtained.

A similar method that is also built on SSD and uses downsampled images is *Enriched Feature Guided Refinement Network* [64] in which a single downsampled image is provided as an input to the Multi-Scale Contextual Features (MSCF) Module. The MSCF consists of two consecutive convolutional layers followed by three parallel dilated convolutions, which is also similar to the idea in Trident Networks [93]. The outputs of the dilated convolutions are again combined using 1×1 convolutions. The authors set the downsampled image size to meet the first prediction layer in the regular SSD architecture (e.g. for 320×320 input, the size of the downsampled image is 40×40). While Pang *et al.* [92] only provide experiments with the SSD backbone, Nie *et al.* [64] show that their modules can also be used in the ResNet backbone.

An alternative approach is to generate super-resolution feature maps. Noh *et al.* [94] proposed *super-resolution for small object detection* for two-stage object detectors which lack strong representations of small RoIs after RoI standardization layers. Their architecture adds four additional modules to the baseline detector: (i) Given the original image, target extractor outputs the targets for the discriminator by using dilated convolutions. This network also shares parameters with the backbone network. (ii) Given the features obtained from the backbone network using the original image and a 0.5x downsampled image, the generator network generates super resolution feature maps for small RoIs. (iii) Given the outputs of (i) and (ii), a discriminator is trained in the conventional GAN setting. (iv) Finally, if the RoI is a small object according to a threshold, then the prediction is carried out by the small predictor, or else by the large predictor.

Another approach, *Scale Aware Trident Networks* [93], combines the advantages of the methods based on feature pyramids and image pyramids without using multiple downsampled images but employing only dilated convolutions. The authors use dilated convolutions [95] with dilation rates 1, 2 and 3 in parallel branches in order to generate scale-specific feature maps, making the approach more accurate compared to feature pyramid based methods. In order to ensure that each branch is specialized for a specific scale, an input bounding box is provided to the appropriate branch according to its size. Their analysis on the effect of receptive field size on objects of different scales shows that larger dilation rates are more appropriate for objects with larger scales. In

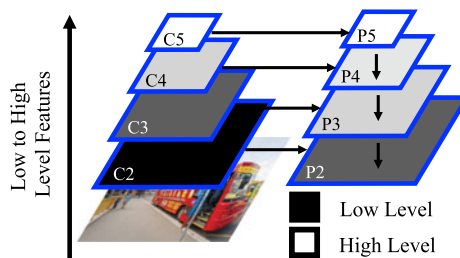


Fig. 9. Feature-level imbalance is illustrated on the FPN architecture.

addition, since using multiple branches is expected to degrade the efficiency due to the increasing number of operations, they proposed a method for approximating these branches with a single parameter-sharing branch, with minimal (insignificant) performance loss.

5.2 Feature-Level Imbalance

Definition. The integration of the features from the backbone network is expected to be balanced in terms of low- and high-level features so that consistent predictions can follow. To be more specific, if we analyze the conventional FPN architecture in Fig. 9, we notice that, while there are several layers from the C2 layer of the bottom-up pass with low-level features to the P5 layer of the feature pyramid, the C2 layer is directly integrated to the P2 layer, which implies the effect of the high-level and low-level features in the P2 and P5 layers to be different.

Solutions. There are several methods to address imbalance in the FPN architectures, which range from designing improved top-down pathway connections [62], [63] to completely novel architectures. Here, we consider the methods to alleviate the feature-level imbalance problem using novel architectures, which we group into two according to what they use as a basis, pyramidal or backbone features.

5.2.1 Methods Using Pyramidal Features as a Basis

These methods aim to improve the pyramidal features gathered by FPN using additional operations or steps – see an overview of these methods in Figs. 10a, 10b.

Path Aggregation Network (PANet) [96] is the first to show that the features gathered by an FPN can be further enhanced and an RoI can be mapped to each layer of the pyramid rather than associating it with a single one. The authors suggest that low-level features, such as edges, corners, are useful for localizing objects, however, the FPN architecture does not sufficiently make use of these features. Motivated from this observation, PANet, depicted in Fig. 10a, improves the FPN architecture with two new contributions:

- 1) *Bottom-up Path Augmentation* extends the feature pyramid in order to allow the low-level features to arrive at the layers where the predictions occur in shorter steps (see red arrows in Fig. 10a within FPN and to the final pyramidal features to see the shortcut). For this reason, in a way a shortcut is created for the features in the initial layers. This is important since these features have rich information about localization thanks to edges or instance parts.
- 2) While in the FPN, each RoI is associated with a single level of feature based on its size, PANet associates

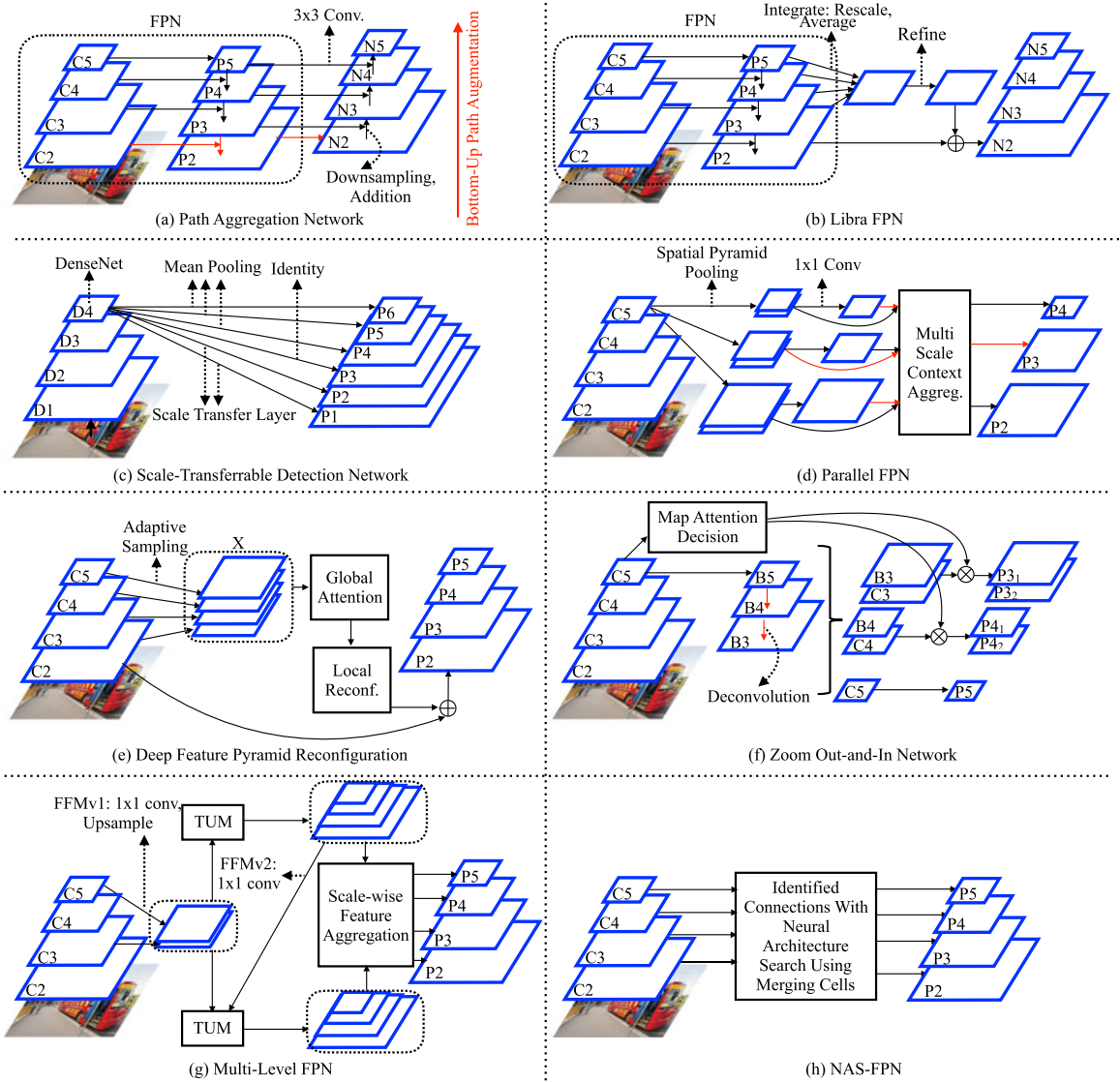


Fig. 10. High-level diagrams of the methods designed for feature-level imbalance. (a) *Path Aggregation Network*. FPN is augmented by an additional bottom-up pathway to facilitate a shortcut of the low-level features to the final pyramidal feature maps. Red arrows represent the shortcuts. (b) *Libra FPN*. FPN pyramidal features are integrated and refined to learn a residual feature map. We illustrate the process originating from P_2 feature map. Remaining feature maps (P_3 - P_5) follow the same operations. (c) *Scale Transferrable Detection Network*. Pyramidal features are learned via pooling, identity mapping and scale transfer layers depending on the layer size. (d) *Parallel FPN*. Feature maps with difference scales are followed by spatial pyramid pooling. These feature maps are fed into the multi-scale context aggregation (MSCA) module. We show the input and outputs of MSCA module for P_3 by red arrows. (e) *Deep Feature Pyramid Reconfiguration*. A set of residual features are learned via global attention and local reconfiguration modules. We illustrate the process originating from P_2 feature map. Remaining feature maps (P_3 - P_5) follow the same operations. (f) *Zoom Out-And-In Network*. A zoom-in phase based on deconvolution (shown with red arrows) is adopted before stacking the layers of zoom-out and zoom-in phases. The weighting between them is determined by map attention decision module. (g) *Multi-Level FPN*. Backbone features from two different levels are fed into Thinned U-Shape Module (TUM) recursively to generate a sequence of pyramidal features, which are finally combined into one by a scale-wise feature aggregation module. (h) *NAS-FPN*. The layers between backbone features and pyramidal features are learned via Neural Architecture Search.

each RoI to every level, applies RoI Pooling, fuses using element-wise max or sum operation and the resulting fixed-sized feature grid is propagated to the detector network. This process is called *Adaptive Feature Pooling*.

Despite these contributions, PANet still uses a sequential pathway to extract the features.

Different from the sequential enhancement pathway of PANet, *Libra FPN* [29] aims to learn the residual features by using all of the features from all FPN layers at once (see Fig. 10b). Residual feature layer computation is handled in two steps:

- 1) *Integrate*: All feature maps from different layers are reduced to one single feature map by rescaling and averaging. For this reason, this step does not have any learnable parameter.
- 2) *Refine*: The integrated feature map is refined by means of convolution layers or non-local neural networks [97].

Finally the refined features are added to each layer of the pyramidal features. The authors argue that in addition to FPN, their method is complementary to other methods based on pyramidal features as well, such as PANet [96].

5.2.2 Methods Using Backbone Features as a Basis

These methods build their architecture on the backbone features and ignore the top-down pathway of FPN by employing different feature integration mechanisms, as displayed in Figs. 10c, 10d, 10e, 10f, 10g, 10h.

Scale-Transferrable Detection Network (STDN) [98] generates the pyramidal features from the last layer of the backbone features which are extracted using DenseNet [99] blocks (Fig. 10c). In a DenseNet block, all the lower level features are propagated to every next layer within a block. In Fig. 10c, the number of DenseNet (dense) blocks is four and the i th block is denoted by D_i . Motivated by the idea that direct propagation of lower-level layers to the subsequent layers also carries lower-level information, STDN builds pyramidal features consisting of six layers by using the last block of DenseNet. In order to map these layers to lower sizes, the approach uses mean pooling with different receptive field sizes. For the fourth feature map, an identity mapping is used. For the last two layers which the feature maps of DenseNet are to be mapped to higher dimensions, the authors propose a *scale transfer layer* approach. This layer does not have any learnable parameter and given r , the desired enlargement for a feature map, the width and height of the feature map are enlarged by r by decreasing the total number of feature maps (a.k.a. channels). STDN incorporates high- and low-level features with the help of DenseNet blocks and is not easily adaptable to other backbone networks. In addition, no method is adopted to balance the low- and high-level features within the last block of DenseNet.

Similar to STDN, *Parallel FPN* [100] also employs only the last layer of the backbone network and generates multi-scale features by exploiting the spatial pyramid pooling (SPP) [101] – Fig. 10d. Differently, it increases the width of the network by pooling the last D feature maps of the backbone network multiple times with different sizes, such that feature maps with different scales are obtained. Fig. 10e shows the case when it is pooled for three times and $D = 2$. The number of feature maps is decreased to 1 by employing 1×1 convolutions. These feature maps are then fed into the *multi-scale context aggregation (MSCA) module*, which integrates context information from other scales for a corresponding layer. For this reason, MSCA, operating on a scale-based manner, has the following inputs: Spatial pyramid pooled D feature maps and reduced feature maps from other scales. We illustrate the inputs and outputs to the MSCA module for the middle scale feature map by red arrows in Fig. 10e. MSCA first ensures the sizes of each feature map to be equal and applies 3×3 convolutions.

While previous methods based on backbone features only use the last layer of the backbone network, *Deep Feature Pyramid Reconfiguration* [102] combines features from different levels of backbone features into a single tensor (X in Fig. 10e) and then learns a set of residual features from this tensor. A sequence of two modules are applied to the tensor X in order to learn a residual feature map to be added to each layer of the backbone network. These modules are,

dimensional features for each feature map initially (i.e. squeeze step), and then a weight is learned for each feature map based on learnable functions including non-linearity (i.e. excitation step).

- 2) *Local Configuration Module* aims to improve the features after global attention module by employing convolutional layers. The output of this module presents the residual features to be added for a feature layer from the backbone network.

Similarly, *Zoom Out-and-In Network* [104] also combines low- and high-level features of the backbone network. Additionally, it includes deconvolution-based zoom-in phase in which intermediate step pyramidal features, denoted by Bis in Fig. 10f, are learned. Note that, unlike FPN [26], there is no lateral connection to the backbone network during the zoom-in phase, which is basically a sequence of deconvolutional layers (see red arrows for the zoom-in phase). Integration of the high- and low-level features are achieved by stacking the same-size feature maps from zoom-out and zoom-in phases after zoom-in phase (i.e. $B3$ and $C3$). On the other hand, these concatenated feature maps are to be balanced especially for the $B3 - C3$ and $B4 - C4$ blocks since $B3$ and $C3$ (or $B4$ and $C4$) are very far from each other in the feature hierarchy, which makes them have different representations of the data. In order to achieve this, the proposed *map attention decision module* learns a weight distribution on the layers. Note that the idea is similar to the squeeze and excitation modules [103] employed by Kong *et al.* [102], however, it is shown by the authors that their design performs better for their architecture. One drawback of the method is that it is built upon Inception v2 (a.k.a. Inception BN) [105] and corresponding inception modules are exploited throughout the method, which may make the method difficult to adopt for other backbone networks.

Different from Kong *et al.* [102] and Li *et al.* [104], *Multi-Level FPN* [106] stacks one highest and one lower level feature layers and recursively outputs a set of pyramidal features, which are all finally combined into a single feature pyramid in a scale-wise manner (see Fig. 10g). *Feature fusion module (FFM) v1* equals the dimensions of the input feature maps by a sequence of 1×1 convolution and upsampling operations. Then, the resulting two-layer features are propagated to *thinned U-shape modules (TUM)*. Excluding the initial propagation, each time these two-layer features are integrated to the output of the previous TUM by 1×1 convolutions in *FFMv2*. Note that the depth of the network is increasing after each application of the TUM and the features are becoming more high level. As a result of this, a similar problem with the FPN feature imbalance arise again. As in the work proposed by [102], the authors employed squeeze and excitation networks [103] to combine different pyramidal shape features.

Rather than using hand-crafted architectures, *Neural Architecture Search FPN (NAS-FPN)* [107] aims to search for the best architecture to generate pyramidal features given the backbone features by using neural architecture search methods [108] – Fig. 10h. This idea was also previously applied to the image classification task and showed to perform well [109], [110], [111]. *Auto-FPN* [135] is also another example for using NAS while learning the connections from backbone features to pyramidal features and beyond. While

- 1) *Global Attention Module* aims to learn the inter-dependencies among different feature maps for tensor X . The authors adopt Squeeze and Excitation Blocks [103] in which the information is squeezed to lower

NAS-FPN achieves higher performance, Auto-FPN is more efficient and has less memory footprint. The idea is also applied to backbone design for object detection by Chen *et al.* [112], however it is not within the scope of our paper. Considering their performance in other tasks such as EfficientNet [111] and different definitions of search spaces may lead to better performance in NAS methods, more work is expected for FPN design using NAS.

5.3 Comparative Summary

SSD [19] provides an analysis on the importance of making predictions from different number of layers with varying scales. Increasing the number of prediction layers leads to a significant performance gain. While predicting from one layer provides 62.4 mAP@0.5 on Pascal VOC 2007 test set [51] with SSD-300, it is 70.7 and 74.6 when the number of prediction layers is 3 and 5 respectively (while keeping the number of predictions almost equal). Therefore, once addressed in the detection pipeline, scale imbalance methods can significantly boost performance.

Feature pyramid based methods increased the performance significantly compared to SSD. Once included in Faster R-CNN with ResNet-101, the pioneering FPN study [26] has a relative improvement of 3.7 percent on MS COCO testdev (from 34.9 to 36.2). Another virtue of the feature pyramids is the efficiency due to having lighter detection networks: e.g., the model with FPN is reported to be more than two times faster than baseline Faster R-CNN with ResNet-50 backbone (150ms vs 320ms per image on a single NVIDIA M40 GPU) [26]. Currently, the most promising results are obtained via NAS by learning how to combine the backbone features from different levels. Using the ResNet-50 backbone with 1024×1024 image size, a 10.2 percent relative improvement is obtained on MS COCO testdev (from 40.1 to 44.2) by NAS-FPN [107] but it needs also to be noted that number of parameters in the learned structure is approximately two times higher and inference time is 26 percent more (92.1 ms vs 73.0 ms per image on a single P100 GPU). Inference time is also a major drawback of the image pyramid based methods. While SNIP reports inference at approximately 1 image/sec, the faster version, SNIPER, improves it to 5 images/sec on a V100 GPU (200 ms/image).

In order to adjust the inference time-performance trade-off in scale imbalance, a common method is to use multi-scale images with one backbone and multiple lighter networks. All methods in Section 5.1.4 are published last year. To illustrate the performance gains: Pang *et al.* [92] achieved 19.5 percent on MS COCO test-dev relative improvement with 12ms per image inference time compared to SSD300 with the same size image and backbone network having 10ms per image inference time (mAPs are 25.1 vs 30.0).

5.4 Open Issues

Here we discuss open problems concerning scale imbalance.

5.4.1 Characteristics of Different Layers of Feature Hierarchies

In feature-pyramid based methods (Section 5.2), a prominent and common pattern is to include a top-down pathway in order to integrate higher-layer features with lower-layer ones.

Although this approach has yielded promising improvements in performance, an established perspective about what critical aspects of features (or information) are handled differently in those methods is missing. Here, we highlight three such aspects:

(i) *Abstractness*. Higher layers in a feature hierarchy carry high-level, semantically more meaningful information about the objects or object parts whereas the lower layers represent low-level information in the scene, such as edges, contours, corners etc. In other words, higher-layer features are more abstract.

(ii) *Coarseness*. To reduce dimensions, feature networks gradually reduce the size of the layers towards the top of the hierarchy. Although this is reasonable given the constraints, it has an immediate outcome on the number of neurons that a fixed bounding box at the image level encapsulates at the different feature layers. Namely, the BB will include less neurons when projected to the highest layer. In other words, higher layers are more coarse.

(iii) *Cardinality*. In FPN and many of its variants, prediction is performed for an object from the layer that matches the object's scale. Since the scales of objects are not balanced, this approach has a direct affect on the number of predictions and backpropagation performed through a layer.

We argue that analyzing and addressing these aspects in a more established manner is critical for developing more profound solutions. Although we see that some methods handle imbalance in these aspects (e.g. Libra FPN [29] addresses all three aspects, Path Aggregation Network [96] handles abstractness and cardinality, whereas FPN solves only abstractness to a certain extent), these aspects should be quantified and used for comparing different methods.

5.4.2 Image Pyramids in Deep Object Detectors

Open Issue. It is hard to exploit image pyramids using neural networks due to memory limitations. Therefore, finding solutions alleviating this constraint (e.g., as in SNIP [27]) is still an open problem.

Explanation. The image pyramids (Fig. 8d) were commonly adopted by the pre-deep learning era object detectors. However, memory limitations motivated the methods based on pyramidal features which, with less memory, are still able to generate a set of features with different scales allowing predictions to occur at multiple scales. On the other hand, feature-pyramids are actually approximations of the features extracted from image pyramids, and there is still room for improvement given that using image pyramids is not common among deep object detectors.

6 IMBALANCE 3: SPATIAL IMBALANCE

Definition. *Size, shape, location – relative to both the image or another box – and IoU are spatial attributes of bounding boxes. Any imbalance in such attributes is likely to affect the training and generalization performance. For example, a slight shift in position may lead to drastic changes in the regression (localization) loss, causing an imbalance in the loss values, if a suitable loss function is not adopted. In this section, we discuss these problems specific to the spatial attributes and regression loss.*

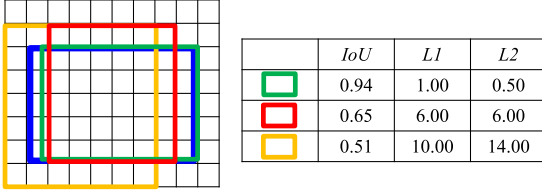


Fig. 11. An illustration of imbalance in regression loss. Blue denotes the ground truth BB. There are three prediction boxes, marked with green, red and yellow colors. In the table on the right, $L1$ and $L2$ columns show the sum of $L1$ and $L2$ errors between the box corners of the associated prediction box and the ground-truth (blue) box. Note that the contribution of the yellow box to the $L2$ loss is more dominating than its effect on total $L1$ error. Also, the contribution of the green box is less for the $L2$ error.

6.1 Imbalance in Regression Loss

Definition. This imbalance problem is concerned with the uneven contributions of different individual examples to the regression loss. Fig. 11 illustrates the problem using $L1$ and $L2$ losses, where the hard example (i.e. the one with low IoU, the yellow box) is dominating the $L2$ loss, whereas $L1$ loss assigns relatively more balanced errors to all examples.

Solutions. The regression losses for object detection have evolved under two main streams: The first one is the Lp -norm-based (e.g. $L1$, $L2$) loss functions and the second one is the IoU-based loss functions. Table S2 in the Supplementary Material presents a comparison of the widely used regression losses.

Replacing $L2$ regression loss [17], [113], *Smooth $L1$ Loss* [17] is the first loss function designed specifically for deep object detectors, and it has been widely adopted (e.g. [19], [21], [22]) since it reduces the effect of the outliers (compared to $L2$ loss) and it is more stable for small errors (compared to $L1$ loss). Smooth $L1$ loss, a special case of Huber Loss [114], is defined as:

$$L1_{smooth}(\hat{x}) = \begin{cases} 0.5\hat{x}^2, & \text{if } |\hat{x}| < 1 \\ |\hat{x}| - 0.5, & \text{otherwise.} \end{cases} \quad (9)$$

where \hat{x} is the difference between the estimated and the target BB coordinates.

Motivated by the fact that the gradients of the outliers still have a negative effect on learning the inliers with smaller gradients in Smooth $L1$ loss, *Balanced $L1$ Loss* [29] increases the gradient contribution of the inliers to the total loss value. To achieve this, the authors first derive the definition of the loss function originating from the desirable balanced gradients across inliers and outliers:

$$\frac{\partial L1_{balanced}}{\partial \hat{x}} = \begin{cases} \alpha \ln(b|\hat{x}| + 1), & \text{if } |\hat{x}| < 1 \\ \theta, & \text{otherwise,} \end{cases} \quad (10)$$

where α controls how much the inliers are promoted (small α increases the contribution of inliers); θ is the upper bound of the error to help balancing between the tasks. Integrating Equation (10), $L1_{balanced}$ is derived as follows:

$$L1_{balanced}(\hat{x}) = \begin{cases} \frac{\alpha}{b} (b|\hat{x}| + 1) \ln(b|\hat{x}| + 1) - \alpha|\hat{x}|, & \text{if } |\hat{x}| < 1 \\ \gamma|\hat{x}| + Z, & \text{otherwise,} \end{cases} \quad (11)$$

where b ensures $L1_{balanced}(\hat{x} = 1)$ is a continuous function, Z is a constant and the association between the hyper-parameters is:

Authorized licensed use limited to: UNIVERSIDAD CARLOS III MADRID. Downloaded on August 11, 2024 at 18:22:01 UTC from IEEE Xplore. Restrictions apply.

$$\alpha \ln(b + 1) = \gamma. \quad (12)$$

Having put more emphasis on inliers, Balanced $L1$ Loss improves the performance especially for larger IoUs (namely, $AP@0.75$ improves by %1.1).

Another approach, *Kullback-Leibler Loss* (KL Loss) [53], is driven by the fact that the ground truth boxes can be ambiguous in some cases due to e.g. occlusion, shape of the object or inaccurate labeling. For this reason, the authors aim to predict a probability distribution for each BB coordinate rather than direct BB prediction. The idea is similar to the networks with an additional localization confidence associated prediction branch [115], [116], [117], [118], [119], besides classification and regression, in order to use the predicted confidence during inference. Differently, KL Loss, even without its proposed NMS, has an improvement in localization compared to the baseline. The method assumes that each box coordinate is independent and follows a Gaussian distribution with mean \hat{x} and standard deviation σ . Therefore, in addition to conventional boxes, a branch is added to the network to predict the standard deviation, that is σ , and the loss is backpropagated using the KL divergence between the prediction and the ground truth such that the ground truth boxes are modelled by the dirac delta distribution centered at the box coordinates. With these assumptions, KL Loss is proportional to:

$$L_{KL}(\hat{x}, \sigma) \propto \frac{\hat{x}^2}{2\sigma^2} + \frac{1}{2} \log \sigma^2. \quad (13)$$

They also employ gradient clipping similar to smooth $L1$ in order to decrease the effect of the outliers. During NMS, a voting scheme is also proposed to combine bounding boxes with different probability distributions based on the certainty of each box; however, this method is out of the scope of our paper. Note that the choice of probability distribution for bounding boxes matters since the loss definition is affected by this choice. For example, Equation (13) degenerates to Euclidean distance when $\sigma = 1$.

In addition to the Lp -norm-based loss functions, there are also IoU based loss functions which exploit the differentiable nature of IoU. An earlier example is the *IoU Loss* [120], where an object detector is successfully trained by directly formulating a loss based on the IoU as:

$$L_{IoU} = -\ln(IoU). \quad (14)$$

Another approach to exploit the metric-nature of $1 - IoU$ is the *Bounded IoU loss* [121]. This loss warps a modified version of $1 - IoU$ to the smooth $L1$ function. The modification involves bounding the IoU by fixing all the parameters except the one to be computed, which implies the computation of the maximum attainable IoU for one parameter:

$$L_{BIOU}(x, \hat{x}) = 2L1_{smooth}(1 - IoU_B(x, \hat{x})), \quad (15)$$

where the bounding boxes are represented by center coordinates, width and height as $[c_x, c_y, w, h]$. Here, we define the bounded IoU only for c_x and w since c_y and h have similar definitions. We follow our notation in Section 2 to denote ground truth and detection (i.e. c_x for ground truth and \hat{c}_x for detection). With this notation, IoU_B , the bounded IoU, is defined as follows:

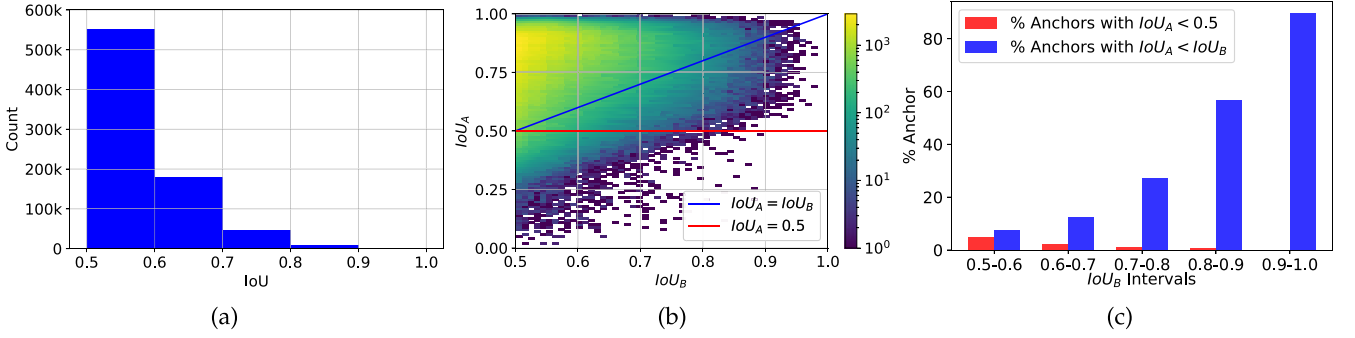


Fig. 12. The IoU distribution of the positive anchors for a converged RetinaNet [22] with ResNet-50 [57] on MS COCO [54] before regression (a), and (b) the density of how the IoU values are affected by regression (IoU_B : before regression, IoU_A : after regression). For clarity, the density is plotted in log-scale. (c) A concise summary of how regression changes IoUs of anchors as a function of their starting IoUs (IoU_B). Notice that while it is better not to regress the boxes in larger IoUs, regressor causes false positives more in lower IoUs.

$$IoU_B(c_x, \bar{c}_x) = \max\left(0, \frac{w - 2|\bar{c}_x - c_x|}{w + 2|\bar{c}_x - c_x|}\right), \quad (16)$$

$$IoU_B(w, \bar{w}) = \min\left(\frac{\bar{w}}{w}, \frac{w}{\bar{w}}\right). \quad (17)$$

In such a setting, $IoU_B \geq IoU$. Also, an IoU based loss function is warped into the smooth L1 function in order to make the ranges of the classification and localization task consistent and to decrease the effect of outliers.

Motivated by the idea that the best loss function is the performance metric itself, in *Generalized Intersection over Union* (GIoU) [122] showed that IoU can be directly optimized and that IoU and the proposed GIoU can be used as a loss function. GIoU is proposed as both a performance measure and a loss function while amending the major drawback of the IoU (i.e. the plateau when $IoU=0$) by incorporating an additional smallest enclosing box E . In such a way, even when two boxes do not overlap, a GIoU value can be assigned to them and this allows the function to have non-zero gradient throughout the entire input domain rather being limited to $IoU>0$. Unlike IoU, the $GIoU(B, \bar{B}) \in [-1, 1]$. Having computed E , GIoU is defined as:

$$L_{GIoU}(B, \bar{B}) = IoU(B, \bar{B}) - \frac{A(E) - A(B \cup \bar{B})}{A(E)}, \quad (18)$$

where GIoU is a lower bound for IoU, and it converges to IoU when $A(B \cup \bar{B}) = A(E)$. GIoU preserves the advantages of IoU, and makes it differentiable when $IoU=0$. On the other hand, since positive labeled BBs have IoU larger than 0.5 by definition, this portion of the function is never visited in practice, yet still, GIoU Loss performs better than using IoU directly as a loss function.

A different idea proposed by Zheng *et al.* [123] is to add penalty terms to the conventional IoU error (i.e. $1 - IoU(B, \bar{B})$) in order to ensure faster and more accurate convergence. To achieve that, in Distance IoU (DIoU) Loss, a penalty term related with the distances of the centers of B and \bar{B} is added as:

$$L_{DIoU}(B, \bar{B}) = 1 - IoU(B, \bar{B}) + \frac{d^2(B_C, \bar{B}_C)}{E_D^2}, \quad (19)$$

where $d()$ is the Euclidean distance, B_C is the center point of box B and E_D is the diagonal length of E (i.e. smallest enclosing box). To further enhance their method, L_{DIoU} is extended with an additional penalty term for inconsistency in aspect ratios of two boxes. The resulting loss function, coined as Complete IoU (CIoU), is defined as:

$$L_{CIoU}(B, \bar{B}) = 1 - IoU(B, \bar{B}) + \frac{d^2(B_C, \bar{B}_C)}{E_D^2} + \alpha v, \quad (20)$$

such that

$$v = \frac{4}{\pi^2} \left(\arctan\left(\frac{w}{h}\right) - \arctan\left(\frac{\bar{w}}{\bar{h}}\right) \right)^2, \quad (21)$$

and

$$\alpha = \frac{v}{(1 - IoU(B, \bar{B})) + v}. \quad (22)$$

In this formulation, α , the trade-off parameter, ensures the importance of the cases with smaller IoUs to be higher. In the paper, one interesting approach is to validate faster and more accurate convergence by designing simulation scenarios since it is not straightforward to analyze IoU-based loss functions (see Section 6.5).

6.2 IoU Distribution Imbalance

Definition. IoU distribution imbalance is observed when input bounding boxes have a skewed IoU distribution. The problem is illustrated in Fig. 12a, where the IoU distribution of the anchors in RetinaNet [22] are observed to be skewed towards lower IoUs. It has been previously shown that this imbalance is also observed while training two-stage detectors [124]. Differently, we present in Fig. 12b how each anchor is affected by regression. The rate of degraded anchors after regression (i.e. positive anchors under the blue line) decreases towards the threshold that the regressor is trained upon, which quantitatively confirms the claims of Cai *et al.* (see Fig. 12c). On the other hand, the rate of the anchors that become a false positive (i.e. positive anchors under the red line), is increasing towards the 0.5 – 0.6 bin, for which around 5 percent of the positive anchors are lost by the regressor (see Fig. 12c). Furthermore, comparing the average IoU error of the anchors before and after regression on a converged model, we notice that it is better off without applying regression and use the unregressed anchors for the IoU_B intervals 0.8 – 0.9 and

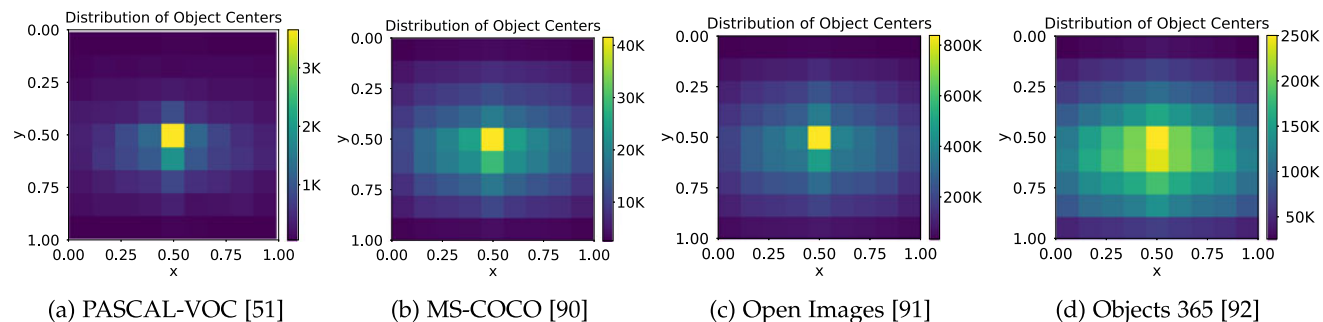


Fig. 13. Distribution of the centers of the objects in the common datasets over the normalized image.

0.9 – 1.0. These results suggest that there is still room to improve by observing the effect of the regressor on the IoUs of the input bounding boxes.

Solutions. The *Cascade R-CNN* method [124] has been the first to address the IoU imbalance. Motivated by the arguments that (i) a single detector can be optimal for a single IoU threshold, and (ii) skewed IoU distributions make the regressor overfit for a single threshold, they show that the IoU distribution of the positive samples has an effect on the regression branch. In order to alleviate the problem, the authors trained three detectors, in a cascaded pipeline, with IoU thresholds 0.5, 0.6 and 0.7 for positive samples. Each detector in the cascade uses the boxes from the previous stage rather than sampling them anew. In this way, they show that the skewness of the distribution can be shifted from the left-skewed to approximately uniform and even to the right-skewed, thereby allowing the model to have enough samples for the optimal IoU threshold that it is trained with. The authors show that such a cascaded scheme works better compared to the previous work, such as Multi-Region CNN [125] and AttracNet [126], that iteratively apply the same network to the bounding boxes. Another cascade-based structure implicitly addressing the IoU imbalance is *Hierarchical Shot Detector* (HSD) [127]. Rather than using a classifier and regressor at different cascade-levels, the method runs its classifier after the boxes are regressed, resulting in a more balanced distribution.

In another set of studies, randomly generated bounding boxes are utilized to provide a set of positive input bounding boxes with balanced IoU distribution to the second stage of Faster R-CNN. *IoU-uniform R-CNN* [117] adds controllable jitters and in such a way provides approximately uniform positive input bounding boxes to the regressor only (i.e. the classification branch still uses the RPN RoIs). Differently, *pRoI Generator* [77] trains both branches with the generated RoIs but the performance improvement is not that significant probably because the training set covers a much larger space than the test set. However, one significant contribution of Oksuz *et al.* [77] is, rather than adding controllable jitters, they systematically generate bounding boxes using the proposed bounding box generator. Using this pRoI generator, they conducted a set of experiments for different IoU distributions and reported the following: (i) The IoU distribution of the input bounding boxes has an effect not only on the regression but also on the classification performance. (ii) Similar to the finding of Pang *et al.* [29], the IoU of the examples is related to their

hardness. However, contrary to Cao *et al.* [30], who argued that OHEM [24] has an adverse effect when applied only to positive examples, Oksuz *et al.* [77] showed that the effect of OHEM depends on the IoU distribution of the positive input BBs. When a right-skewed IoU distribution is used with OHEM, a significant performance improvement is observed. (iii) The best performance is achieved when the IoU distribution is uniform.

6.3 Object Location Imbalance

Definition. The distribution of the objects throughout the image matters because current deep object detectors employ densely sampled anchors as sliding window classifiers. For most of the methods, the anchors are evenly distributed within the image, so that each part in the image is considered with the same importance level. On the other hand, the objects in an image do not follow a uniform distribution (see Fig. 13), i.e. there is an imbalance about object locations.

Solutions. Motivated by the fact that the objects are not distributed uniformly over the image, Wang *et al.* [128] aim to learn the location, scale and aspect ratio attributes of the anchors concurrently in order to decrease the number of anchors and improve recall at the same time. Specifically, given the backbone feature maps, a prediction branch is designed for each of these tasks to generate anchors: (i) anchor location prediction branch predicts a probability for each location to determine whether the location contains an object, and a hard thresholding approach is adopted based on the output probabilities to determine the anchors, (ii) anchor shape prediction branch generates the shape of the anchor for each location. Since the anchors vary depending on the image, different from the conventional methods (i.e. one-stage generators and RPN) using a fully convolutional classifier over the feature map, the authors proposed anchor-guided feature adaptation based on deformable convolutions [129] in order to have a balanced representation depending on the anchor size. Rather than learning the anchors, *free anchor* method [130] loosens the hard constraint of the matching strategy (i.e. a sample being positive if $IoU > 0.5$) and in such a way, each anchor is considered a matching candidate for each ground truth. In order to do that, the authors pick a bag of candidate anchors for each ground truth using the sorted IoUs of the anchors with the ground truth. Among this bag of candidates, the proposed loss function aims to enforce the most suitable anchor by considering both the regression and the classification task to be matched with the ground truth.

6.4 Comparative Summary

Addressing spatial imbalance problems has resulted in significant improvements for object detectors. In general, while the methods for imbalance in regression loss and IoU distribution imbalance yield improvement especially in the regressor branch, removing the bias in the anchors in the location imbalance improves classification, too.

Despite the widespread use of the Smooth $L1$ loss, four new loss functions have been proposed last year. Chen *et al.* [131] compared Balanced $L1$, IoU, GIoU, Bounded IoU on Faster R-CNN+FPN against Smooth $L1$, and reported relative improvement of 0.8, 1.1, 1.4 and -0.3 percent respectively. With the same configuration, DIOU and CIOU losses are reported to have a relative improvement of 0.2 and 1.7 percent against the GIoU baseline (without DIOU-NMS). Compared to the loss functions designed for the foreground-background imbalance problem (i.e. Focal Loss - see Section 4.1.2), the relative improvement of the regression losses over Smooth $L1$ are smaller. We also analyzed for which cases the baseline loss function fails in Fig. 12b, which can be used as a tool to compare the pros/cons of different regression loss functions.

Cascaded structures are proven to be very useful for object detection by regulating the IoU distribution. Cascade R-CNN, being a two-stage detector, has a relative improvement of 18.2 percent on MS-COCO testdev compared to its baseline Faster R-CNN+FPN with ResNet-101 backbone (36.2 vs. 42.8). A one-stage architecture, HSD, also has a relative improvement of 34.7 percent compared to the SSD512 with VGG-16 backbone (28.8 vs 38.8). It is difficult to compare Cascade R-CNN and HSD since they are different in nature (one-stage and two-stage pipelines), and their performances were reported for different input resolutions. However, we observed that HSD with a 768×768 input image runs approximately $1.5\times$ faster than Cascade R-CNN for a 1333×800 image and achieves slightly lower performance on MS-COCO testdev (42.3 vs. 42.8). One observation between these two is that, even though HSD performs worse than Cascade R-CNN at $AP@0.5$, it yields better performance for $AP@0.75$ than Cascade R-CNN, which implies that the regressor of HSD is better trained.

As for the methods that we discussed for location imbalance, guided anchoring [128] increases average recall by 9.1 percent with 90 percent fewer anchors via learning the parameters of the anchors. Compared to guided anchoring, free anchor [130] reports a lower relative improvement for average recall with 2.4 percent against RetinaNet with ResNet50, however it has a 8.4 percent relative improvement (from 35.7 to 38.7).

6.5 Open Issues

This section discusses the open issues related to the spatial properties of the input bounding boxes and objects.

6.5.1 A Regression Loss With Many Aspects

Open Issue. Recent studies have proposed alternative regression loss definitions with different perspectives and aspects. Owing to their benefits, a single regression loss function that can combine these different aspects can be beneficial.

Explanation. Recent regression loss functions have different motivations: (i) Balanced $L1$ Loss [29] increases the

contribution of the inliers. (ii) KL Loss [53] is motivated from the ambiguity of the positive samples. (iii) IoU-based loss functions have the motive to use a performance metric as a loss function. These seemingly mutually exclusive motives can be integrated to a single regression loss function.

6.5.2 Analyzing the Loss Functions

In order to analyze how outliers and inliers affect the regression loss, it is useful to analyze the loss function and its gradient with respect to the inputs. To illustrate such an analysis, in Focal Loss [22], the authors plot the loss function with respect to the confidence score of the ground truth class with a comparison to the cross entropy loss, the baseline. Similarly, in Balanced $L1$ Loss [29], both the loss function itself and the gradients are depicted with a comparison to Smooth $L1$ Loss. Such an analysis might be more difficult for the recently proposed more complex loss functions. As an example, AP Loss [68] is computed considering the ranking of the individual examples, which is based on the confidence scores of all BBs. So, the loss depends on the entire set rather than individual examples, which makes it difficult to plot the loss (and its gradient) for a single input as conventionally done. Another example is GIoU Loss [122], which uses the ground truth box and the smallest enclosing box in addition to the detection box. Each box is represented by four parameters (see Section 6.1), which creates a total of twelve parameters. For this reason, it is necessary to develop appropriate analysis methods to observe how these loss functions penalize the examples.

6.5.3 Designing Better Anchors

Designing an optimal anchor set with high recall has received limited attention. The Meta Anchor [132] method attempts to find an optimal set of aspect ratios and scales for anchors. More recently, Wang *et al.* [128] have improved recall more than 9 percent on MS COCO dataset [54] while using 90 percent less anchors than RPN [21]. Addressing the imbalanced nature of the locations and scales of the objects seems to be an open issue.

6.5.4 Relative Spatial Distribution Imbalance

Open Issue. As we discussed in Section 6, the distribution of the IoU between the estimated BBs and the ground truth is imbalanced and this has an influence on the performance. A closer inspection [77] reveals that the locations of estimated BBs relative to the matching ground truths also have an imbalance. Whether this imbalance affects the performance of the object detectors remains to be investigated.

Explanation. During the conventional training of the object detectors, input bounding boxes are labeled as positive when their IoU with a ground truth is larger than 0.5. This is adopted in order to provide more diverse examples to the classifier and the regressor, and to allow good quality predictions at test time from noisy input bounding boxes. The work by Oksuz *et al.* [77] is currently the only study that points to an imbalance in the distribution of the relative location of BBs. Exploiting the scale-invariance and shift-invariance properties of the IoU, they plotted the top-left point of the RoIs from the RPN (of Faster R-CNN) with respect to a single reference

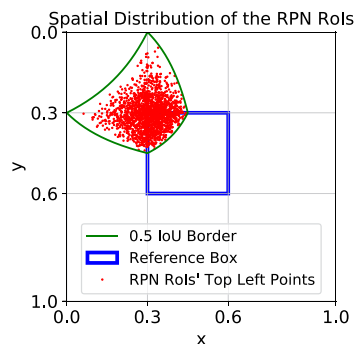


Fig. 14. The (relative) spatial distributions of $1K$ Rols with IoU between 0.5 to 0.6 from the RPN (of Faster R-CNN with ResNet101 backbone) collected from Pascal [51] during the last epoch of the training. A bias is observed around the top-left corners of ground truths such that Rols are densely concentrated at the top-left corner of the normalized ground truth box.

box representing the ground truth (see Fig. 14). They reported that the resulting spatial distribution of the top-left points of the RPN Rols are skewed towards the top-left point of the reference ground truth box. We see that the samples are scarce away from the top-left corner of the reference box.

6.5.5 Imbalance in Overlapping BBs

Open Issue. Due to the dynamic nature of bounding box sampling methods (Section 4.1), some regions in the input image may be over-sampled (i.e. regions coinciding with many overlapping boxes) and some regions may be under-sampled (or not even sampled at all). The effect of this imbalance caused by BB sampling methods has not been explored.

Explanation. Imbalance in overlapping BBs is illustrated in Figs. 15a, 15b, 15c on an example grid representing the image and six input BBs (four negative and two positive). The number of overlapping BBs for each pixel is shown in Fig. 15c; in this example, this number ranges from 0 to 5.

This imbalance may affect the performance for two reasons: (i) The number of highly sampled regions will play more role in the final loss functions, which can lead the method to overfit for specific features. (ii) The fact that some regions are over-sampled and some are under-sampled might have adverse effects on learning, as the size of sample (i.e. batch size) is known to be related to the optimal learning rate [133].

6.5.6 Analysis of the Orientation Imbalance

Open Issue. The effects of imbalance in the orientation distribution of objects need to be investigated.

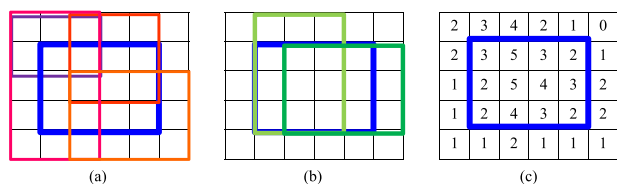


Fig. 15. An illustration of the imbalance in overlapping BBs. The grid represents the pixels of the image/feature map, and blue bounding box is the ground-truth. (a) Four negative input bounding boxes. (b) Two positive input bounding boxes. (c) Per-pixel number-of-overlaps of the input bounding boxes. Throughout the image, the number of sampling frequencies for the pixels vary due to the variation in the overlapping number of bounding boxes.

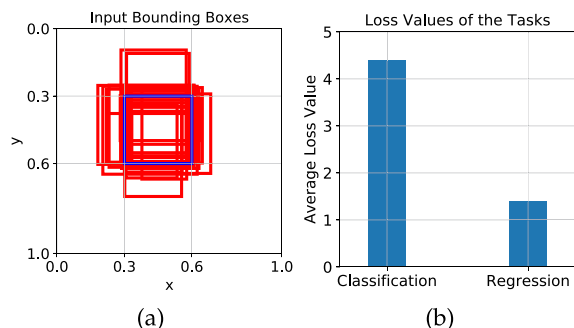


Fig. 16. (a) Randomly sampled 32 positive Rols using the pRol Generator [77]. (b) Average classification and regression losses of these Rols at the initialization of the object detector for MS COCO dataset [54] with 80 classes. We use cross entropy for the classification task assuming that initially each class has the same confidence score, and smooth L1 loss for regression task. Note that right after initialization, the classification loss has more effect on the total loss.

Explanation. The distribution of the orientation of object instances might have an effect on the final performance. If there is a typical orientation for the object, then the detector will likely overfit to this orientation and will make errors for the other orientations. To the best of our knowledge, this problem has not yet been explored.

7 IMBALANCE 4: OBJECTIVE IMBALANCE

Definition. Objective imbalance pertains to the objective (loss) function that is minimized during training. By definition, object detection requires a multi-task loss in order to solve classification and regression tasks simultaneously. However, different tasks can lead to imbalance because of the following differences: (i) The norms of the gradients can be different for the tasks, and one task can dominate the training (see Fig. 16). (ii) The ranges of the loss functions from different tasks can be different, which hampers the consistent and balanced optimization of the tasks. (iii) The difficulties of the tasks can be different, which affects the pace at which the tasks are learned, and hence hinders the training process [134].

Fig. 16 illustrates a case where the loss of the classification dominates the overall gradient.

Solutions. The most common solution is *Task Weighting* which balances the loss terms by an additional hyper-parameter as the weighting factor. The hyper-parameter is selected using a validation set. Naturally, increasing the number of tasks, as in the case of two-stage detectors, will increase the number of weighting factors and the dimensions of the search space (note that there are four tasks in two-stage detectors and two tasks in one-stage detectors).

An issue arising from the multi-task nature is the possible range inconsistencies among different loss functions. For example, in *AP Loss*, smooth L1, which is in the logarithmic range (since the input to the loss is conventionally provided after applying a logarithmic transformation) with $[0, \infty)$, is used for regression while $\mathcal{L}_{AP} \in [0, 1]$. Another example is the *GIoU Loss* [122], which is in the $[-1, 1]$ range and used together with cross entropy loss. The authors set the weighting factor of *GIoU Loss* to 10 and regularization is exploited to balance this range difference and ensure balanced training.

Since it is more challenging to balance terms with different ranges, it is a better strategy to first make the ranges comparable.

A more prominent approach to combine classification and regression tasks is *Classification-Aware Regression Loss* (CARL) [30], which assumes that classification and regression tasks are correlated. To combine the loss terms, the regression loss is scaled by a coefficient determined by the (classification) confidence score of the bounding box:

$$L_{CARL}(x) = c'_i L_{smooth}(x), \quad (23)$$

where c'_i is a factor based on p_i , i.e., an estimation from the classification task. In this way, regression loss provides gradient signals to the classification branch as well, and therefore, this formulation improves localization of high-quality (prime) examples.

An important contribution of CARL is to employ the correlation between the classification and regression tasks. However, as we discuss in Section 7.2, this correlation should be investigated and exploited more extensively.

Recently, Chen *et al.* [67] showed that cumulative loss originating from cross entropy needs to be dynamically weighted since, when cross entropy loss is used, the contribution rate of individual loss component at each epoch can be different. To prevent this imbalance, the authors proposed *Guided Loss* which simply weights the classification component by considering the total magnitude of the losses as:

$$\frac{w_{reg} L_{Reg}}{L_{cls}}. \quad (24)$$

The motivation of the method is that regression loss consists of only foreground examples and is normalized only by number of foreground classes, therefore it can be used as a normalizer for the classification loss.

7.1 Comparative Summary

Currently, except for linear task weighting, there is no method suitable for all architectures alleviating objective imbalance, and unless the weights of the tasks are set accordingly, training diverges [67]. While many studies use equal weights for the regression and classification tasks [19], [22], it is also shown that appropriate linear weighting can lead to small improvements on the performance [131]. However, an in-depth analysis on objective imbalance is missing in the literature.

CARL [30] is a method to promote examples with higher IoUs, and in such a way, 1.6 percent relative improvement is achieved compared to prime sampling without CARL (i.e. from 37.9 to 38.5). Another method, Guided Loss [67], weights the classification loss dynamically to ensure balanced training. This method achieves a similar performance with the baseline RetinaNet, without using Focal Loss. However, their method does not discard the linear weighting, and they search for the optimal weight for different architectures. Moreover, the effect of Guided Loss is not clear for the two-stage object detectors, which conventionally employ cross entropy loss.

7.2 Open Issues

Open Issue. Currently, the most common approach is to linearly combine the loss functions of different tasks to obtain

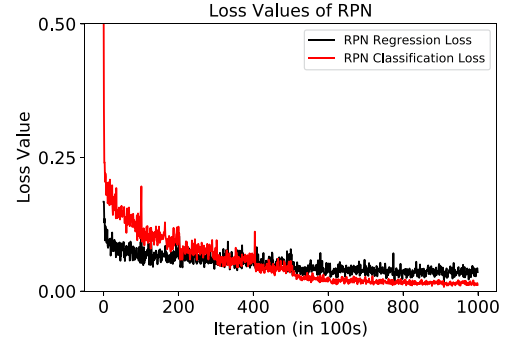


Fig. 17. Paces of the tasks in RPN (with Resnet-101 [57] backbone) [21] on Pascal VOC 2007 training set [51].

the overall loss function (except for classification-aware regression loss in [30]). However, as shown in Figs. 18a, 18b, 18c, as the input bounding box is slid over the image, both classification and regression losses are affected, implying their dependence. This suggests that current linear weighting strategy may not be able to address the imbalance of the tasks that is related to (i) the loss values and their gradients, and (ii) the paces of the tasks.

Explanation. The loss function of one task (i.e. classification) can affect the other task (i.e. regression). To illustrate, *AP Loss* [68] did not modify the regression branch; however, COCO style $AP@0.75$ increased around 3 percent. This example shows that the loss functions for different branches (tasks) are not independent (see also Fig. 18). This interdependence of tasks has been explored in classification-aware regression loss by Cao *et al.* [30] (as discussed in Section 7) to a certain extent. Further research is needed for a more detailed analysis of this interdependence and fully exploiting it for object detection.

Some studies in multi-task learning [134] pointed out that learning pace affects performance. With this in mind, we plotted the regression and classification losses of the RPN [21] during training on the Pascal VOC dataset [51] in Fig. 17. We observe that the classification loss decreases faster than the regression loss. Analyzing and balancing the learning paces of different tasks involved in the object detection problem might be another fruitful research direction.

8 OPEN ISSUES FOR ALL IMBALANCE PROBLEMS

In this section, we identify and discuss the issues relevant to all imbalance problems. For open issues pertaining to a specific imbalance problem, please see its corresponding section in the text.

8.1 A Unified Approach to Addressing Imbalance

Open Issue. One of the main challenges is to come up with a unified approach that addresses all imbalance problems by considering the inter-dependence between them.

Explanation. We illustrate this inter-dependency using a toy example in Fig. 18. In this figure, we shift an input bounding box with a high IoU (see Fig. 18a) to worse qualities in terms of IoU in two steps (see Figs. 18b, 18c) and observe how this shift affects the different imbalance problems. For the base case in Fig. 18a, there are two positive bounding boxes (relevant for class imbalance) with different scales

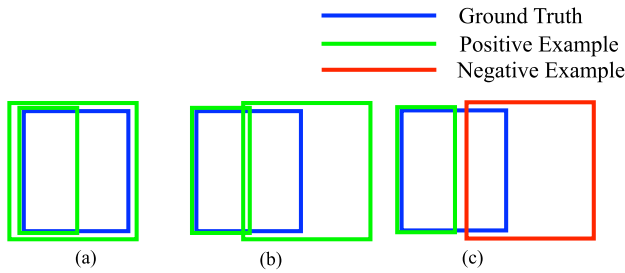


Fig. 18. An example suggesting the necessity of considering different imbalance problems together in a unified manner. Blue, green and red colors indicate ground-truth, positive example (prediction) and negative examples, respectively. The larger example (i.e. the one with higher IoU with the blue box) in (a) is shifted right. In (b), its IoU is significantly decreased and it eventually becomes a negative example in (c). This example shows the interplay between class imbalance, scale imbalance, spatial imbalance and objective imbalance by a change in BB positions.

(relevant for scale imbalance), loss values (relevant for objective imbalance) and IoUs (relevant for BB imbalance). Shifting the box to the right, we observe the following:

- In Fig. 18b, we still have two positives, both of which now have less IoU with the ground truth (compared to (a)).
This leads to the following: (i) There are more hard examples (considering hard example mining [24], [29]), and less prime samples [30]. For this reason, the methods for class imbalance are affected. (ii) The scales of the RoIs and ground truth do not change. Considering this, the scale imbalance seems not affected. (iii) Objective imbalance is affected in two ways: Firstly, the shifted BB will incur more loss (for regression, and possibly for classification) and thus, become more dominant in its own task. Secondly, since the cumulative individual loss values change, the contribution to the total loss of the individual loss values will also change, which implies its effect on objective imbalance. (iv) Finally, both BB IoU distribution and spatial distribution of the positive examples will be affected by this shift.
- In Fig. 18c, by applying a small shift to the same BB, its label changes.

This leads to the following: (i) There are less positive examples and more negative examples. The ground truth class loses an example. Note that this example evolves from being a hard positive to a hard negative in terms hard example mining [24], [29], and the criterion that the prime sample attention [30] considers for the example changed from IoU to classification score. For this reason, in this case, the methods involving class imbalance and foreground-foreground class imbalance are affected. (ii) A positive RoI is removed from the set of RoIs with similar scales. Therefore, there will be less positive examples with the same scale, which affects scale imbalance. (iii) Objective imbalance is affected in two ways: Firstly, the now-negative BB is an additional hard example in terms of classification possibly with a larger loss value. Secondly, the shifted example is totally free from the regression branch, and moved to the set of hard examples in terms of classification. Hence, the contribution

of the regression loss to the total loss is expected to decrease, and the contribution of classification would increase. (iv) Finally, the IoU distribution of the positive examples will be affected by this shift since a positive example is lost.

Therefore, the aforementioned imbalance problems have an intertwined nature, which needs to be investigated and identified in detail to effectively address all imbalance problems.

8.2 Measuring and Identifying Imbalance

Open Issue. Another critical issue that has not been addressed yet is how to quantify or measure imbalance, and how to identify imbalance when there is one. We identify three questions that need to be studied:

- 1) *What is a balanced distribution for a property that is critical for a task?* This is likely to be uniform distribution for many properties like, e.g. class distribution. However, different modalities may imply a different concept of balance. For example, OHEM prefers a skewed distribution around 0.5 for the IoU distribution; left-skewed for the positives and right-skewed for the negatives.
- 2) *What is the desired distribution for the properties that are critical for a task?* Note that the desired distribution may be different from the balanced distribution since skewing the distribution in one way may be beneficial for faster convergence and better generalization. For example, online hard negative mining [24] favors a right-skewed IoU distribution towards 0.5 [29], whereas prime sample attention prefers the positive examples with larger IoUs [30] and the class imbalance methods aim to ensure a uniform distribution from the classes.
- 3) *How can we quantify how imbalanced a distribution is?* A straightforward approach is to consider optimal transport measures such as the Wasserstein distance; however, such methods would neglect the effect of a unit change (imbalance) in the distribution on the overall performance, thereby jeopardizing a direct and effective consideration (and comparison) of the imbalance problems using the imbalance measurements.

8.3 Labeling a Bounding Box as Positive or Negative

Open Issue. Currently, object detectors use IoU-based thresholding (possibly with different values) for labeling an example as positive or negative and there is no consensus on this (i.e. Fast R-CNN [17], Retina Net [22] and RPN [21] label input BBs with IoUs 0.5, 0.4 and 0.3 as negatives respectively.). However, a consensus on this is critical since labeling is very relevant to determining whether an example is a hard example.

Explanation. Labeling bounding boxes is highly relevant to imbalance problems since this is the step where the set of all bounding boxes are split as positives and negatives in an online manner. Of special concern are the bounding boxes around the decision boundary, which are typically considered as hard examples, and a noisy labeling over them would result in large gradients in opposite directions. In

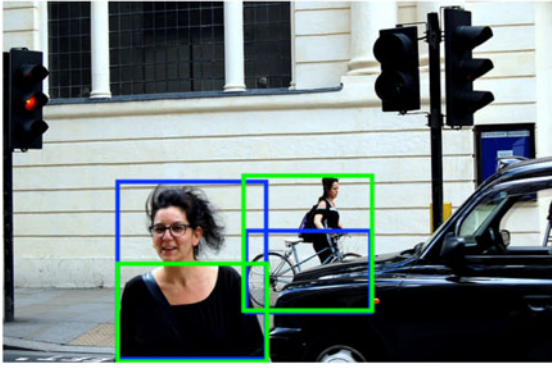


Fig. 19. An illustration on ambiguities resulting from labeling examples. The blue boxes denote the ground truth. The green boxes are the estimated positive boxes with $IoU > 0.5$. The input image is from the MS COCO [54].

other words, in order to define the hard negatives reliably, the number of outliers should be as small as possible. For this reason, consistent labeling of the input bounding boxes as positive or negative is a prerequisite of the imbalance problems in object detection.

Currently the methods use a hard IoU threshold (generally 0.5) to split the examples; however, Li *et al.* [65] showed that this scheme results in a large number of noisy examples. In Fig. 19, we illustrate two input bounding boxes that can be misleadingly labeled as positive; and once they are labeled as positive, it is likely that they will be sampled as hard positives:

- The estimated (positive) box for the bicycle (green) has two problems: It has occlusion (for the bicycle), and a big portion of it includes another object (a person). For this reason, during training, this is not only a hard example for the bicycle class but also a misleading example for the person class in that this specific example will try to suppress the probability of this box to be classified as person.
- The estimated (positive) box for the person class (green) consists of black pixels in most of it. In other words, the box does hardly include any visually descriptive part for a person. For this reason, this is a very hard example which is likely to fail in capturing the ground truth class well.

8.4 Imbalance in Bottom-Up Object Detectors

Open Issue. Bottom-up detectors [23], [48], [49] adopt a completely different approach to object detection than the one-stage and the two-stage detectors (see Section 2.1). Bottom-up detectors might share many of the imbalance problems seen in the top-down detectors, and they may have their own imbalance issues as well. Further research needs to be conducted for (i) analyzing the known methods addressing imbalance problems in the context of bottom-up object detectors, and (ii) imbalance problems that are specific to bottom-up detectors.

Explanation. Addressing imbalance issues in bottom-up object detectors has received limited attention. CornerNet [23] and ExtremeNet [49] use focal loss [22] to address foreground-background class imbalance, and the hourglass network [81] to compensate for the scale imbalance. On the

other hand, use of hard sampling methods and the effects of other imbalance problems have not been investigated. For the top-down detectors, we can recap some of the findings: from the class imbalance perspective, Shrivastava *et al.* [24] show that the examples with larger losses are important; from the scale imbalance perspective, different architectures [29], [96], [107] and training methods [27], [28] involving feature and image pyramids are proven to be useful and finally from the objective imbalance perspective, Pang *et al.* [29] showed that smooth $L1$ loss underestimates the effect of the inliers. Research is needed to come up with such findings for bottom-up object detectors.

9 CONCLUSION

In this paper, we provided a thorough review of the imbalance problems in object detection. In order to provide a more complete and coherent perspective, we introduced a taxonomy of the problems as well as the solutions for addressing them. Following the taxonomy on problems, we discussed each problem separately in detail and presented the solutions with a unifying yet critical perspective.

In addition to the detailed discussions on the studied problems and the solutions, we pinpointed and presented many open issues and imbalance problems that are critical for object detection. In addition to the many open aspects that need further attention for the studied imbalance problems, we identified new imbalance issues that have not been addressed or discussed before.

With this review and our taxonomy functioning as a map, we, as the community, can identify where we are and the research directions to be followed to develop better solutions to the imbalance problems in object detection.

ACKNOWLEDGMENTS

This work was supported in part by the Scientific and Technological Research Council of Turkey (TÜBİTAK) through the project titled “Object Detection in Videos with Deep Neural Networks” (grant number 117E054). The work of Kemal Öksüz was supported by the TÜBİTAK 2211-A National Scholarship Programme for Ph.D. students. Sinan Kalkan and Emre Akbas contributed equally to this work.

REFERENCES

- [1] Q. Fan, L. Brown, and J. Smith, “A closer look at faster R-CNN for vehicle detection,” in *Proc. IEEE Intell. Veh. Symp.*, 2016, pp. 124–129.
- [2] Z. Fu, Y. Chen, H. Yong, R. Jiang, L. Zhang, and X. Hua, “Foreground gating and background refining network for surveillance object detection,” *IEEE Trans. Image Process.*, vol. 28, no. 12, pp. 6077–6090, Dec. 2019.
- [3] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? The kitti vision benchmark suite,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 3354–3361.
- [4] X. Dai, “Hybridnet: A fast vehicle detection system for autonomous driving,” *Signal Process.: Image Commun.*, vol. 70, pp. 79–88, 2019.
- [5] P. F. Jaeger *et al.*, “Retina u-net: Embarrassingly simple exploitation of segmentation supervision for medical object detection,” *Mach. Learn. Health Workshop NeurIPS*, 2019.
- [6] S.-G. Lee, J. S. Bae, H. Kim, J. H. Kim, and S. Yoon, “Liver lesion detection from weakly-labeled multi-phase ct volumes with a grouped single shot multibox detector,” in *Proc. Int. Conf. Med. Image Comput. Comput. Assisted Interv.*, 2018, pp. 693–701.

- [7] M. Rad and V. Lepetit, "BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 3848–3856.
- [8] W. Kehl, F. Manhardt, F. Tombari, S. Ilıc, and N. Navab, "SSD-6D: Making rgb-based 3d detection and 6d pose estimation great again," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 1530–1538.
- [9] B. Tekin, S. N. Sinha, and P. Fua, "Real-time seamless single shot 6D object pose prediction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 292–301.
- [10] T. Hodan et al., "BOP: Benchmark for 6D object pose estimation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 19–35.
- [11] G. Du, K. Wang, and S. Lian, "Vision-based robotic grasping from object localization, pose estimation, grasp detection to motion planning: A review," 2019, *arXiv:1905.06658*.
- [12] I. Bozcan and S. Kalkan, "Cosmo: Contextualized scene modeling with boltzmann machines," *Robot. Auton. Syst.*, vol. 113, pp. 132–148, 2019.
- [13] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1627–1645, Sep. 2010.
- [14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
- [15] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 6517–6525.
- [16] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 580–587.
- [17] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 1440–1448.
- [18] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: Object detection via region-based fully convolutional networks," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 379–387.
- [19] W. Liu et al., "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 21–37.
- [20] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 779–788.
- [21] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.
- [22] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. Int. Conf. Comput. Vis.*, 2017, pp. 2999–3007.
- [23] H. Law and J. Deng, "Cornersnet: Detecting objects as paired keypoints," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 642–656.
- [24] A. Shrivastava, A. Gupta, and R. Girshick, "Training region-based object detectors with online hard example mining," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 761–769.
- [25] W. Ouyang, X. Wang, C. Zhang, and X. Yang, "Factors in finetuning deep model for object detection with long-tail distribution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 864–873.
- [26] T. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 936–944.
- [27] B. Singh and L. S. Davis, "An analysis of scale invariance in object detection - snip," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3578–3587.
- [28] B. Singh, M. Najibi, and L. S. Davis, "Sniper: Efficient multi-scale training," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 9333–9343.
- [29] J. Pang, K. Chen, J. Shi, H. Feng, W. Ouyang, and D. Lin, "Libra R-CNN: Towards balanced learning for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 821–830.
- [30] Y. Cao, K. Chen, C. C. Loy, and D. Lin, "Prime Sample Attention in Object Detection," 2019, *arXiv:1904.04821*.
- [31] L. Liu et al., "Deep learning for generic object detection: A survey," *Int. J. Comput. Vis.*, vol. 128, pp. 261–318, 2020.
- [32] Z. Zou, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," 2018, *arXiv:1905.05055*.
- [33] S. Agarwal, J. O. D. Terrail, and F. Jurie, "Recent advances in object detection in the age of deep convolutional neural networks," 2018, *arXiv:1809.03193*.
- [34] Z. Sun, G. Bebis, and R. Miller, "On-road vehicle detection: a review," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 5, pp. 694–711, May 2006.
- [35] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 4, pp. 743–761, Apr. 2012.
- [36] S. Zafeiriou, C. Zhang, and Z. Zhang, "A survey on face detection in the wild," *Comput. Vis. Image Understanding*, vol. 138, pp. 1–24, 2015.
- [37] Q. Ye and D. Doermann, "Text detection and recognition in imagery: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 7, pp. 1480–1500, Jul. 2015.
- [38] X. Yin, Z. Zuo, S. Tian, and C. Liu, "Text detection, tracking and recognition in video: A comprehensive survey," *IEEE Trans. Image Process.*, vol. 25, no. 6, pp. 2752–2773, Jun. 2016.
- [39] G. Litjens et al., "A survey on deep learning in medical image analysis," *Medical Image Anal.*, vol. 42, pp. 60–88, 2017.
- [40] B. Krawczyk, "Learning from imbalanced data: Open challenges and future directions," *Progress Artif. Intell.*, vol. 5, no. 4, pp. 221–232, 2016.
- [41] J. L. Leevy, T. M. Khoshgoftaar, R. A. Bauder, and N. Seliya, "A survey on addressing high-class imbalance in big data," *J. Big Data*, vol. 5, no. 42, 2018.
- [42] A. Fernández, S. García, M. Galar, R. Prati, B. Krawczyk, and F. Herrera, *Learning From Imbalanced Data Sets*. Berlin, Germany: Springer, 2018.
- [43] J. M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," *J. Big Data*, vol. 6, no. 21, 2019, Art. no. 27.
- [44] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, and A. W. M. Smeulders, "Selective search for object recognition," *Int. J. Comput. Vis.*, vol. 104, no. 2, pp. 154–171, 2013.
- [45] C. L. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 391–405.
- [46] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, "DSSD: Deconvolutional single shot detector," 2017, *arXiv:1701.06659*.
- [47] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," 2018, *arXiv:1804.02767*.
- [48] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "Centernet: Keypoint triplets for object detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 6569–6578.
- [49] X. Zhou, J. Zhuo, and P. Krahenbuhl, "Bottom-up object detection by grouping extreme and center points," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 850–859.
- [50] A. Newell, Z. Huang, and J. Deng, "Associative embedding: End-to-end learning for joint detection and grouping," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 2277–2287.
- [51] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (VOC) challenge," *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, 2010.
- [52] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.
- [53] Y. He, C. Zhu, J. Wang, M. Savvides, and X. Zhang, "Bounding box regression with uncertainty for accurate object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2888–2897.
- [54] T.-Y. Lin et al., "Microsoft COCO: common objects in context," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 740–755.
- [55] A. Kuznetsova et al., "The open images dataset V4: unified image classification, object detection, and visual relationship detection at scale," *Int. J. Comput. Vis.*, 2020.
- [56] S. Shao et al., "Objects365: A large-scale, high-quality dataset for object detection," in *Proc. Int. Conf. Comput. Vis.*, 2019, pp. 8429–8438.
- [57] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.
- [58] H. A. Rowley, S. Baluja, and T. Kanade, "Human face detection in visual scenes," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 1995, pp. 875–881.
- [59] K.-K. Sung and T. Poggio, "Example-based learning for view-based human face detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 1, pp. 39–51, Jan. 1998.
- [60] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2001, pp. I-1.
- [61] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2005, pp. 886–893.

- [62] T. Kong, F. Sun, A. Yao, H. Liu, M. Lu, and Y. Chen, "Ron: Reverse connection with objectness prior networks for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5244–5252.
- [63] S. Zhang, L. Wen, X. Bian, Z. Lei, and S. Z. Li, "Single-shot refinement neural network for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4203–4212.
- [64] J. Nie, R. M. Anwer, H. Cholakkal, F. S. Khan, Y. Pang, and L. Shao, "Enriched feature guided refinement network for object detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 9536–9545.
- [65] B. Li, Y. Liu, and X. Wang, "Gradient harmonized single-stage detector," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 8577–8584.
- [66] J. Chen, D. Liu, B. Luo, X. Peng, T. Xu, and E. Chen, "Residual objectness for imbalance reduction," 2019, *arXiv:1908.09075*.
- [67] J. Chen *et al.*, "Is sampling heuristics necessary in training deep object detectors?" 2019, *arXiv:1909.04868*.
- [68] K. Chen *et al.*, "Towards accurate one-stage object detection with ap-loss," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5114–5122.
- [69] Q. Qian, L. Chen, H. Li, and R. Jin, "DR Loss: Improving Object Detection by Distributional Ranking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020.
- [70] Y. Song, A. Schwing, Richard, and R. Urtasun, "Training deep neural networks via direct loss minimization," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 2169–2177.
- [71] P. Henderson and V. Ferrari, "End-to-end training of object class detectors for mean average precision," in *Proc. Asian Conf. Comput. Vis.*, 2016, pp. 198–213.
- [72] X. Wang, A. Shrivastava, and A. Gupta, "A-fast-rcnn: Hard positive generation via adversary for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 3039–3048.
- [73] D. Dwibedi, I. Misra, and M. Hebert, "Cut, paste and learn: Surprisingly easy synthesis for instance detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 1301–1310.
- [74] N. Dvornik, J. Mairal, and C. Schmid, "Modeling visual context is key to augmenting object detection datasets," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 375–391.
- [75] S. Tripathi, S. Chandra, A. Agrawal, A. Tyagi, J. M. Rehg, and V. Chari, "Learning to generate synthetic data via compositing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 461–470.
- [76] H. Wang, Q. Wang, F. Yang, W. Zhang, and W. Zuo, "Data augmentation for object detection via progressive and selective instance-switching," 2019, *arXiv:1906.00358*.
- [77] K. Oksuz, B. C. Cam, S. Kalkan, and E. Akbas, "Generating positive bounding boxes for balanced training of object detectors," in *Proc. IEEE Winter Appl. Comput. Vis.*, 2020, pp. 894–903.
- [78] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 1–9.
- [79] K. Oksuz, B. C. Cam, E. Akbas, and S. Kalkan, "Localization recall precision (LRP): A new performance metric for object detection," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 521–537.
- [80] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Representations*, 2015, pp. 1–14.
- [81] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 483–499.
- [82] A. G. Howard *et al.*, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.
- [83] Z. Li, C. Peng, G. Yu, X. Zhang, Y. Deng, and J. Sun, "Detnet: Design backbone for object detection," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 334–350.
- [84] F. Yang, W. Choi, and Y. Lin, "Exploit all the layers: Fast and accurate CNN object detector with scale dependent pooling and cascaded rejection classifiers," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2129–2137.
- [85] Z. Cai, Q. Fan, R. Feris, and N. Vasconcelos, "A unified multi-scale deep convolutional neural network for fast object detection," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 354–370.
- [86] J. Li, X. Liang, S. Shen, T. Xu, J. Feng, and S. Yan, "Scale-aware fast R-CNN for pedestrian detection," *IEEE Trans. Multimedia*, vol. 20, no. 4, pp. 985–996, Apr. 2018.
- [87] L. Zhu *et al.*, "Bidirectional feature pyramid network with recurrent attention residual modules for shadow detection," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 122–137.
- [88] Y. Sun, P. S. K. P. J. Shimamura, and A. Sagata, "Concatenated feature pyramid network for instance segmentation," in *Proc. IEEE 5th Int. Conf. Multimedia Big Data*, 2019, pp. 297–301.
- [89] S. S. Seferbekov, V. I. Iglovikov, A. V. Buslaev, and A. A. Shvets, "Feature pyramid network for multi-class land segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2018, pp. 272–2723.
- [90] A. Kirillov, R. B. Girshick, K. He, and P. Dollár, "Panoptic feature pyramid networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 6399–6408.
- [91] E. H. Adelson, C. H. Anderson, J. R. Bergen, P. J. Burt, and J. M. Ogden, "Pyramid methods in image processing," *RCA Engineer*, vol. 29, no. 6, pp. 33–41, 1984.
- [92] Y. Pang, T. Wang, R. M. Anwer, F. S. Khan, and L. Shao, "Efficient featurized image pyramid network for single shot detector," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 7328–7336.
- [93] Y. Li, Y. Chen, N. Wang, and Z. Zhang, "Scale-aware trident networks for object detection," in *Proc. Int. Conf. Comput. Vis.*, 2019, pp. 6053–6062.
- [94] J. Noh, W. Bae, W. Lee, J. Seo, and G. Kim, "Better to follow, follow to be better: Towards precise supervision of feature super-resolution for small object detection," in *Proc. Int. Conf. Comput. Vis.*, 2019, pp. 9724–9733.
- [95] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *Proc. Int. Conf. Learn. Representations*, 2016, pp. 1–13.
- [96] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8759–8768.
- [97] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7794–7803.
- [98] P. Zhou, B. Ni, C. Geng, J. Hu, and Y. Xu, "Scale-transferrable object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 528–537.
- [99] G. Huang, Z. Liu, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2261–2269.
- [100] S.-W. Kim, H.-K. Kook, J.-Y. Sun, M.-C. Kang, and S.-J. Ko, "Parallel feature pyramid network for object detection," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 239–256.
- [101] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 346–361.
- [102] T. Kong, F. Sun, W. Huang, and H. Liu, "Deep feature pyramid reconfiguration for object detection," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 172–188.
- [103] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7132–7141.
- [104] H. Li, Y. Liu, W. Ouyang, and X. Wang, "Zoom out-and-in network with map attention decision for region proposal and object detection," *Int. J. Comput. Vis.*, vol. 127, no. 3, pp. 225–238, 2019.
- [105] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.
- [106] Q. Zhao *et al.*, "M2Det: A single-shot object detector based on multi-level feature pyramid network," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 9259–9266.
- [107] G. Ghiasi, T. Lin, R. Pang, and Q. V. Le, "NAS-FPN: learning scalable feature pyramid architecture for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 7029–7038.
- [108] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," in *Proc. Int. Conf. Learn. Representations*, 2017, pp. 1–16.
- [109] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8697–8710.
- [110] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le, "Regularized evolution for image classifier architecture search," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 4780–4789.
- [111] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6105–6114.
- [112] Y. Chen, T. Yang, X. Zhang, G. Meng, C. Pan, and J. Sun, "Detnas: Backbone search for object detection," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 6642–6652.

- [113] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. Lecun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," in *Proc. Int. Conf. Learn. Representations.*, 2014, pp. 1–16.
- [114] P. J. Huber, "Robust estimation of a location parameter," *Ann. Statist.*, vol. 53, no. 1, pp. 73–101, 1964.
- [115] B. Jiang, R. Luo, J. Mao, T. Xiao, and Y. Jiang, "Acquisition of localization confidence for accurate object detection," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 784–799.
- [116] E. Goldman, R. Herzig, A. Eisenschlat, J. Goldberger, and T. Hassner, "Precise detection in densely packed scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5222–5231.
- [117] Z. Li, Z. Xie, L. Liu, B. Tao, and W. Tao, "Iou-uniform R-CNN: Breaking through the limitations of rpn," 2019, *arXiv:1912.05190*.
- [118] J. Choi, D. Chun, H. Kim, and H.-J. Lee, "Gaussian YOLOv3: An accurate and fast object detector using localization uncertainty for autonomous driving," in *Proc. Int. Conf. Comput. Vis.*, 2019, pp. 502–511.
- [119] Z. Tan, X. Nie, Q. Qian, N. Li, and H. Li, "Learning to rank proposals for object detection," in *Proc. Int. Conf. Comput. Vis.*, 2019, pp. 8272–8280.
- [120] J. Yu, Y. Jiang, Z. Wang, Z. Cao, and T. Huang, "Unitbox: An advanced object detection network," in *The ACM International Conference on Multimedia*, 2016, pp. 516–520.
- [121] L. Tychsen-Smith and L. Petersson, "Improving object localization with fitness NMS and bounded iou loss," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 6877–6885.
- [122] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 658–666.
- [123] Z. Zheng, P. Wang, W. Liu, J. Li, Y. Rongguang, and R. Dongwei, "Distance-IoU loss: Faster and better learning for bounding box regression," in *Proc. AAAI Conf. Artif. Intell.*, 2020.
- [124] Z. Cai and N. Vasconcelos, "Cascade R-CNN: Delving into high quality object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 6154–6162.
- [125] S. Gidaris and N. Komodakis, "Object detection via a multi-region and semantic segmentation-aware CNN model," in *Proc. Int. Conf. Comput. Vis.*, 2015, pp. 1134–1142.
- [126] S. Gidaris and N. Komodakis, "Attend refine repeat: Active box proposal generation via in-out localization," in *Proc. Brit. Mach. Vis. Conf.*, 2016, pp. 1–90.
- [127] J. Cao, Y. Pang, J. Han, and X. Li, "Hierarchical shot detector," in *Proc. Int. Conf. Comput. Vis.*, 2019, pp. 9704–9713.
- [128] J. Wang, K. Chen, S. Yang, C. C. Loy, and D. Lin, "Region proposal by guided anchoring," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2960–2969.
- [129] J. Dai et al., "Deformable convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 764–773.
- [130] X. Zhang, F. Wan, C. Liu, R. Ji, and Q. Ye, "Freeanchor: Learning to match anchors for visual object detection," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 147–155.
- [131] K. Chen et al., "MMDetection: Open mmlab detection toolbox and benchmark," 2019, *arXiv:1906.07155*.
- [132] T. Yang, X. Zhang, Z. Li, W. Zhang, and J. Sun, "Metaanchor: Learning to detect objects with customized anchors," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 320–330.
- [133] C. Peng et al., "Megdet: A large mini-batch object detector," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 6181–6189.
- [134] M. Guo, A. Haque, D.-A. Huang, S. Yeung, and L. Fei-Fei, "Dynamic task prioritization for dynamic task learning," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 282–299.
- [135] H. Xu, L. Yao, W. Zhang, X. Liang, and Z. Li, "Auto-fpn: Automatic network architecture adaptation for object detection beyond classification," in *Proc. Int. Conf. Comput. Vis.*, 2019, pp. 6649–6658.



Kemal Oksuz received the BSc degree in system engineering from the Land Forces Academy of Turkey, in 2008, and the MSc degree (with high honor degree) in computer engineering, in 2016, from Bogazici University. He is currently working toward the PhD in Middle East Technical University, Ankara, Turkey. His research interest includes computer vision with a focus on object detection.



Baris Can Cam received the BSc degree in electrical and electronics engineering from Eskisehir Osmangazi University, Turkey, in 2016. He is currently working toward the MSc degree in Middle East Technical University, Ankara, Turkey. His research interest includes computer vision with a focus on object detection.



Sinan Kalkan received the MSc degree in computer engineering from Middle East Technical University (METU), Turkey, in 2003, and the PhD degree in informatics from the University of Göttingen, Germany, in 2008. After working as a postdoctoral researcher at the University of Göttingen and at METU, he became an assistant professor at METU, in 2010. Since 2016, he has been working as an associate professor on problems within computer vision and developmental robotics.



Emre Akbas received the BSc and MSc degrees in computer engineering from Middle East Technical University (METU), and the PhD degree in electrical and computer engineering from the University of Illinois at Urbana-Champaign. He is currently an assistant professor at the Department of Computer Engineering, METU. Prior to joining METU, he was a postdoctoral research associate at the Department of Psychological and Brain Sciences, University of California, Santa Barbara. His research interests include computer vision and machine learning with a focus on object detection and human pose estimation.

► For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/csdl.