

IDENTIFYING AND ADDRESSING IMBALANCE PROBLEMS IN VISUAL
DETECTION

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

KEMAL ÖKSÜZ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
COMPUTER ENGINEERING

MAY 2021

Approval of the thesis:

**IDENTIFYING AND ADDRESSING IMBALANCE PROBLEMS IN VISUAL
DETECTION**

submitted by **KEMAL ÖKSÜZ** in partial fulfillment of the requirements for the degree of **Doctor of Philosophy in Computer Engineering Department, Middle East Technical University** by,

Prof. Dr. Halil Kalıpçılar
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. Halit Oğuztüzün
Head of Department, **Computer Engineering**

Assoc. Prof. Dr. Sinan Kalkan
Supervisor, **Computer Engineering, METU**

Assist. Prof. Dr. Emre Akbaş
Co-supervisor, **Computer Engineering, METU**

Examining Committee Members:

Assist. Prof. Dr. Ramazan Gökberk Cinbiş
Computer Engineering, METU

Assoc. Prof. Dr. Sinan Kalkan
Computer Engineering, METU

Assoc. Prof. Dr. Erkut Erdem
Computer Engineering, Hacettepe University

Assist. Prof. Dr. Hande Alemdar
Computer Engineering, METU

Assist. Prof. Dr. Hacer Yalım Keleş
Computer Engineering, Hacettepe University

Date: 06.05.2021



I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname: Kemal Öksüz

Signature :

ABSTRACT

IDENTIFYING AND ADDRESSING IMBALANCE PROBLEMS IN VISUAL DETECTION

Öksüz, Kemal

Ph.D., Department of Computer Engineering

Supervisor: Assoc. Prof. Dr. Sinan Kalkan

Co-Supervisor: Assist. Prof. Dr. Emre Akbaş

May 2021, 291 pages

This thesis has two aims: **(Aim 1)** Identifying imbalance problems in visual detection, and **(Aim 2)** addressing these problems using loss functions based on performance measures. For Aim 1, we present a comprehensive review of the imbalance problems in object detection including a problem-based taxonomy and a detailed discussion for each problem with its solutions and open issues. To achieve Aim 2, we identify two challenges: (i) Average Precision (AP), the common performance measure, has certain drawbacks. To remedy them, we propose Localisation Recall Precision (LRP) Error as a novel performance measure. (ii) Loss functions derived from performance measures are ranking-based functions whose derivatives are zero or infinite, thus, they cannot directly be used with backpropagation. To overcome this, based on perceptron learning, we propose Identity Update, a simple and general optimisation method for ranking-based losses, which provably ensures balance in terms of total gradient magnitudes of positives and negatives. Having addressed these challenges, using LRP Error and Identity Update, we propose average LRP Loss and Rank & Sort (RS) Loss for balanced training of visual detectors. We show that our loss functions have the

following unique benefits: (i) They are easy-to-tune with a single hyper-parameter, different from common methods with ~ 7 hyper-parameters on average, (ii) they enforce correlation among sub-tasks of visual detectors (i.e. classification and different localisation tasks), which affects both the remaining detections after Non-Maximum-Suppression and performance measure AP, and (iii) they are applicable to a diverse set of visual detectors (i.e. one-stage, multi-stage, anchor-based, anchor-free, with balanced or severely imbalanced data). As a result of these benefits, for example with RS Loss, we train four object detection and three instance segmentation methods only by tuning the learning rate and consistently improve their performance.

Keywords: Visual detection, segmentation, object detection, performance measure, Average Precision, loss function, optimisation method, ranking, sorting

ÖZ

GÖRSEL TESPİTTEKİ DENGESİZLİK PROBLEMLERİNİN BELİRLENMESİ VE ÇÖZÜMLENMESİ

Öksüz, Kemal

Doktora, Bilgisayar Mühendisliği Bölümü

Tez Yöneticisi: Doç. Dr. Sinan Kalkan

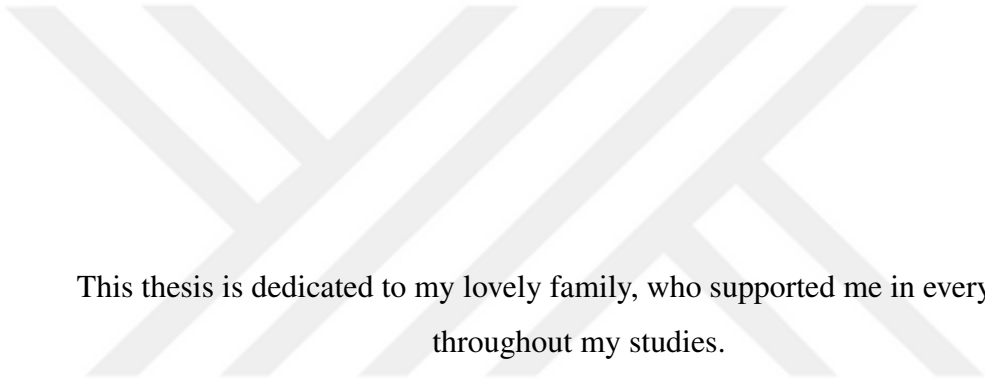
Ortak Tez Yöneticisi: Dr. Öğr. Üyesi Emre Akbaş

Mayıs 2021, 291 sayfa

Bu tezin iki ana amacı vardır: (**Amaç 1**) Görsel tespitteki dengesizlik problemlerini belirleme ve (**Amaç 2**) bu problemleri performans metriklerine dayanan kayıp fonksiyonları ile çözümü. Amaç 1 için nesne tespit görevindeki dengesizlik problemleri için problem-tabanlı bir sınıflandırma ve her bir problem için yöntem ve açık noktaları ile birlikte detaylı bir tartışma içeren bir inceleme sunuyoruz. Amaç 2'ye ulaşmak için iki zorluk belirliyoruz: (i) Yaygın performans metriği olan AP'nin belirli sakıncaları bulunmaktadır. Bunlara çare olan yeni bir performans metriği olarak Localisation Recall Precision (LRP) Hatasını öneriyoruz. (ii) Performans metriklerinden türetilen kayıp fonksiyonları türevleri sıfır ya da sonsuz olan sıralama-tabanlı fonksiyonlardır ve geri yayılım ile doğrudan kullanılamazlar. Bunu aşmak için, perceptron öğrenmeye dayanarak, sıralama-tabanlı kayıp fonksiyonları için basit ve genel bir optimizasyon metodu olan, pozitif ve negatiflerin toplam gradyan büyüklükleri açısından ispatlanabilir bir denge sağlayan Identity Update'i öneriyoruz. Bu zorlukları çözümledikten sonra, LRP Hatası ve Identity Update'i kullanarak, görsel nesne tespit edicilerin den-

geli eğitimi için average LRP (aLRP) ve Rank & Sort (RS) kayıplarını öneriyoruz. Kayıp fonksiyonlarımızın şu eşsiz faydaları sağladığını gösteriyoruz: (i) Ortalama ~ 7 hiper-parametreye sahip yaygın metotlardan farklı olarak, tek hiper-parametre ile ayarlanmaları kolaydır, (ii) Maksimum-Olmayarı-Bastırma ve performans ölçüsü AP üzerinde etkisi olan görsel tespit edicilerin alt görevleri (sınıflandırma ve farklı konumlandırma görevleri) arasındaki korelasyonu sağlamaktadırlar ve (iii) birçok farklı metoda (tek aşamalı, çok aşamalı, çapa-tabanlı, çapasız, dengeli veya dengesiz veri ile) uygulanabilirler. Bu faydalarının sonucunda, örneğin RS kaybımız ile sadece öğrenme oranını ayarlayarak dört nesne tespit edici ve üç bölütleme metodunu eğitiyor ve performanslarını tutarlı olarak artırıyoruz.

Anahtar Kelimeler: Görsel tespit, bölütleme, nesne tespiti, performans metriği, Ortalama Kesinlik, kayıp fonksiyonu, optimizasyon yöntemi, sıralama



This thesis is dedicated to my lovely family, who supported me in every respect throughout my studies.

ACKNOWLEDGMENTS

Firstly, I would like to express my sincere gratitude to my advisors Dr. Sinan Kalkan and Dr. Emre Akbaş for their continuous support of my Ph.D. study and related research, for their patience, motivation, and immense knowledge. Their guidance supported and motivated me all the time during my research. I really feel lucky to have them as advisors and could not have imagined having better advisors and mentors.

Besides my advisor, I would like to thank the rest of my thesis committee, Dr. Ramazan Gökberk Cinbiş, Dr. Erkut Erdem, Dr. Hande Alemdar and Dr. Hacer Yalın Keleş for their insightful comments and encouragement.

I thank my fellow Barış Can Cam for stimulating discussions, for sleepless nights we spent together especially around deadlines and for his contribution and positive attitude; which, I believe, allowed us to collaborate in great harmony during our studies.

I thank TÜBİTAK to support me by the TÜBİTAK 2211-A National Scholarship Programme for Ph.D. students during my research. The numerical calculations reported in this thesis are mainly performed on the resources of TÜBİTAK ULAKBİM High Performance and Grid Computing Center (TRUBA) and Roketsan Missiles Inc. Therefore, I also thank these organizations for their support. TÜBİTAK also partially supported this work under the Grants 117E054 and 120E494.

Last but not the least, I would like to thank my family: I am grateful to my parents, who have encouraged me for higher level and better education throughout my life. My beloved wife, Seda, has one of the biggest shares in the accomplishment of this thesis. Despite the long hours I reserved to my studies almost every day for more than four years, she was always supportive and motivating. Without her encouraging attitude, I may not have completed this thesis. And our little one, Güneş, has been a gift since the second year of my studies. It was a pleasure for me to study for this thesis, and I wish Güneş to find something he will enjoy pursuing as well.

TABLE OF CONTENTS

ABSTRACT	v
ÖZ	vii
ACKNOWLEDGMENTS	x
TABLE OF CONTENTS	xi
LIST OF TABLES	xxi
LIST OF FIGURES	xxv
LIST OF ABBREVIATIONS	xxix
CHAPTERS	
1 INTRODUCTION	1
1.1 Motivation and Problem Definition	4
1.2 Scope of the Thesis	6
1.2.1 Identifying Imbalance Problems in Visual Detection	6
1.2.2 Addressing Imbalance Problems by Loss Functions based on Performance Measures	9
1.3 Contributions and Novelties	13
1.3.1 Publications	14
1.3.2 Software Contributions	15
1.4 The Outline of the Thesis	16

2	FREQUENTLY USED NOTATIONS AND DEFINITIONS	17
2.1	Frequently Used Notations	17
2.2	Frequently Used Terms	20
3	IMBALANCE PROBLEMS IN OBJECT DETECTION	25
3.1	A Taxonomy of the Imbalance Problems and their Solutions in Object Detection	28
3.2	Imbalance 1: Class Imbalance	31
3.2.1	Foreground-Background (a.k.a. Positive-Negative) Class Imbalance	33
3.2.1.1	Hard Sampling Methods	34
3.2.1.2	Soft Sampling Methods	36
3.2.1.3	Sampling-Free Methods	39
3.2.1.4	Generative Methods	39
3.2.2	Foreground-Foreground (a.k.a. Positive-Positive) Class Imbalance	41
3.2.2.1	Foreground-Foreground Imbalance Owing to the Dataset	42
3.2.2.2	Foreground-Foreground Imbalance Owing to the Batch	43
3.2.3	Comparative Summary	44
3.2.4	Open Issues	45
3.2.4.1	Sampling More Useful Examples	45
3.2.4.2	Foreground-Foreground Class Imbalance Problem	46
3.2.4.3	Foreground-Foreground Imbalance Owing to the Batch	47
3.2.4.4	Ranking-Based Loss Functions	47
3.3	Imbalance 2: Scale Imbalance	48

3.3.1	Object/Box-Level Scale Imbalance	49
3.3.1.1	Methods Predicting from the Feature Hierarchy of Backbone Features	50
3.3.1.2	Methods Based on Feature Pyramids	51
3.3.1.3	Methods Based on Image Pyramids	52
3.3.1.4	Methods Combining Image and Feature Pyramids	53
3.3.2	Feature-level Imbalance	54
3.3.2.1	Methods Using Pyramidal Features as a Basis	55
3.3.2.2	Methods Using Backbone Features as a Basis	57
3.3.3	Comparative Summary	60
3.3.4	Open Issues	61
3.3.4.1	Characteristics of Different Layers of Feature Hierarchies	62
3.3.4.2	Image Pyramids in Deep Object Detectors	63
3.4	Imbalance 3: Spatial Imbalance	63
3.4.1	Imbalance in Regression Loss	63
3.4.2	IoU Distribution Imbalance	68
3.4.3	Object Location Imbalance	71
3.4.4	Comparative Summary	73
3.4.5	Open Issues	74
3.4.5.1	A Regression Loss with Many Aspects	74
3.4.5.2	Analyzing the Loss Functions	74
3.4.5.3	Designing Better Anchors	75
3.4.5.4	Relative Spatial Distribution Imbalance	76

3.4.5.5	Imbalance in Overlapping BBs	77
3.4.5.6	Analysis of the Orientation Imbalance	77
3.5	Imbalance 4: Objective Imbalance	77
3.5.1	Comparative Summary	80
3.5.2	Open Issues	80
3.6	Open Issues for All Imbalance Problems	81
3.6.1	A Unified Approach to Addressing Imbalance	82
3.6.2	Measuring and Identifying Imbalance	84
3.6.3	Labeling a Bounding Box as Positive or Negative	84
3.6.4	Imbalance in Bottom-Up Object Detectors	85
3.7	Conclusion	86
4	BOUNDING BOX GENERATOR TO ANALYSE IMBALANCE PROBLEMS IN VISUAL DETECTION TASKS	89
4.1	Introduction	89
4.2	Related Work	92
4.3	The Generators	94
4.3.1	Bounding Box Generator	94
4.3.1.1	Determining Feasible Space for the Desired IoU	96
4.3.1.2	Controlling the Relative Spatial Distribution of the Boxes	97
4.3.2	pRoI Generator: Training by Generated BBs	98
4.4	Experimental Setup	100
4.5	Imbalance Problems and Analysis of RPN RoIs	100
4.5.1	IoU Distribution Imbalance	100

4.5.2	Foreground-Foreground Class Imbalance	102
4.5.3	Effect of Online Hard Positive Mining	102
4.5.4	Relative Spatial Imbalance	103
4.6	Practical Improvements	104
4.6.1	Online Foreground Balanced Sampling	105
4.6.2	Generating More Samples in Higher IoUs	106
4.7	Conclusion	109
5	LOCALISATION RECALL PRECISION (LRP) ERROR FOR EVALUATING VISUAL DETECTION TASKS	111
5.1	Introduction	111
5.1.1	Important features for a performance measure	112
5.1.2	Limitations of AP	114
5.1.3	Motivation for our Localisation Recall Precision (LRP) Error	115
5.1.4	Other alternatives to AP	115
5.1.5	Contributions of the Chapter	116
5.1.6	Outline of the Chapter	117
5.2	Related Work	117
5.3	Average Precision	119
5.3.1	Definition of AP	120
5.3.2	An Analysis of AP	120
5.4	Panoptic Quality	123
5.4.1	Definition of PQ	123
5.4.2	An Analysis of PQ	123

5.5	Localisation-Recall-Precision (LRP) Error	125
5.5.1	LRP: The Performance Metric	126
5.5.2	An Analysis of LRP	127
5.5.3	Optimal LRP (oLRP): Evaluating and Thresholding Soft Predictions	129
5.5.4	Potential Extensions of LRP	130
5.6	A Comparison of LRP with AP and PQ	131
5.7	Experimental Evaluation	133
5.7.1	Evaluated Models, Datasets and Performance Measures	135
5.7.2	Evaluating Soft Predictions on Object Detection, Keypoint Detection and Instance Segmentation Tasks	136
5.7.2.1	Analysis with respect to Completeness	140
5.7.2.2	Analysis with respect to Interpretability	141
5.7.2.3	Analysis with respect to Practicality	143
5.7.3	Evaluating Hard Predictions on Panoptic Segmentation Task	146
5.7.3.1	Analysis with respect to Interpretability	146
5.7.3.2	Analysis with respect to Practicality	146
5.7.4	Thresholding Visual Object Detectors	147
5.8	Conclusion	150
6	IDENTITY UPDATE: USING ERROR-DRIVEN UPDATE WITH BACK-PROPAGATION TO OPTIMISE RANKING-BASED LOSS FUNCTIONS	151
6.1	Previous Work: Definition, Computation and Optimisation of AP Loss	152
6.2	Identity Update to Compute and Optimise Ranking-based Loss Functions	155
6.3	A Case Study: Optimising AP Loss by Identity Update	160

7	AVERAGE LOCALISATION-RECALL-PRECISION LOSS: A RANKING-BASED, BALANCED LOSS FUNCTION UNIFYING CLASSIFICATION AND LOCALISATION IN OBJECT DETECTION	161
7.1	Introduction	162
7.2	Related Work	163
7.3	Average Localisation-Recall-Precision (aLRP) Loss	164
7.3.1	Optimisation of the aLRP Loss	165
7.3.2	A Self-Balancing Extension for the Localisation Task	167
7.4	Experiments	168
7.4.1	Ablation Study	168
7.4.2	More insight on aLRP Loss	170
7.4.3	Comparison with State of the Art (SOTA)	171
7.4.4	Using aLRP Loss with Different Object Detectors	172
7.5	Conclusion	174
8	RANK & SORT LOSS FOR OBJECT DETECTION AND INSTANCE SEGMENTATION	175
8.1	Introduction	175
8.2	Related Work	178
8.3	Rank & Sort Loss	179
8.4	Using RS Loss to Train Visual Detectors	181
8.4.1	Dataset and Implementation Details	181
8.4.2	Analysis and Tuning-Free Design Choices	181
8.4.3	Training Different Architectures	183
8.5	Experiments	185

8.5.1	Experiments on Object Detection	185
8.5.1.1	Multi-stage Object Detectors	185
8.5.1.2	One-stage Object Detectors	187
8.5.1.3	Comparison with SOTA	190
8.5.2	Experiments on Instance Segmentation	190
8.5.2.1	Multi-stage Instance Segmentation Methods	191
8.5.2.2	One-stage Instance Segmentation Methods	191
8.5.2.3	Comparison with SOTA	194
8.6	Conclusion	194
9	CONCLUSION	197
9.1	Summary	197
9.2	Discussion	198
9.3	Limitations and Future Work	200
	REFERENCES	203
	APPENDICES	
A	IMBALANCE PROBLEMS IN OTHER DOMAINS	229
A.1	Image Classification	229
A.2	Metric Learning	232
A.3	Multi-Task Learning	235
B	DETAILS OF THE BOUNDING BOX GENERATOR	237
B.1	Details of Finding the Feasible Space for Top-Left Point	237
B.2	Details of Finding the Feasible Space for Bottom-Right Point	238

C	PROOF THAT PQ ERROR IS NOT A METRIC	241
D	PROOF THAT LRP ERROR IS A METRIC	243
E	WEIGHTING THE COMPONENTS OF THE LRP ERROR FOR PRACTICAL NEEDS OF DIFFERENT APPLICATIONS	245
F	WHY AVERAGE LRP (ALRP) IS NOT AN IDEAL PERFORMANCE MEASURE?	247
G	THE SIMILARITY BETWEEN PQ AND LRP ERRORS	249
H	MORE EXPERIMENTS WITH LRP ERROR	251
	H.1 A Use-Case of LRP-Optimal Thresholds in Video Object Detection	251
	H.2 Analysing Latency of LRP Computation	255
	H.3 Analyzing the Effect of TP Validation Threshold	256
I	A GENERALISATION OF ERROR-DRIVEN OPTIMISATION FOR RANKING-BASED LOSSES	259
J	DEFINING AND OPTIMISING ALRP LOSS USING IDENTITY UPDATE	263
K	DETAILS OF ALRP LOSS	265
	K.1 A Soft Sampling Perspective for aLRP Localisation Component	265
	K.2 The Relation between aLRP Loss Value and Total Gradient Magnitudes	266
	K.3 Self-balancing the Gradients Instead of the Loss Value	267
L	ADDITIONAL EXPERIMENTS WITH ALRP LOSS	269
	L.1 More Ablation Experiments: Using Self Balance and GIoU with AP Loss	269
	L.2 Anchor Configuration	270
	L.3 Using a Wrong Target for the Primary Term in the Error-driven Update Rule	271
	L.4 Implementation Details for FoveaBox and Faster R-CNN	271

M	DETAILS OF RS LOSS	273
M.1	Derivation of the Gradients	273
M.2	More Insight on RS Loss Computation and Gradients on an Example	275
N	COMPARATIVE ANALYSIS OF RS LOSS AND ALRP LOSS	279
O	THE RELATION OF RS LOSS WITH LRP ERROR	283
P	MORE EXPERIMENTS WITH RS LOSS	287
P.1	Effect of smoothing unit-step function, the single hyper-parameter, for RS Loss.	287
P.2	Training Cascade R-CNN with RS Loss	287
P.3	Hyper-parameters of R-CNN Variants in Table 8.2	288
P.4	Using different localisation qualities as continuous labels to super- vise instance segmentation methods	289
P.5	Training time comparison and inference time of methods	290
P.6	Detailed Results	290

LIST OF TABLES

TABLES

Table 1.1	Number of hyper-parameters in the loss functions of the state-of-the-art (SOTA) one-stage and two-stage methods.	7
Table 1.2	Imbalance problems reviewed in this thesis.	10
Table 3.1	Toy example depicting common hard and soft sampling methods. . .	37
Table 3.2	Comparison of major generative methods addressing class imbalance.	40
Table 3.3	A list of widely used loss functions for the BB regression task. . . .	65
Table 4.1	Effect of the batch properties for generated positive samples for different RoI sources.	101
Table 4.2	Comparison of OFB Sampling with random sampling on PASCAL VOC.	105
Table 4.3	Comparison of OFB Sampling with random sampling and online hard positive mining on COCO.	105
Table 4.4	Performance Comparison of pRoI Generator with RPN on PASCAL VOC.	107
Table 4.5	Effect of <i>RoI Num</i> parameter of pRoI Generator.	108
Table 5.1	A comparison of LRP and PQ on example detectors.	124
Table 5.2	Comparison of AP, PQ and LRP in terms of desired properties. . . .	134

Table 5.3	The repositories of models that we downloaded, evaluated and utilized for comparison of performance measures.	136
Table 5.4	AP and oLRP performances of several methods for soft-prediction tasks (i.e. object detection, keypoint detection and instance segmentation) on COCO	137
Table 5.5	AP and oLRP performances of several object detectors for “person” and “broccoli” classes on COCO.	138
Table 5.6	Detector-level performance results of panoptic segmentation methods as hard predictions.	145
Table 7.1	Ablation analysis of aLRP Loss on COCO.	169
Table 7.2	Analysis of self-balancing for aLRP Loss.	170
Table 7.3	Analysis of initialisation of self-balancing.	170
Table 7.4	Effect of correlating rankings. AP^C denotes COCO-style AP.	170
Table 7.5	Comparison of aLRP Loss with the SOTA detectors on COCO <i>test-dev</i>	173
Table 7.6	Comparison of aLRP Loss on anchor-free FoveaBox.	173
Table 7.7	Comparison of aLRP Loss on two-stage Faster R-CNN.	174
Table 8.1	Comparison of instance- and task-level weighting methods on RS-ATSS to determine design choices of RS Loss.	183
Table 8.2	Comparison of our RS-R-CNN with different multi-stage object detection methods on COCO.	186
Table 8.3	The performance of RS-R-CNN with balanced and imbalanced data.	188
Table 8.4	Comparison of our RS Loss with different loss functions robust to imbalance on one-stage object detection methods on COCO.	189

Table 8.5	Comparison of RS Loss with SOTA for object detection on COCO <i>test-dev</i>	190
Table 8.6	Comparison of RS Loss on Mask R-CNN on COCO.	192
Table 8.7	Comparison of RS Loss on YOLACT on COCO.	193
Table 8.8	Comparison of RS Loss on SOLOv2 on COCO.	194
Table 8.9	Comparison of RS Loss with SOTA for instance segmentation on COCO <i>test-dev</i>	195
Table B.1	Top-Left space bounds and equations for Bounding Box Generator. .	237
Table B.2	Bottom-Right space bounds for Bounding Box Generator.	239
Table B.3	Bottom-Right space equations for Bounding Box Generator.	239
Table H.1	Comparison of LRP-Optimal thresholds with different thresholding approaches.	254
Table L.1	More ablation experiments on aLRP Loss.	270
Table N.1	An analysis depicting the impact of the range pressure in self bal- ancing of aLRP Loss.	280
Table N.2	An analysis of ranking-based weighting of aLRP Loss.	282
Table P.1	Effect of δ_{RS} on RS Loss.	287
Table P.2	Performance comparison of RS Loss on Cascade R-CNN.	288
Table P.3	Analysis of using different continuous labels for instance segmen- tation problem on YOLACT.	289
Table P.4	Average iteration time of methods trained by the standard loss vs. RS Loss.	290

Table P.5 Comprehensive performance results of models trained by RS-Loss on COCO.	291



LIST OF FIGURES

FIGURES

Figure 1.1	Different visual detection tasks.	2
Figure 1.2	A generic visual detection architecture.	3
Figure 1.3	An illustration depicting the effect of correlation on performance measure, AP and Non-Maximum Suppression.	8
Figure 3.1	Problem-based categorization of the methods used for imbalance problems.	29
Figure 3.2	The steps where the imbalance problems arise in the common training pipeline of object detectors.	30
Figure 3.3	Number of papers per imbalance problem category through years.	32
Figure 3.4	Illustration of the class imbalance problems.	32
Figure 3.5	Some statistics of common datasets (training sets) to present foreground-foreground class imbalance.	42
Figure 3.6	Illustration of batch-level class imbalance.	48
Figure 3.7	Imbalance in scales of the objects in common datasets.	49
Figure 3.8	An illustration and comparison of the solutions for scale imbalance.	50
Figure 3.9	Illustration of Feature-Level imbalance.	55

Figure 3.10	High-level diagrams of the methods designed for feature-level imbalance.	56
Figure 3.11	An illustration of imbalance in regression loss.	64
Figure 3.12	An analysis on IoU distribution imbalance.	69
Figure 3.13	Distribution of the centers of the objects in the common datasets over the normalized image.	72
Figure 3.14	The (relative) spatial distributions of $1K$ RoIs.	75
Figure 3.15	An illustration of the imbalance in overlapping BBs.	76
Figure 3.16	An illustration of objective imbalance.	78
Figure 3.17	Paces of the tasks in RPN.	81
Figure 3.18	An example suggesting the necessity of considering different imbalance problems together in a unified manner.	82
Figure 3.19	An illustration on ambiguities resulting from labeling examples.	86
Figure 4.1	Illustrations of Bounding Box Generator and Positive RoI Generator.	90
Figure 4.2	An example bounding box generation using Bounding Box Generator.	95
Figure 4.3	$1K$ generated boxes using Bounding Box Generator.	96
Figure 4.4	Splitting the regions around top-left and bottom-right point.	97
Figure 4.5	IoU distribution of different RoI Sources.	101
Figure 4.6	Relative spatial distribution of 2,500 RPN RoIs TL points.	104
Figure 5.1	Three different object detection results for an image from COCO with very different PR curves but the same AP	113

Figure 5.2	An illustration that shows how a transition from a FP to TP is handled differently by PQ and LRP.	125
Figure 5.3	A visual comparison of AP and LRP.	128
Figure 5.4	How LRP Error, PR Error (i.e. $1 - (\text{Precision} \times \text{Recall})$) and PQ Error (i.e. $1 - \text{PQ}$) behave over different inputs.	131
Figure 5.5	The relation of LRP with precision error, recall error and PQ error.	132
Figure 5.6	Example PR curves for four classes.	139
Figure 5.7	The effect of interpolation on AP.	143
Figure 5.8	Class-level LRP_{FP} vs. LRP_{FN} comparison.	144
Figure 5.9	s-LRP curves of different object detectors for the “person” class.	148
Figure 5.10	The distributions of the class-specific LRP-Optimal Thresholds for different methods.	149
Figure 6.1	Three-step computation (green arrows) and optimisation (orange arrows) algorithms of ranking-based loss functions.	156
Figure 7.1	A toy example to depict aLRP Loss enforces high-precision detections to have high-IoUs, while others do not.	163
Figure 7.2	An illustration to present aLRP Loss assigns gradients to each branch based on the outputs of both branches.	165
Figure 7.3	aLRP Loss and its components.	167
Figure 7.4	Gradient and loss comparison of aLRP Loss	171
Figure 8.1	A ranking-based classification loss vs our RS Loss.	176
Figure 8.2	Training visual detectors using RS Loss.	184

Figure C.1	A counter-example presenting PQ Error (i.e., 1-PQ) violates triangle inequality	241
Figure H.1	Example PR curves of the methods on three example classes. . .	253
Figure H.2	The effect of τ on oLRP and its components.	256
Figure L.1	The effect of using a wrong target for the primary term in the error-driven update rule.	271
Figure M.1	An example case to illustrate the computation of RS Loss. . . .	276
Figure M.2	An example case to illustrate the computation of primary terms in RS Loss.	277

LIST OF ABBREVIATIONS

aLRP	Average Localisation Recall Precision
AP	Average Precision (see notation in Chapter 2 for different AP variants)
BB	Bounding Box
CE	Cross Entropy
FL	Focal Loss
FN	False Negative
FP	False Positive
IoU	Intersection-over-Union
GIoU	Generalized Intersection-over-Union
GT	Ground Truth
LRP	Localisation Recall Precision (see notation in Chapter 2 for LRP components)
NMS	Non-Maximum-Suppression
oLRP	Optimal Localisation Recall Precision
PDQ	Probabilistic Detection Quality
POD	Probabilistic Object Detection
PQ	Panoptic Quality
RS	Rank & Sort
RoI	Region-of-Interest
SOTA	State-of-the-art
TP	True Positive
VD	Visual Detection
wrt.	with respect to



CHAPTER 1

INTRODUCTION

Visual detection is the simultaneous estimation of categories and locations of object instances in an image. The location of an object can be represented in different ways and these different ways are studied as different visual detection tasks (Fig. 1.1): e.g. “object detection” uses boxes bounding to localise the objects, “keypoint detection” aims to localise keypoints on objects, “instance segmentation” methods represent the location in the pixel-level via masks, and “panoptic segmentation”, an extension of “instance segmentation”, additionally requires the background classes in the image to be segmented. Visual detection 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 the detection was cast as a machine learning problem, the first generation 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) [15]. After the extremely influential work by Krizhevsky et al. in 2012 [16], deep learning (or deep neural networks) has started to dominate various problems in computer vision and visual detection was no exception. The current generation 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 benchmark dataset (PASCAL VOC), while the DPM [15] achieved 0.34 Average Precision (AP), current deep learning based visual detectors achieve around 0.80 mAP [17].

In general, a deep visual detector (a.k.a. visual detection method, visual detector)

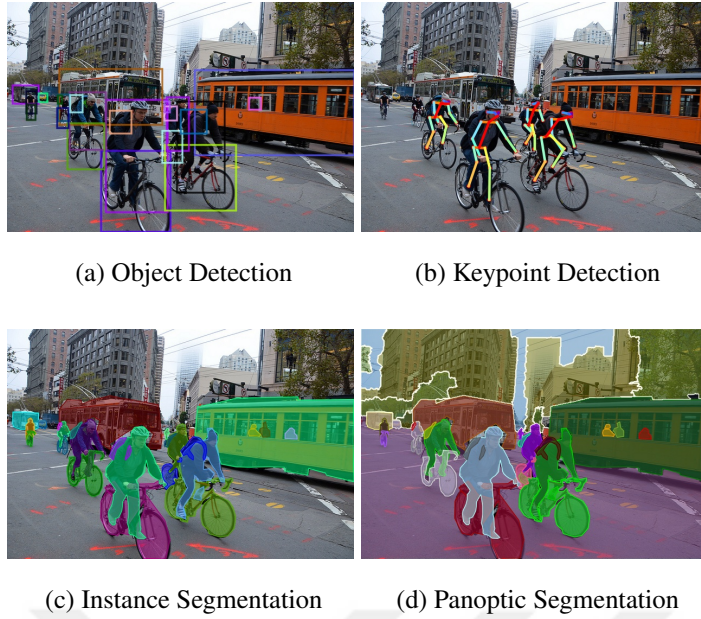


Figure 1.1: Different visual detection tasks use different representations to localise the objects. Keypoint detection is illustrated for the “person” class. The image is taken from the COCO dataset [13]. Detectron2 [14] is used to plot the ground truths.

comprises of a backbone network to extract features from the image, followed by at least one detection network consisting of heads (a.k.a. sub-networks) to generate the required outputs for the corresponding visual detection task (Fig. 1.2). To illustrate, deep object detectors essentially adopt classification and box regression heads and instance segmentation methods at least have classification and mask prediction heads. The methods conventionally classify and predict the location of a large number (e.g. more than $150k$ anchors per image in RetinaNet [18] with images of size 1333×800) of proposals (we use “proposal” to indicate all types of object hypotheses, e.g. anchor, point, region-of-interest, see Chapter 2 for definitions) placed on the image in order to capture objects in different locations and scales. To determine the label of each proposal for supervision, each proposal is first scored against every ground truth object based on a similarity function (e.g. Intersection-over-Union – IoU), then using a simple assignment heuristic, each proposal is labelled as “positive (foreground) example” or “negative (background) example” (or “ignored” during training in some cases [18]). Due to this large number of proposals, the methods typically necessitate a non-maximum suppression (NMS) operation to discard highly overlapping predic-

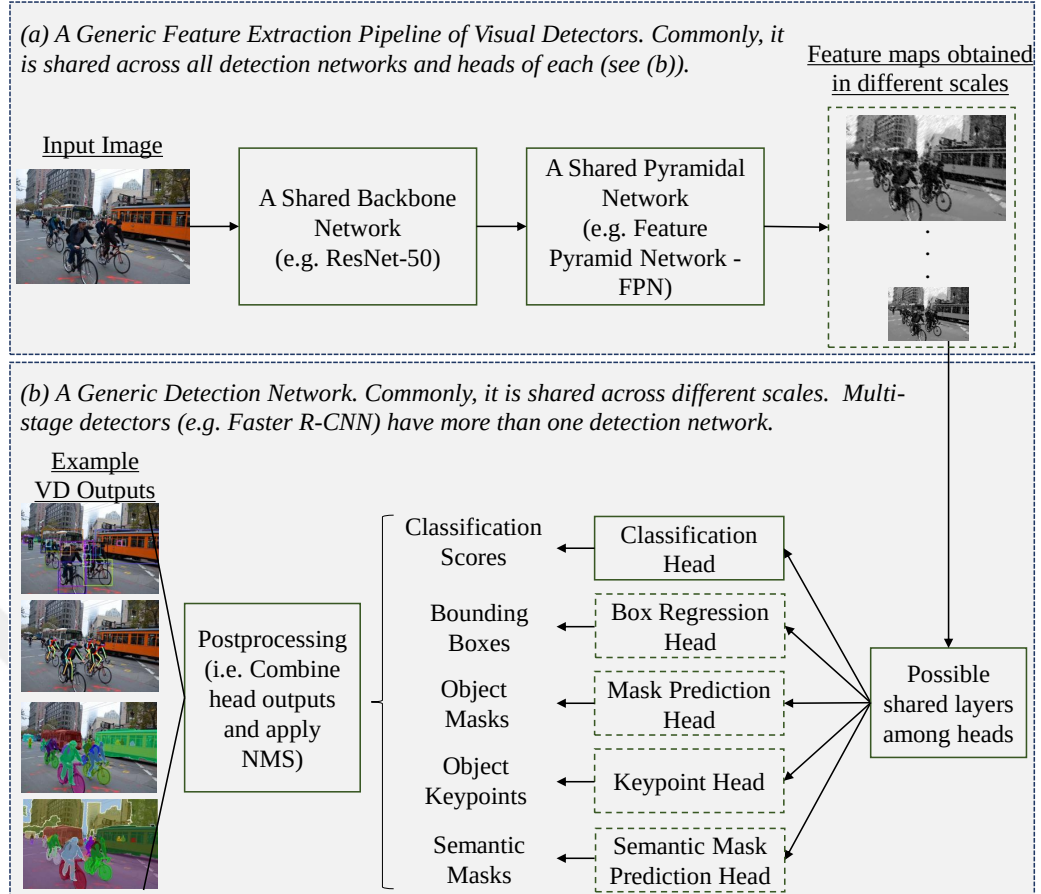


Figure 1.2: A generic visual detection architecture consists of a shared feature extraction network and at least one detection network. (a) A generic feature extraction network. In order to extract features, shared backbone is commonly augmented by a FPN-based network in order to obtain multi-scale feature maps from the image, improving the detection performance across objects with different scales. (b) A generic detection network. It is generally shared across feature maps with different scales and consists of (i) some possible shared parameters across different heads, (ii) different heads required to generate necessary output for the corresponding VD problem (solid box: included in all common VD pipelines, dashed: used depending on the VD problem and architecture), and (iii) a post-processing step including the combination of the outputs of different heads (i.e. also includes discarding detections classified as background possibly by thresholding classification scores) and NMS, the suppression of the duplicate detections, to obtain final VD output. While the architecture in the figure is sufficient for one-stage detectors, multi-stage detectors have multiple detection networks, but still use a single shared backbone.

tions detecting the same object.

Common visual detectors can be split into two as multi-stage detectors and one-stage detectors. Multi-stage detectors, such as Faster R-CNN [19] and Cascade R-CNN [20], aim to decrease the large number of negative examples resulting from the predefined dense proposals to a manageable size by using an initial detection mechanism [19] which determines the regions where the objects most likely appear, called Region of Interests (RoIs). These RoIs are further processed by subsequent detection network/networks in order to output the classification and localisation output for each objects. Differently, one-stage visual detectors are designed to predict the detection results directly from proposals without any initial elimination stage.

It is a well-known fact that the number of positive examples are much smaller than the number of negative examples during the training of visual detectors (e.g. the rate of positives to negatives is 0.001 for RetinaNet – Chapter 3), ensuing a significant imbalance between positive and negative classes to be addressed during training. While multi-stage visual detectors alleviate this imbalance by resorting to sampling a balanced batch of positives and negatives among proposals in both stages during training, one-stage visual detectors rely on loss functions (e.g. Focal Loss [18]) robust to imbalanced data. In this thesis, our focus is the imbalance problems in visual detection tasks, and the next section discusses why there is still a need to identify and address these problems.

1.1 Motivation and Problem Definition

In the last five years, although the major driving force of progress in visual detection has been the incorporation of deep neural networks [18, 19, 21, 22, 23, 24, 25, 26], imbalance problems in visual detection at several levels have also received significant attention [27, 28, 29, 30, 31, 32, 33]. Despite the fact that almost every visual detection paper uses the terms related to balanced training (e.g. balance, imbalance, bias etc.), these terms can be employed to refer to different types of imbalance problems. For example, while a pioneer method, Fast R-CNN [22], refers to objective imbalance by “the balance between the two task losses”, Focal Loss [18] focuses on class imbal-

ance and SNIPER [31] is motivated from the difference in the scales of the objects. On the other hand, a systematic identification and categorisation of these imbalance problems for any visual detection task is missing in the literature.

The common approach to address these imbalance problems is to design robust loss functions¹ of the form:

$$\mathcal{L}_{VD} = \sum_{k \in \mathcal{K}} \sum_{t \in \mathcal{T}} \lambda_t^k \mathcal{L}_t^k, \quad (1.1)$$

which combines $\mathcal{L}_t^k$, the loss function for task t on stage k (e.g. $|\mathcal{K}| = 2$ for Faster R-CNN [19] with RPN and R-CNN – Fig. 1.2), weighted by a hyper-parameter λ_t^k . However, this loss formulations has certain drawbacks:

- (D1) The number of hyper-parameters can easily exceed 10 with additional hyper-parameters arising from task-specific imbalance problems (Table 1.1), e.g. the positive-negative imbalance in the classification task, and if a multi-stage architecture is used (e.g. HTC [34] employs 3 R-CNNs with different λ_t^k). Thus, although such loss functions have led to unprecedented successes in several benchmarks, they necessitate tuning, which is time consuming, leads to sub-optimal solutions and makes fair comparison of methods challenging.
- (D2) This loss formulations consider each task independently and does not correlate them (e.g. classification, box regression and mask prediction), and hence, does not guarantee high-quality localisation for high-precision examples, which have a positive effect on both NMS and performance evaluation measure (Fig. 1.3).
- (D3) Despite designed for the similar detection tasks (i.e. multi-stage detectors can be represented as a cascaded one-stage detectors – Fig. 1.2), multi-stage and one-stage detectors address imbalance differently: while almost all existing R-CNN variants (e.g. Faster R-CNN [19], Mask R-CNN [35], Cascade R-CNN [20], Dynamic R-CNN [36] etc.) employ additional sampling stages in all stages followed by a standard cross entropy loss; one-stage detectors generally relies on Focal Loss [18] and discard sampling. As a result, a single method to train all visual detectors and achieve SOTA performance is missing in the literature.

¹ As we will see in Chapter 3, sampling in multi-stage detectors can also be formulated as part of the loss function.

1.2 Scope of the Thesis

Based on the observations mentioned in Section 1.1, this thesis has two overarching aims:

- **Aim 1.** Identifying imbalance problems in visual detection,
- **Aim 2.** Addressing imbalance problems by loss functions based on performance measures.

Following sections presents an overview of our research under the scope of each aim.

1.2.1 Identifying Imbalance Problems in Visual Detection

Having motivated from the gap in the literature, we first define an imbalance problem:

Definition 1. *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.*

To illustrate, the most commonly known imbalance problem in object detection is the positive-negative (a.k.a. foreground-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. Besides, due to the multi-task nature of visual detectors and the data including objects in different scales, poses etc., different imbalance problems affect the performance of visual detectors besides positive-negative imbalance. Considering these peculiarities of visual detection, in Chapter 3, we present a comprehensive review of the imbalance problems in object detection, a representative task among visual detection tasks, including a detailed discussion on the solutions and open issues for each problem. In our review, we identify eight different imbalance

²We assume that in both stages, vanilla Faster R-CNN, Mask R-CNN and Mask Scoring R-CNN employ L1 Loss with no hyper-parameter following recent practice. However, Dynamic R-CNN and Libra R-CNN propose their own regression losses, hence we followed original papers.

Table 1.1: Number of hyper-parameters in the loss functions of the state-of-the-art (SOTA) one-stage and two-stage methods. The method with the least number of hyper-parameter is AP Loss, which optimises Average Precision Loss, and it has 3 hyper-parameters since AP Loss and Smooth L1 are each with 1 hyper-parameter, and they are combined with a single balancing scalar (λ_t^k in Eq .1.1). As for two-stage methods, the most simple-to-tune method is the vanilla Faster R-CNN with 3 λ_t^k s and 4 additional hyper-parameters to define the number of positives and negatives to be sampled, hence 7 in total². Overall, while the methods in the table requires ~ 7.4 hyper-parameters to be tuned on average, our aLRP Loss and RS Loss have significantly less number of hyper-parameters to train both types of architectures.

Type	Method	Number of Hyper-parameters
One-stage Methods	AP Loss [37]	3
	Retina Net [18]	4
	ATSS [38]	5
	FreeAnchor [39]	8
	Center Net [40]	10
	aLRP Loss (Ours)	1
	RS Loss (Ours)	1
Two-stage Methods	Faster R-CNN [19]	7
	Mask R-CNN [35]	8
	Mask Scoring R-CNN [41]	9
	Dynamic R-CNN [36]	10
	Libra R-CNN [32]	11
	aLRP Loss (Ours)	3
	RS Loss (Ours)	3

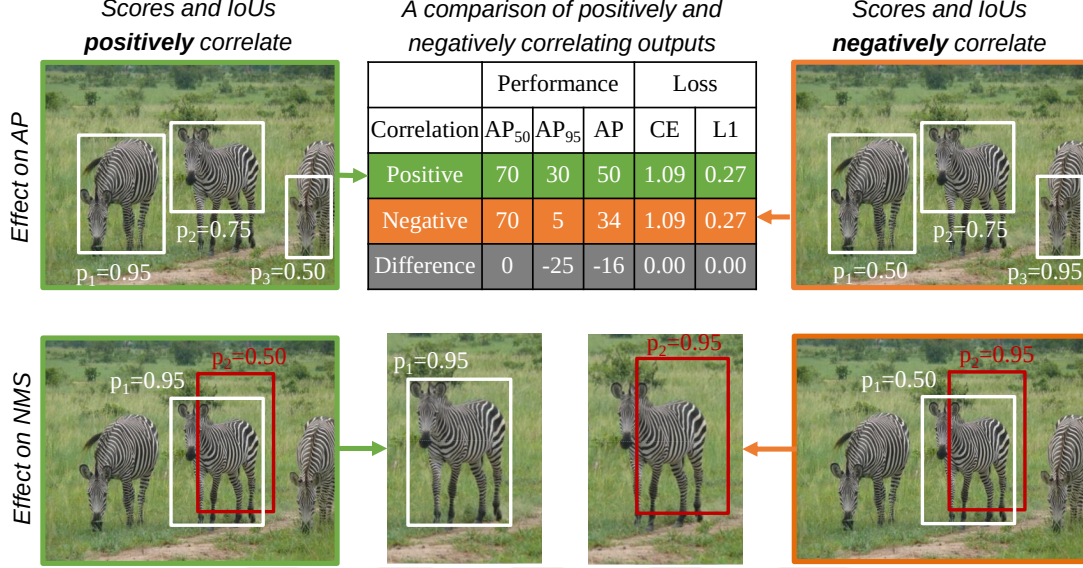


Figure 1.3: An illustration depicting the effect of correlation on performance measure, AP and Non-Maximum Suppression (NMS). (top) A larger correlation between prediction scores and IoUs of the final detections (i.e. possibly after NMS) improves the performance value, COCO-style AP (AP in short). The correlation has an effect on especially in larger IoUs and the resulting AP. AP₅₀ and AP₇₅ refer to APs when the true positives are validated from IoU 0.50 and 0.75 respectively. (down) A larger correlation between prediction scores and IoUs of the detections before NMS improves detection quality since the detection with the largest confidence score survives from the conventional NMS among detections with significant overlap. More specifically, for the corresponding zebra in the example, while the white detection with a better localisation quality survives from NMS when the scores and IoUs positively correlate; the surviving detection is the red one with poor localisation when they negatively correlate. Note that the table in top also depicts that the conventional loss formulation following Eq. 1.1 ignores correlation.

problems and group these problems in a taxonomy with four main types: class imbalance, scale imbalance, spatial imbalance and objective imbalance (Table 1.2). **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 positive-negative imbalance, there is also imbalance among the positive (foreground) 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 object detection (e.g. classification and box regression losses).

With the aim to enable further analysis of imbalance problems in visual detection, in Chapter 4, we propose a “Bounding Box Generator” that can generate a random bounding box given an input bounding box and minimum desired IoU value. Considering that using bounding box-based proposals is quite common in visual detector tasks [18, 42, 43], our Bounding Box Generator is a basis for the generation of different proposal sets with different distributional characteristics. To illustrate this, we design a positive region-of-interest generator to train the second stage of Faster R-CNN and analyse three different imbalance problems.

1.2.2 Addressing Imbalance Problems by Loss Functions based on Performance Measures

Our main motivation to alleviate these imbalance problems is to devise loss functions based on performance measures (e.g. AP), thereby bridging the gap between training and evaluation objectives. With the following three potential advantages, we motivate that our idea can address the three drawbacks (D1)-(D3) of the conventional loss formulation (Eq. 1.1) we identified in Section 1.1:

- (A1) As the performance measures (e.g. AP) do not typically have any hyper-parameters (or even if they have, they have to be fixed for all methods before training for a fair performance evaluation across methods), the resulting loss function is

Table 1.2: Imbalance problems reviewed in this thesis (Chapter 3). 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 (“Type”) shows the major imbalance categories and the second column shows the imbalance problem and its associated input property concerning the definition of the imbalance problem.

Type	Imbalance Problem: Related Input Property
Class	Foreground-Background Class Imbalance: The numbers of proposals pertaining to different classes (Section 3.2.1)
	Foreground-Foreground Class Imbalance: The numbers of proposals pertaining to different classes (Section 3.2.2)
Scale	Object/box-level Scale Imbalance (Section 3.3.1):The scales of input and ground-truth bounding boxes
	Feature-level Imbalance (Section 3.3.2): Contribution of the feature layer from different abstraction levels of the backbone network (i.e. high and low level)
Spatial	Imbalance in Regression Loss (Section 3.4.1): Contribution of the individual examples to the regression loss
	IoU Distribution Imbalance (Section 3.4.2) :IoU distribution of positive proposals
	Object Location Imbalance (Section 3.4.3): Locations of the objects throughout the image
Objective	Objective Imbalance (Section 3.5): Contribution of different tasks (i.e. classification, regression) to the overall loss

expected to be simpler-to-tune (Table 1.1).

- (A2) A loss function based on a performance measure can either directly enforce correlation (as in COCO-style AP in 1.3) or else it can be designed in the desired way as long as the base performance metric considers all performance aspects of visual detector (i.e. False Positive Rate, False Negative Rate and Localisation Error).
- (A3) The performance measures are inherently robust to class-imbalance due to their ranking-based error definition, which penalises the negatives only when it is ranked above a positive. Thus, we expect that the loss functions with a ranking-based error definition can directly be used by both one-stage and multi-stage detectors.

With these motivations, in order to optimise a performance metric as a loss function, we first identify two important challenges:

- **Challenge 1.** The performance metric which the loss function is to base on needs to be chosen carefully. For example, as we will present in Section 5, the common performance metric, AP, does not precisely consider localisation quality (i.e. instead, AP considers the localisation quality loosely by applying thresholding on the localisation quality of each detection to validate it as a true positive.), and hence, optimising only AP to train a visual detector may not be the best solution.
- **Challenge 2.** Ranking-based nature of the performance metric yields null or infinite derivatives, preventing the direct usage of ranking-based loss functions with stochastic gradient descent using backpropagation.

Addressing Challenge 1. Having examined AP, we first identify that it has significant drawbacks: (i) AP does not precisely consider all performance aspects of object detectors (i.e. False Positive Rate, False Negative Rate and Localisation Error), (ii) the resulting AP value is difficult to interpret, and (iii) AP has some practical limitations which we detail in Chapter 5. To address these drawbacks, we propose Localisation-Recall-Precision (LRP) Error as a novel performance metric which yields a “complete” (i.e. LRP considers all performance aspects explicitly), an “interpretable” (i.e.

LRP has a component to present the detection quality of each performance aspect) and a “unified” (i.e. LRP can evaluate all visual detection tasks) evaluation of visual detectors. We use LRP and oLRP to evaluate 35 visual detectors from four different visual detection tasks (object detection, keypoint detection, instance segmentation and panoptic segmentation).

Addressing Challenge 2. In order to allow our LRP Error to be a basis for our loss functions, we build upon how Chen et al. [37] optimised AP Loss, the loss form of the AP, in which they incorporated error-driven update from Perceptron Learning [44] into backpropagation. Considering that their method is specific for AP Loss, in Chapter 6, we propose “Identity Update” as a general and simple computation and optimisation method for ranking-based loss functions. We prove that the ranking-based loss functions optimised using our Identity Update have provable balance in terms of gradient magnitudes between positives and negatives.

Our Ranking-based and Balanced Loss Functions. Finally, we propose two novel loss functions, which are defined based on our LRP Error and optimised using our Identity Update:

1. In Chapter 7, we present average Localisation-Recall-Precision (aLRP) Loss, which considers the outputs of both classification and box regression heads to supervise each head (i.e. classification and box regression), thereby enforcing correlation of these heads. In such a way AP Loss ensures ~ 5 AP improvement over the baseline ranking-based loss baseline, AP Loss [37], on Retina Net [18].
2. In Chapter 8, we propose Rank & Sort (RS) Loss, improving upon aLRP Loss:
 - (i) With its novel sorting objective, RS Loss directly aims to supervise the visual detector to sort the positive examples with respect to localisation qualities, which is consistent with NMS and computation of the performance measures.
 - (ii) RS Loss does not need longer training iterations unlike aLRP Loss and AP Loss. We apply RS Loss on a diverse set of seven visual detectors (both multi-stage and one-stage) only by tuning the learning rate, significantly simplify the training pipeline and consistently outperform both conventional and ranking baseline loss functions.

We note that as hypothesized, both of our loss functions (i) have only one hyper-parameter to be tuned for one-stage detectors, and three hyper-parameters for multi-stage detectors, which is significantly less than common methods (Table 1.1), (ii) enforces correlation on all heads, and (iii) can train both one-stage and multi-stage visual detectors following the same training pipeline.

1.3 Contributions and Novelties

Our contributions in this thesis are as follows:

- We propose a taxonomy of imbalance problems in object detection and provide a critical literature review of their solutions following our taxonomy.
- We propose “Bounding Box Generator” as a tool for analysing different imbalance problems. Using Bounding Box Generator, we devise a Positive Region-of-Interest Generator that generates bounding boxes with different class, IoU and spatial distributions to analyse imbalance problems in the second stage of Faster R-CNN.
- We thoroughly analyse Average Precision (AP) and Panoptic Quality (PQ) performance measures, and then propose “Localisation-Precision-Recall (LRP) Error” performance metric to evaluate *all* visual detection tasks. We show that LRP Error addresses the limitations of AP and PQ. Moreover, LRP Error is an upper bound for the error versions of precision, recall and PQ. Therefore, minimizing LRP is guaranteed to minimize the other measures.
- While LRP Error can directly be used when the confidence scores are not included in the output (e.g. panoptic segmentation), for evaluating the outputs with confidence scores (e.g. object detection), we propose “Optimal LRP” (oLRP).
- We show that the performances of visual object detectors are sensitive to thresholding, and based on oLRP, we propose “LRP-Optimal Threshold” to reduce the number of detections in an optimal manner.

- We propose “Identity Update” to compute and optimise ranking-based loss functions. Identity Update not only provides a simple and general framework to incorporate such loss functions into backpropagation but also ensures provable balance in terms of the magnitude of the total gradients of positive and negative examples.
- We propose “average Localisation-Recall-Precision (aLRP) Loss”, a unified, bounded, balanced and ranking-based loss function for both classification and localisation tasks in object detection. aLRP Loss has a single hyper-parameter to be tuned, hence it is easy-to-tune, can be used with one-stage and multi-stage methods easily. Replacing our ranking-based baseline, AP Loss, combined with Smooth L1 Loss by aLRP Loss for training RetinaNet improves the performance by $\sim 5\text{AP}$.
- We propose “Rank & Sort (RS) Loss”, as a ranking-based loss function to train deep object detection and instance segmentation methods. RS Loss supervises the classifier, a sub-network of these methods, to rank each positive above all negatives as well as to sort positives among themselves with respect to (wrt.) their continuous localisation qualities. With RS Loss, we significantly simplify training, train seven diverse visual detectors only by tuning the learning rate and consistently improve performance: e.g. RS Loss improves Faster R-CNN by ~ 3 box AP and Mask R-CNN by ~ 2 mask AP.

1.3.1 Publications

These contributions are published in the following papers:

- Kemal Oksuz, Baris Can Cam, Emre Akbas* and Sinan Kalkan*, “Rank & Sort Loss for Object Detection and Instance Segmentation”, under review.
- Kemal Oksuz, Baris Can Cam, Sinan Kalkan* and Emre Akbas*, “One Metric to Measure them All: Localisation Recall Precision (LRP) for Evaluating Visual Detection Tasks”, under review.

* Equal contribution for senior authorship.

- Kemal Oksuz, Baris Can Cam, Emre Akbas* and Sinan Kalkan*, “A Ranking-based, Balanced Loss Function Unifying Classification and Localisation in Object Detection”, Advances on Neural Information and Processing Systems (NeurIPS), 2020.
- Kemal Oksuz, Baris Can Cam, Emre Akbas* and Sinan Kalkan*, “Generating Positive Bounding Boxes for Balanced Training of Object Detectors”, IEEE Winter Conference on Applications of Computer Vision (WACV), 2020.
- Kemal Oksuz, Baris Can Cam, Sinan Kalkan* and Emre Akbas*, “Imbalance Problems in Object Detection: A Review”, Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2020.
- Kemal Oksuz, Baris Can Cam, Emre Akbas* and Sinan Kalkan*, “Localization Recall Precision (LRP): A New Performance Metric for Object Detection”, European Conference on Computer Vision (ECCV), 2018.

1.3.2 Software Contributions

Under the scope of this thesis, the following software have been released as open source:

- 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 available at <https://github.com/kemaloksuz/ObjectDetectionImbalance>.
- We provide the code for our Bounding Box Generator and the positive RoI Generator at <https://github.com/kemaloksuz/BoundingBoxGenerator>.
- We incorporated our LRP Error into the official evaluation codes of the common datasets, COCO [13], LVIS [45], Panoptic COCO [46] at <https://github.com/kemaloksuz/LRP-Error>.
- We release the code for our aLRPLoss at <https://github.com/kemaloksuz/aLRPLoss>, and the code of our RS Loss will be released after its ongoing review process is completed.

1.4 The Outline of the Thesis

This thesis is organized as follows: Chapter 2 provides the frequently used notation and terms throughout the thesis. Chapter 3 provides our review on the imbalance problems in object detection and Chapter 4 introduces Bounding Box Generator as an analysis tool for imbalance. Then, the next two chapters provide necessary tools to define and optimise our ranking-based loss functions: Chapter 5 introduces Localisation-Recall-Performance Error as our novel performance metric and Chapter 6 presents Identity Update to compute and optimise ranking-based loss functions. Subsequently, Chapters 7 and 8 introduce our aLRP Loss and RS Loss respectively, and finally Chapter 9 concludes this thesis.



CHAPTER 2

FREQUENTLY USED NOTATIONS AND DEFINITIONS

This chapter presents the notation used throughout this chapter and the definitions of the frequently used terms.

2.1 Frequently Used Notations

Below is an alphabetically sorted list of frequently used notations throughout the paper.

- AP^C , COCO-Style AP.
- AP_τ , AP when the TPs are validated from the localisation quality, $lq(\cdot, \cdot)$, threshold of τ .
- AR_r^C , Average Recall where r is the number of top-scoring detections to include in the computation of AR.
- $\mathbf{b} \in \mathbb{R}^{|P| \times 4}$, Ground truth BBs. We assume one-to-one correspondence with the predictions, $\hat{\mathbf{b}}$.
- $\hat{\mathbf{b}} \in \mathbb{R}^{|P| \times 4}$, Raw box regression prediction of VD. $\hat{b}_i \in \mathbb{R}^4$ represents the i th box regression head raw prediction originating from i th proposal. With IoU-based performance measures and loss functions, we assume that $\hat{b}_i$ represents a BB.
- $B = [x_1, y_1, x_2, y_2]$, a bounding box.
- C , a set of integers to represent class labels in a dataset.

- d , A detection such that $d \in \mathcal{D}$.
- d_g , A TP detection that matches ground truth g .
- $\mathcal{D}$, A set of detections.
- g , A ground truth such that $g \in \mathcal{G}$.
- $\mathcal{G}$, A set of ground truths.
- $H(x)$, Smoothed unit-step function (Eq. 6.5).
- $|K|$, the number of classes predicted by the classification head. $|K| = |C|$ with binary sigmoid classifiers and $|K| = |C| + 1$ with softmax classifiers due to the additional background class.
- $\ell(i)$, the loss value computed on example i for a ranking-based loss $\mathcal{L}$.
- $\ell(i)^*$, the target loss value on example i .
- $\mathcal{L}$, a loss function.
- L_{ij} , the primary term concerning examples i and j is the loss originating from i and distributed over j via a probability mass function $p(j|i)$.
- L_{ij}^* , the target primary term for the primary term L_{ij} .
- $\text{lq}(\cdot, \cdot)$, A localisation quality function. (e.g. $\text{IoU}(\cdot, \cdot)$).
- LRP, LRP value.
- LRP_{Loc} , localisation error component of LRP.
- LRP_{FN} , FN error component of LRP.
- LRP_{FP} , FP error component of LRP.
- N_{FN} , Number of False Negatives.
- N_{FP} , Number of False Positives.
- N_{TP} , Number of True Positives.

- $N_{\text{FP}}(i)$, Number of False Positives for the set thresholded from i th example. In particular, this term is used in the derivations in ranking-based losses, and indicates the number of negatives with logits larger than $\hat{s}_i$.
- $\mathcal{N}$, the set of negative examples identified based on the assignments of the proposals (i.e. anchors/points/RoIs).
- oLRP, oLRP value.
- oLRP_{Loc} , localisation error component of oLRP.
- oLRP_{FN} , FN error component of oLRP.
- oLRP_{FP} , FP error component of oLRP.
- $|P|$, number of proposals (e.g. anchor/RoI/point).
- $p(j|i)$, the probability mass function to distribute the loss on i over j .
- $\mathcal{P}$, the set of positive examples identified based on the assignments of the proposals (i.e. anchors/points/RoIs).
- P_i , Pyramidal feature layer corresponding to i th backbone feature layer.
- $\hat{\mathbf{p}} \in \mathbb{R}^{|P| \times |K|}$, predicted classification confidence scores (i.e. logits) from a VD where $|P|$ is the number of proposals and $|K|$ is the number of classes predicted by the classifier. Note that p_{ij} is obtained by applying element-wise sigmoid or proposal-wise softmax operation depending on the classifier.
- $\text{rank}^+(i)$, the ranking position of the i th example among positives.
- $\text{rank}(i)$, the ranking position of the i th sample among all samples.
- $\mathbf{s} \in \{-1, 0, 1\}^{|P| \times |K|}$, ground truth class labels. We assume one-to-one correspondence with $\hat{\mathbf{s}}$. Examples with $-1, 0, 1$ represent ignored examples, negative (background) examples and positive (foreground) examples respectively. We represent the elements of $\mathbf{s}$ similar to $\hat{\mathbf{s}}$ (Note that different from this context s represents the confidence score of a detection, see its explanation for further details).

- $\hat{s} \in \mathbb{R}^{|P| \times |K|}$, Raw classification predictions (i.e. logits) from a VD where $|P|$ is the number of proposals and $|K|$ is the number of classes predicted by the classifier. We use two different element representations on $\hat{s}$: (i) $\hat{s}_{ij}$ assumes $\hat{s}$ is 2D (matrix) and represents the logit of i th proposal (i.e. anchor/RoI) corresponding to j th class, and (ii) $\hat{s}_i$ assumes $\hat{s}$ is 1D (vector) and represents the i th among outputs.
- s^* , LRP-Optimal confidence score.
- $\mathcal{S}$, A set of confidence scores.
- x_{ij} , the difference transform (i.e. $\hat{s}_j - \hat{s}_i$) between the logit of i th prediction ($\hat{s}_i$) and the logit of j th prediction ($\hat{s}_j$).
- Δx_{ij} , the update in difference transforms x_{ij} .
- τ , TP validation threshold in terms of localisation quality.

While this list of notation is consistently adopted throughout the text, some of the chapter-specific notation (the notation which is used only by a single chapter), which are used locally, is not included in this list for clarity. Furthermore, these local symbols are overloaded in rare cases and kept intentionally as they are in order to preserve some equations as they are. To illustrate, Z is used to denote both the normalization constants of LRP Error (Eq. 5.2) and the generic ranking-based loss function (Eq. 6.17). In that case Z is defined in both of the chapters and used as it is defined throughout that chapter.

2.2 Frequently Used Terms

Below is a list of frequently used terms in this thesis:

Bounding Box: A rectangle. Formally, a bounding box (BB), denoted by B , is generally represented by $[x_1, y_1, x_2, y_2]$ with (x_1, y_1) denoting the top-left corner and (x_2, y_2) the bottom-right corner, with the constraints $x_2 > x_1$ and $y_2 > y_1$. Accordingly, the area of a BB is:

$$A(B) = (x_2 - x_1) \times (y_2 - y_1). \quad (2.1)$$

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

Region of Interest (RoI): The set of bounding boxes generated by a proposal mechanism such as RPN on which the detection network is applied on multi-stage detectors.

Proposal: Any type of object hypotheses, e.g. anchor, RoI, a point/pixel.

Feature Extraction Network: This is the part of the detection pipeline from the input image until the detection network (e.g. Fig. 1.2(a)).

Backbone: This is the part of the feature extraction network until pyramidal network (e.g. FPN [29]), if exists; else it corresponds to the feature extraction network. An example is ResNet [47].

Classification Head/Classification Network/Classifier: This is the part of the detection pipeline from the extracted features to the classification prediction.

Box Regression Head/Box Regression Network/Regression Network/Regressor: This is the part of the detection pipeline from the extracted features to the regression output.

Detection Network/Detector: It is the part of the detection pipeline including possible shared layers among heads followed by visual-detection task specific heads/sub-networks (e.g. classification and box regression heads for object detection – e.g. Fig. 1.2(b)).

Region Proposal Network (RPN): As a single class object detector (i.e. only includes classification and box regression heads) to determine the objectness score of the anchors, RPN is the first detection network of multi-stage networks. More particularly, it is the part of the multi-stage detection pipeline from the multi-scale feature maps until the RoIs.

Intersection Over Union (IoU): For two polygons P_g and P_d , Intersection over Union (IoU) [48, 49], $\text{IoU}(P_g, P_d)$ is defined as :

$$\text{IoU}(P_g, P_d) = \frac{A(P_g \cap P_d)}{A(P_g \cup P_d)} = \frac{A(P_g \cap P_d)}{A(P_g) + A(P_d) - A(P_g \cap P_d)}, \quad (2.2)$$

where $A(P)$ is the area of the polygon P (i.e. number of pixels delimited by P).

These polygons are represented by bounding boxes for object detection, and by masks for segmentation tasks (e.g. instance segmentation, panoptic segmentation). $\text{IoU} \in [0, 1]$ and it is used to evaluate the localisation quality of a detection bounding box/-mask with respect to its ground truth box/mask.

Object Keypoint Similarity (OKS): Given target and estimated keypoint sets, K_g and K_d respectively, with $k_g^i \in K_g$ and $k_d^i \in K_d$ being the i th keypoint of the object represented by a 2D coordinate on the image, Object Keypoint Similarity (OKS) [13] between k_g^i and k_d^i , denoted by $\text{OKS}(k_g^i, k_d^i)$, is

$$\text{OKS}(k_g^i, k_d^i) = \frac{\exp(-\|k_g^i - k_d^i\|^2)}{2(S\kappa^i)^2}, \quad (2.3)$$

where $\|\cdot - \cdot\|$ is Euclidean distance, κ^i is the constant corresponding to i th keypoint to control falloff, and S is the object scale (e.g. the area of the ground truth bounding box divided by the total image area). Then, OKS between K_g and K_d , $\text{OKS}(K_g, K_d)$, is simply the average of $\text{OKS}(k_g^i, k_d^i)$ over single keypoints annotated in the dataset:

$$\text{OKS}(K_g, K_d) = \frac{1}{|K_g|} \sum_{i \in |K_g|} \text{OKS}(k_g^i, k_d^i). \quad (2.4)$$

Similar to $\text{IoU}(\cdot, \cdot)$; $\text{OKS}(\cdot, \cdot) \in [0, 1]$ and a larger $\text{OKS}(\cdot, \cdot)$ implies better localisation quality.

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 (see “Feature maps obtained in different scales” in Fig. 1.2(a)).

Hard Prediction: A type of visual detector output which identifies each object with (i) a set of identifiers to locate an object (e.g. bounding box, mask, keypoints), and (ii) its class label.

Soft Prediction: A type of visual detector output which identifies each object with (i) a set of identifiers to locate an object (e.g. bounding box, mask, keypoints), (ii) its class label and (iii) the confidence score of the prediction.

Keypoint Set: A set of coordinates to represent an object on the image such that each element is a two tuple (x_i, y_i) identifying a keypoint of an object.

Segmentation Mask: A set of pixels presenting which pixels belong to a particular object.





CHAPTER 3

IMBALANCE PROBLEMS IN OBJECT DETECTION

In this chapter, we present a review of the imbalance problems in object detection based on our work,

- Kemal Oksuz, Baris Can Cam, Sinan Kalkan* and Emre Akbas*, “Imbalance Problems in Object Detection: A Review”, Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2020.

We only make some minor changes to fit the text appropriately in the context of this thesis (e.g. notation) and some minor corrections.

The Scope of the Review. Since imbalance problems in general have a large scope in machine learning, computer vision and pattern recognition, we limit our focus to imbalance problems in object detection and since the current state-of-the-art is shaped by deep learning based approaches, the problems and approaches that we discuss here 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 believe that these discussions would provide insights on future research directions for object detection researchers.

Our main 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.

* Equal contribution for senior authorship.

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 chapter (in Appendix A) 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 has been continuously updated with new studies.

Comparison with Previous Reviews. Recent object detection surveys [50, 51, 52] 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. [51] 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) [53, 54, 55, 56]. Although Zehang Sun et al. [53] and Dollar et al. [54] 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. [55] and Yin et al. [57] propose comparative analyses of non-deep and deep

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

methods. Litjens et al. [58] 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 [59, 60, 61, 62]. 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. Secondly, except Johnson et al. [62], 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 1.2).

A Guide to Reading This Chapter/Review. This chapter is organized as follows. Section 3.1 presents our taxonomy of imbalance problems. Sections 3.2-3.5 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 3.6 discusses open issues that are relevant to all imbalance problems. Finally, Section 3.7 concludes this chapter. In order to provide a more general perspective, in Appendix A, we present the solutions addressing imbalance in other but closely related domains.

For readers, familiar with SOTA methods in object detection, this chapter is self-contained and thus is readable without other chapters of this thesis with the exception

that you can refer to Chapter 2 and the list of abbreviations provided in page xxxix for notations, definitions and abbreviations. For readers who lack a background in state-of-the-art object detection, we recommend starting with Chapter 1, in which we provide a basic background and motivation for visual detection and imbalance problems, and if this brief background is not sufficient, we refer the reader to the more comprehensive object detection surveys [50, 51, 52].

3.1 A Taxonomy of the Imbalance Problems and their Solutions in Object Detection

In Chapter 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. 3.2). 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 1.2 presents the complete taxonomy along with a brief definition for each problem. In Fig. 3.1, we present the same taxonomy along with a list of proposed solutions for each problem. Finally, in Fig. 3.2, we illustrate a generic object detection 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 3.2; blue branch in Fig. 3.1) 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. 3.5). The class imbalance is usually handled at the “sampling” stage in the object detection pipeline (Fig. 3.2).

Scale imbalance (Section 3.3; green branch in Fig. 3.1) is observed when object

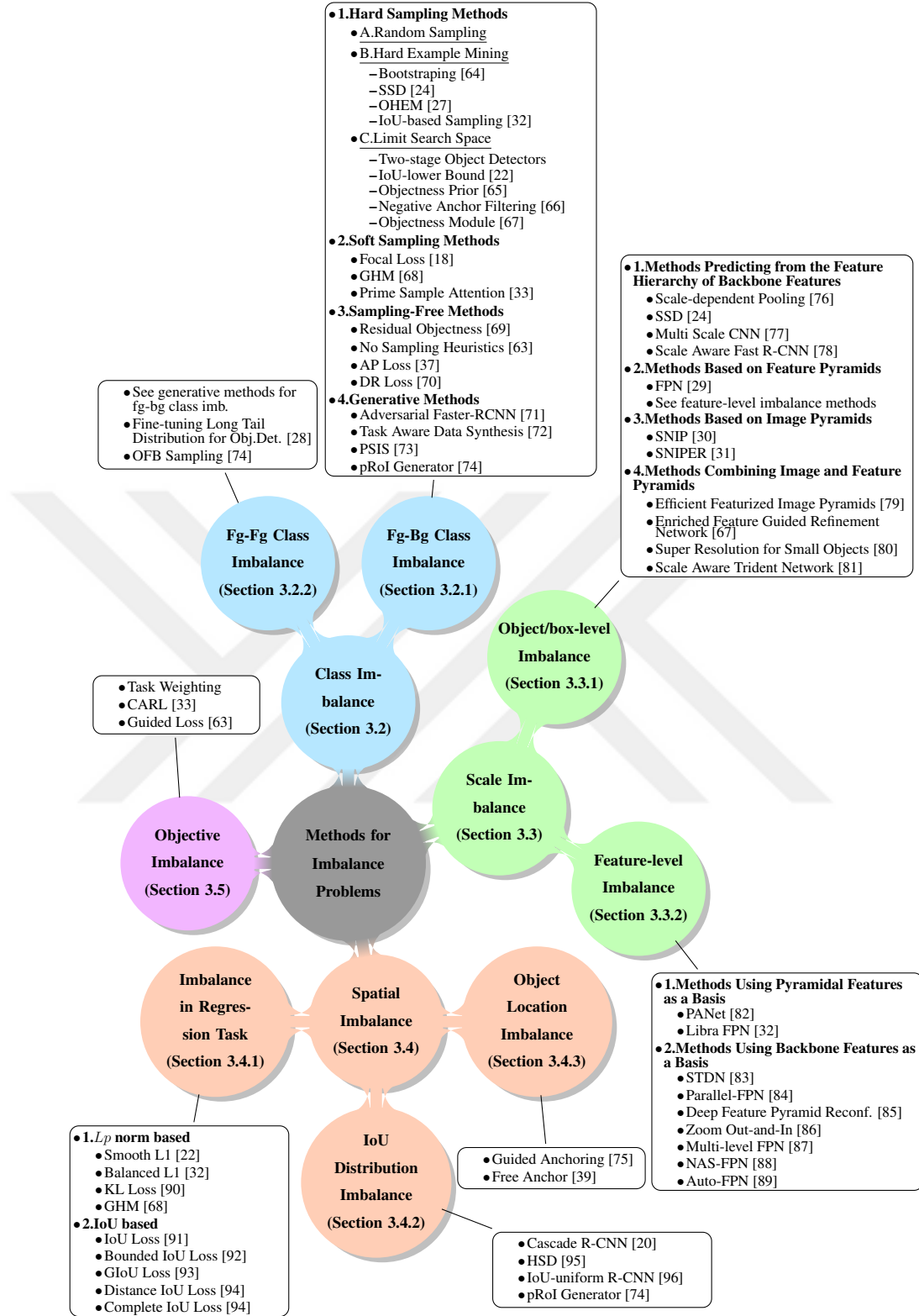


Figure 3.1: Problem based categorization of the methods used for imbalance problems. A work may appear at multiple locations if it addresses multiple problems.

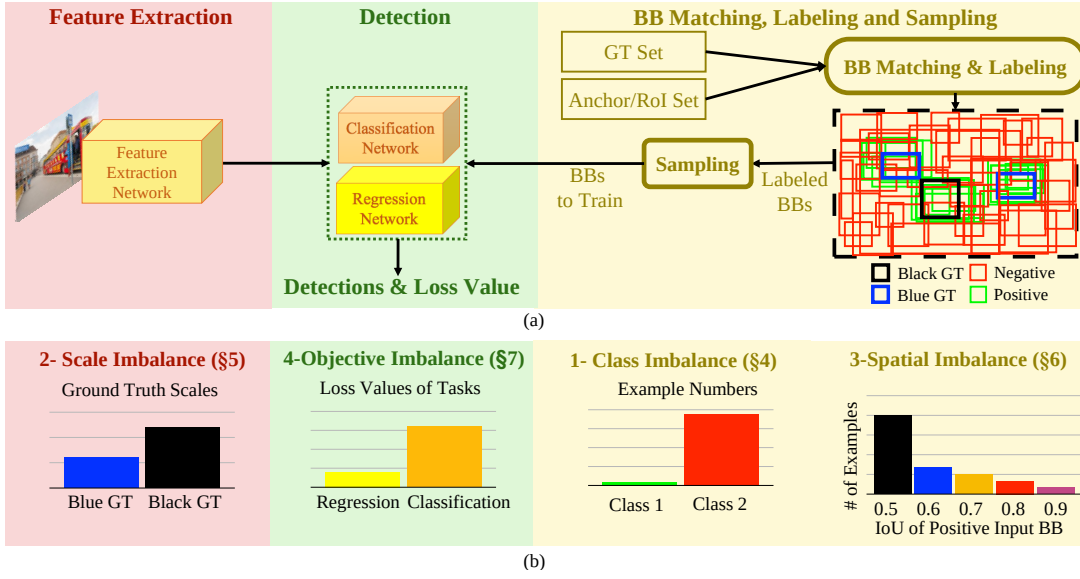


Figure 3.2: **(a)** The common training pipeline of a generic detection network. The pipeline has 3 phases (i.e. feature extraction, detection and BB matching, labeling and sampling) represented by different background colors. **(b)** Illustration of an example imbalance problem from each category for object detection through the training pipeline. Background colors specify at which phase an imbalance problem occurs.

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. 3.2), 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 3.3, proposed methods fall short in addressing scale imbalance, especially for small objects, even when small objects are surprisingly abundant in a dataset.

Spatial imbalance (Section 3.4; orange branch in Fig. 3.1) 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. 3.2) (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 3.5; purple branch in Fig. 3.1) 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. 3.1 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. 3.3) 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 review.

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. [26, 97]).

3.2 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

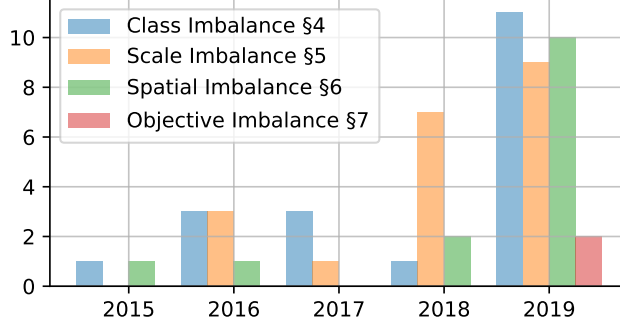


Figure 3.3: Number of papers per imbalance problem category through years.

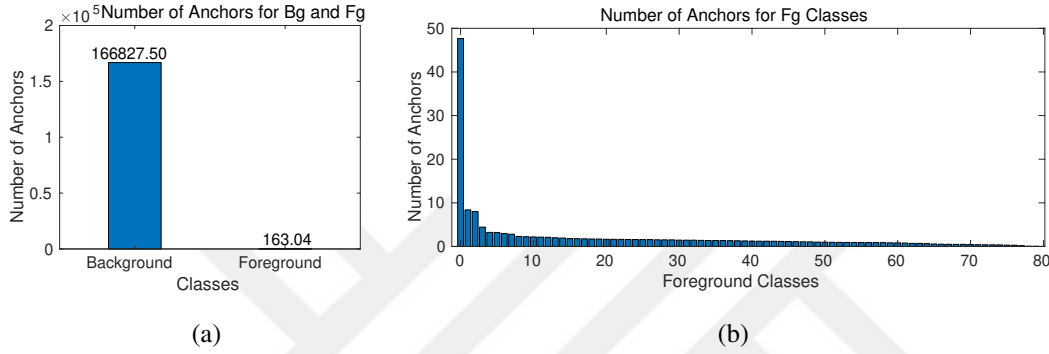


Figure 3.4: Illustration of the class imbalance problems. The numbers of RetinaNet [18] anchors on COCO [13] are plotted for foreground-background (a), and foreground classes (b). The values are normalized with the total number of images in the dataset. The figures depict severe imbalance towards some classes.

imbalance.

Fig. 3.4 illustrates the presence of class imbalance. To generate the figure, we apply the default set of anchors from RetinaNet [18] on the COCO dataset² [13] and calculated the frequencies for the cases where the IoU of an anchor with a ground truth bounding box exceeds 0.50 and where it is less than 0.40 (i.e. it is a background box) following the labeling rule of RetinaNet [18]. When an anchor overlaps with a foreground class (with IoU > 0.50), we kept a count for each class separately and normalized the resulting frequencies with the number of images in the dataset.

² Throughout the text, unless otherwise specified, “COCO”, “PASCAL”, and “Open Images” respectively correspond to the Pascal VOC 2012 trainval [48], COCO 2017 train [13], Open Images v5 training (subset with bounding boxes) [98] and Objects365 train [99] dataset partitions. And, when unspecified, the backbone is ResNet-50 [47].

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.

3.2.1 Foreground-Background (a.k.a. Positive-Negative) 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. 3.4(a). 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_{ij}) of the prediction of the network on the proposal i for class j to the loss function is adjusted as follows:

$$w_{ij} \text{CE}(\hat{s}_{ij}, s_{ij}), \quad (3.1)$$

where $\hat{s}_{ij}$ is the classification network prediction of i th proposal (e.g. anchor/RoI) corresponding to j th class and $\text{CE}()$ is the cross-entropy loss. Hard and soft sampling approaches differ on the possible values of w_i . For the hard sampling approaches, $w_{ij} \in \{0, 1\}$, thus a prediction is either selected or discarded. For soft sampling approaches, $w_{ij} \in [0, 1]$, i.e. the contribution of a sample is adjusted with a weight and each prediction is somehow included in training.

3.2.1.1 Hard Sampling Methods

Hard sampling is a commonly-used method for addressing imbalance in object detection. It restricts w_{ij} 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. $w_{ij} = 1$ for all classes j of the selected example) and the non-selected predictions ($w_{ij} = 0$ for all classes j of the non-selected example) have no contribution to the training for the current iteration. See Table 3.1 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 [19, 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 [22]³. In any case, if the number of positive proposals (i.e. anchor or RoI) 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 [27, 32, 33].

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 [64, 101, 102], human detection [103] and object detection [15]. 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

³ Recent detection frameworks such as mmdetection [100] generally adopt larger numbers: For RPN, 256 positives and negatives are randomly selected, and for R-CNN 128 positives and 384 negatives are selected.

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

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 [24], 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) [27]. However, OHEM needs additional memory and causes the training speed to decrease. Considering the efficiency and memory problems of OHEM, *IoU-based sampling* [32] 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 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 easy to mine. *Two-stage object detectors* [19, 23] are among these methods since they aim to find the most probable bounding boxes (i.e. RoIs) given anchors, and then choose top RoIs with the highest objectness scores, to which an additional sampling method is applied. Fast R-CNN [22] 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. [65] 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. [66] 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. [67] 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.

3.2.1.2 Soft Sampling Methods

Soft sampling adjusts the contribution (w_{ij}) of each prediction 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 3.1 for a summary of the main approaches.

A straightforward approach is to use constant coefficients for both the foreground and background classes. YOLO [25], having less number of anchors compared to other one-stage methods such as SSD [24] and RetinaNet [18], is a straightforward example for soft sampling in which the loss values of the predictions labeled as background are halved (i.e. $w_{ij} = 0.50$).

Focal Loss [18] is the pioneer example that dynamically assigns more weight to the hard examples as follows:

$$w_{ij} = \begin{cases} (1 - \hat{p}_{ij})^\gamma, & \text{if } s_{ij} = 1 \\ \hat{p}_{ij}^\gamma, & \text{if } s_{ij} = 0. \end{cases} \quad (3.2)$$

where $\hat{p}_{ij}$ is the predicted probability for the ground-truth class (e.g. for RetinaNet $\hat{p}_{ij} = \text{sigmoid}(\hat{s}_{ij})$ [18]). Note that as the predicted ground truth probability approximates to the ground truth value, Eq. 3.2 assigns a smaller weight to the prediction, thereby suppressing the easier examples. E.g. when a background example (i.e. $s_{ij} = 0$) is correctly classified (i.e. $p_{ij} \approx 0$), its contribution to the loss will be decreased due to small p_{ij} . Note that when $\gamma = 0$, focal loss degenerates to vanilla cross entropy loss and Lin et al. [18] showed that $\gamma = 2$ ensures a good trade-off between hard and easy examples for their model and dataset.

Similar to focal loss [18], *Gradient Harmonizing Mechanism* (GHM) [68] 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

Table 3.1: Toy example for hard and soft sampling methods. One positive and two negatives are chosen from six proposals. For soft sampling, numbers are the weights of each prediction. We assume one foreground class and use w_i (instead of w_{ij}) for i th proposal.

	Legend&Method	Considered Property	Intuition	Addit. Params.	Positive Examples		Negative Examples			
Boxes	✓ Selected BB ✗ Discarded BB	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×1		Random Sample×2			
	SSD [24]	Loss	Chooses neg. with max loss	–	Random Sample×1		✓	✗	✗	✗
	OHEM [27]	Loss	Chooses pos.&neg. with max loss	–	✗	✓	✓	✗	✗	✗
	IoU-Based Sampling [32]	IoU	Samples equal number of negatives from IoU bins	K=2	Does not specify a criterion for positives		Random Sample×1	Random Sample×1		
	IoU Lower Bound [22]	IoU=0.05	Discards neg. having IoU less than lower bound	–	Does not specify a criterion for positives		Random Sample × 2			
	Objectness Prior [65]	–	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 [66]	$p_s = 0.9$	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 [18]	p_s	Promotes examples with larger loss	$\gamma = 1$	0.2	0.8	0.9	0.7	0.4	0.0
	Gradient Harmonizing Mechanism [68]	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 [33]	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

a sample if there are many samples with similar gradients as follows:

$$w_i = \frac{1}{G(i)/|P|}, \quad (3.3)$$

where $G(i)$ is the count of samples whose gradient norm is close to the gradient norm of i th prediction; and $|P|$ is the number of proposals 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 3.4.1.

Different from the latter soft sampling methods, *PrIme Sample Attention* (PISA) [33] 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:

$$\bar{r}_i = \frac{N_{max} - r_i}{N_{max}}, \quad (3.4)$$

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:

$$w_i = ((1 - \beta)\bar{r}_i + \beta)^\gamma, \quad (3.5)$$

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 Eq. 3.4 and 3.5 increases the contribution of the samples with high IoUs for positives and high scores for negatives to the loss.

3.2.1.3 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. [69] 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 [63] 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 (Section 3.5) 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* [37] which formulates the classification part of the loss as a ranking task (see also *DR Loss* [70] 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. Since AP Loss is the work that we are inspired for our aLRP Loss and RS Loss, we discuss it in detail in Section 6.1.

3.2.1.4 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 3.2 presents a summary of the main approaches.

One approach is to use generative adversarial networks (GANs). A merit of GANs

Table 3.2: Comparison of major generative methods addressing class imbalance.

Generative Method	Generates
Adversarial-Fast-RCNN [71]	Occluded and spatially transformed features during RoI pooling in order to make the examples harder
Task Aware Data Synthesis [72]	Images with hard examples in a GAN setting in which given foreground mask is to be placed onto the given image by the generator.
PSIS [73]	Images by switching the instances between existing images considering the performance of the class during training
pRoI Generator [74]	Positive RoIs (i.e. BBs) following desired IoU, spatial and foreground class distributions

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 [71], 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 [22]). 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 [72, 73, 104, 105] 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. [104]. 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 [105]. In a similar vein, the objects can be swapped between images: *Progressive and Selective Instance-Switching* (PSIS) [73] 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* [72] 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* [74] 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 proposal is related to its hardness [32], the pRoI generator is a basis for simulating, thereby analyzing, hard sampling methods (Section 3.2.1.1).

3.2.2 Foreground-Foreground (a.k.a. Positive-Positive) 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.

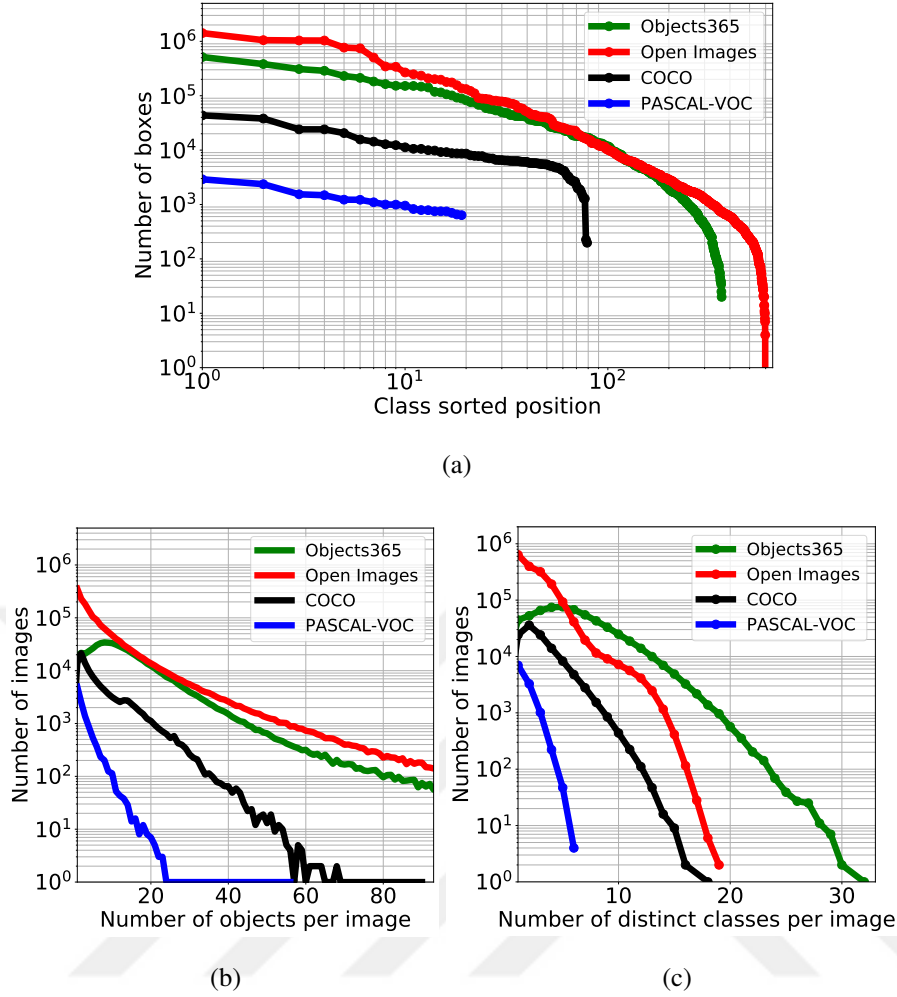


Figure 3.5: Some statistics of common datasets (training sets). For readability, the y axes are in logarithmic scale. **(a)** The total number of examples from each class. **(b)** The number of images vs. the number of examples. **(c)** The number of images vs. the number of classes. (This figure is inspired from Kuznetsova et al. [98] but calculated and drawn from scratch).

3.2.2.1 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 – Fig. 3.5(a) which shows that the datasets suffer from significant gap in class examples. For this reason, over-fitting 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 (Section 3.2.1.4) 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. [28]. 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 [106]), 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 proposal.

3.2.2.2 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. 3.4(b) provides the mean number of anchors per class on the COCO dataset [13]. 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 Fig. 3.5(b), and (c), 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. [74], 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 3.2.4.

3.2.3 Comparative Summary

In early deep object detectors, hard sampling methods were commonly used to ensure balanced training. OHEM [27] was one of the most effective methods with a relative improvement of 14.7% over the Fast R-CNN baseline (with VGG-16 backbone) on 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 [18] achieves 9.3% relative improvement compared to the baseline alpha-balanced cross entropy on Retina-Net (i.e. from 31.1 to 34.0 AP^{C5}). AP Loss [37], with a relative improvement of 3.9% over Focal Loss (i.e. from 33.9 to 35.0 AP^C), is also promising.

Sampling methods for positive examples are not many. One approach, prime sample attention (PISA) [33], reported a relative improvement of 4.4% (from 36.4 to 38.0 AP^C 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 [27] applied to positives. Moreover, the results suggest that the major improvement of the method is on localisation since there is no performance improvement in AP₅₀, and there is significant improvement for APs with higher IoUs (i.e. up to 2.6% in AP₇₅). 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 3.4.2).

⁵ Unless otherwise stated, COCO 2017 minival results are reported using the COCO-style AP, denoted by AP^C.

3.2.4 Open Issues

As we have highlighted before, class imbalance problem can be analysed 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.

3.2.4.1 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 [27] for the positive examples is challenged by the idea that favors examples with higher IoUs, i.e. the easier ones [33]. (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. [68] 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 [18, 68] 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.

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

For the foreground examples, Cao et al. [33] 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 [27] 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.

3.2.4.2 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 [73] 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 [73] 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 (Appendix A). The similarity is that neither has a background class. On the other hand, the proposals 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-background imbalance in object detection as well.

3.2.4.3 Foreground-Background 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. 3.6 from the COCO dataset [13]: 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-background 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-background class imbalance entirely.

3.2.4.4 Ranking-Based Loss Functions

Open Issue. AP Loss [37] 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 [107].



Figure 3.6: 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 COCO dataset.

Explanation. Chen et al. [37] use all the confidence scores all together without paying attention to the fact that the optimal threshold per class may vary [107]. 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.

3.3 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 proposals (i.e. anchor or RoI). The second part, Feature Imbalance, analyses the problems arising at the feature extraction level and concerns the methods using pyramidal features.

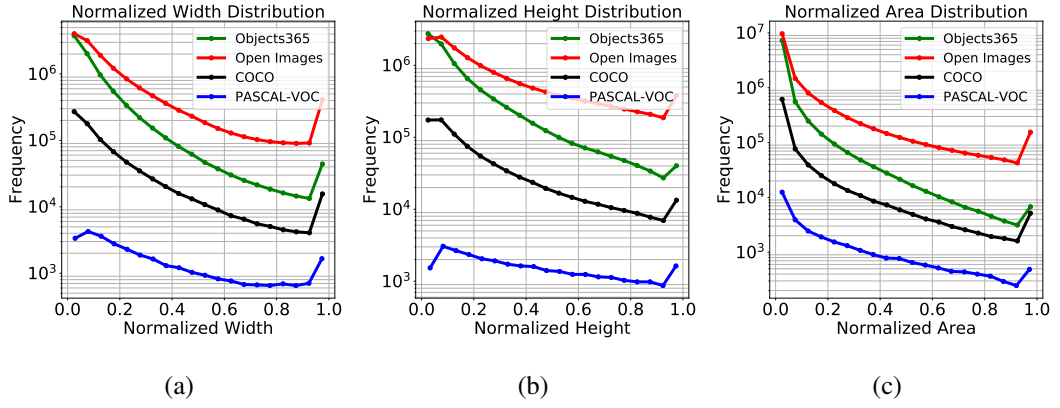


Figure 3.7: Imbalance in scales of the objects in common datasets: the distributions of BB width (a), height (b), and area (c). Values are with respect to the normalized image (i.e. relative to the image). For readability, the y axes are in log-scale.

3.3.1 Object/Box-Level Scale Imbalance

Definition. Scale imbalance occurs when certain sizes of the objects or proposals (i.e. anchor or RoI) are over-represented in the dataset. It has been shown that this affects the scales of the estimated RoIs and the overall detection performance [30]. Fig. 3.7 presents the relative width, height and the area of the objects in the COCO dataset [13]; 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. [47, 106, 108, 109, 110]), pretrained on an image classification task, in order to extract visual features from the input image. Li et al. [111] 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 down-sampling 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 proposal boxes (i.e. anchors/RoIs).

Solutions. First examples of deep object detectors made predictions from the final layer of the backbone network (see [21, 22] and Fig. 3.8(a)), and therefore, neglected the scale-diversity of BBs. The solutions to addressing scale imbalance can be

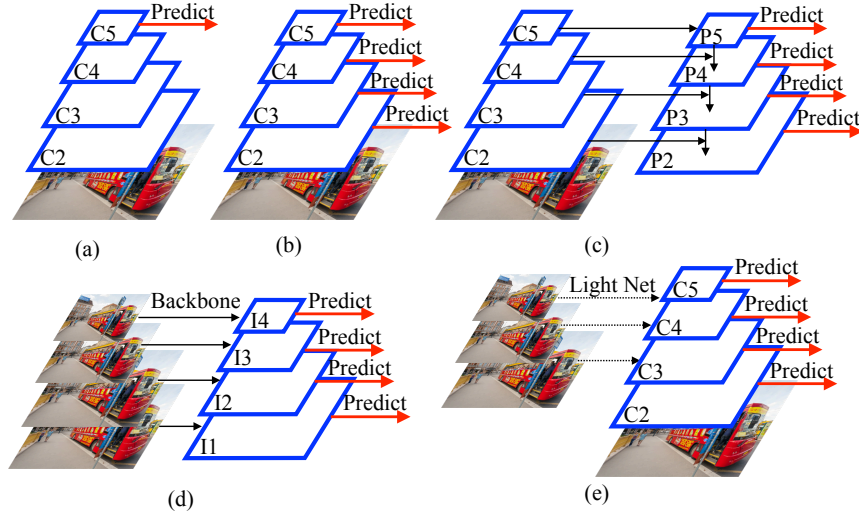


Figure 3.8: An illustration and comparison of the solutions for scale imbalance. “Predict” refers to the prediction performed by a detection network. The layered boxes correspond to convolutional layers. **(a)** No scale balancing method is employed. **(b)** Prediction is performed from backbone features at different levels (i.e. scales) (e.g. SSD [24]). **(c)** The intermediate features from different scales are combined before making prediction at multiple scales (e.g. Feature Pyramid Networks [29]). **(d)** The input image is scaled first and then processed. Each I corresponds to an image pyramidal feature (e.g. Image Pyramids [30]). **(e)** Image and feature pyramids are combined. Rather than applying backbones, light networks are used in order to extract features from smaller images.

grouped into four (Fig. 3.8): methods predicting from the hierarchy of the backbone features (Fig. 3.8(b)), methods based on feature pyramids (Fig. 3.8(c)), methods based on image pyramids (Fig. 3.8(d)) and finally methods combining image and feature pyramids (Fig. 3.8(e)).

3.3.1.1 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. 3.8(b)). 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 [76].

An illustrative example for one-stage detectors is the Single Shot Detector (*SSD*) [24], 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) [77] uses different layers of the backbone network while estimating the regions in the first stage whereas Yang et al. [76] 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* [78] learns an ensemble of two classifiers, one for the small scale and one for the large scale objects, and combines their predictions.

3.3.1.2 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) [29] 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 (Fig. 3.8(c)). 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 popu-

lar and has been used for different (but related) tasks such as shadow detection [112], instance segmentation [113, 114] and panoptic segmentation [115].

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 3.3.2.

3.3.1.3 Methods Based on Image Pyramids

The idea of using multi-scale image pyramids, presented in Fig. 3.8(d), for the image processing tasks goes back to the early work by Adelson et al. [116] 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 [30] 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 proposal (i.e. anchor or RoI) 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* [31].

3.3.1.4 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. 3.8(e). For example, *Efficient Featurized Image Pyramids* [79] uses five images at different scales, four of which are provided to the light-weight featurized 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* [67] 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 [81]. 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. [79] only provide experiments with the SSD backbone, Nie et al. [67] 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. [80] 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* [81], 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 [117] 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, a proposal box (i.e. anchor or RoI) 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 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.

3.3.2 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 analyse the conventional FPN architecture in Fig. 3.9, 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

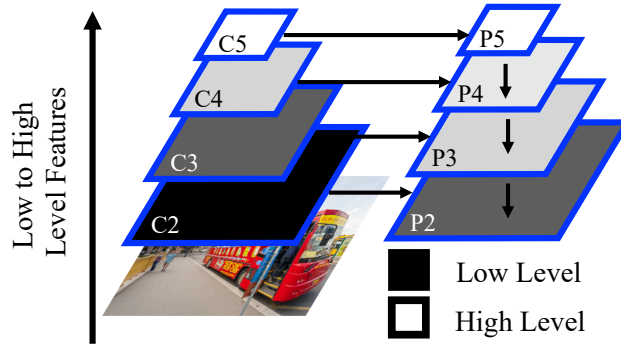


Figure 3.9: Feature-Level imbalance is illustrated on the FPN architecture.

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 [65, 66] 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.

3.3.2.1 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 Fig. 3.10(a,b).

Path Aggregation Network (PANet) [82] 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 localising objects, however, the FPN architecture does not sufficiently make use of these features. Motivated from this observation, PANet, depicted in Fig. 3.10(a), 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 (red arrows in Fig. 3.10(a) within FPN and to the final pyramidal features

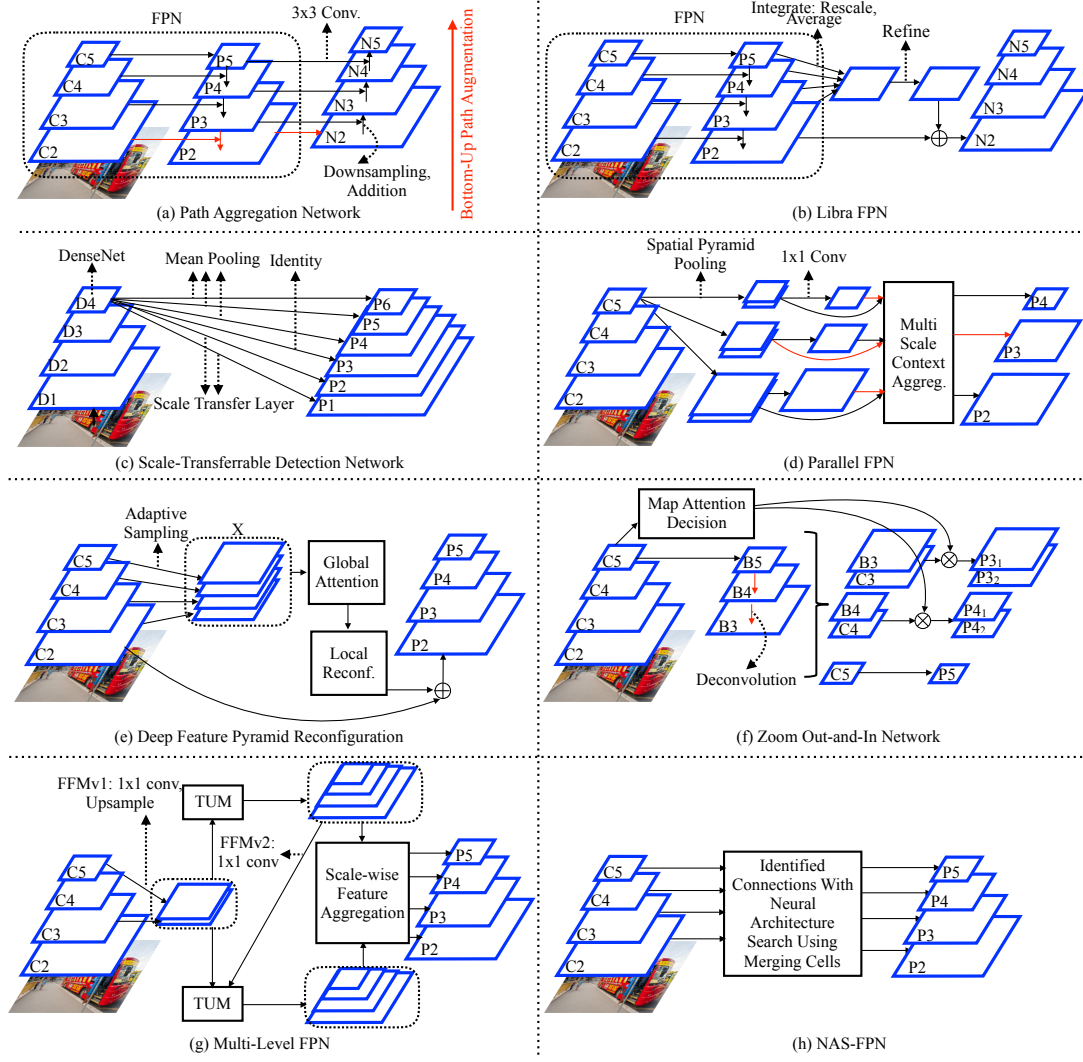


Figure 3.10: High-level diagrams of the methods designed for feature-level imbalance. **(a)** *Path Aggregation Network*. FPN is augmented by an additional bottom-up pathway to facilitate a shortcut (Red arrows) of the low-level features to the final pyramidal features. **(b)** *Libra FPN*. A residual feature map is learned. We illustrate on P_2 . **(c)** *Scale Transferrable Detection Network*. Pyramidal features are learned via pooling, identity mapping and scale transfer layers. **(d)** *Parallel FPN*. Red Arrows: input and outputs of MSCA for P_3 . **(e)** *Deep Feature Pyramid Reconfiguration*. Residual features are learned via global attention and local reconfiguration modules. Illustrated on P_2 . **(f)** *Zoom Out-And-In Network*. **(g)** *Multi-Level FPN*. TUM: Thinned U-Shape Module **(h)** NAS-FPN.

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 localisation 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 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* [32] aims to learn the residual features by using all of the features from all FPN layers at once (Fig. 3.10(b)). 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 [118].

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 [82].

3.3.2.2 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 Fig. 3.10(c-h).

Scale-Transferrable Detection Network (STDN) [83] generates the pyramidal features from the last layer of the backbone features which are extracted using DenseNet

[119] blocks (Fig. 3.10(c)). In a DenseNet block, all the lower level features are propagated to every next layer within a block. In Fig. 3.10(c), 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* [84] also employs only the last layer of the backbone network and generates multi-scale features by exploiting the spatial pyramid pooling (SPP) [120] – Fig. 3.10(d). 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. 3.10(e) 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. 3.10(e). 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* [85] combines features from different levels of backbone features into a single tensor (X in Fig. 3.10(e)) 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,

1. *Global Attention Module* aims to learn the inter-dependencies among different feature maps for tensor X . The authors adopt Squeeze and Excitation Blocks [121] in which the information is squeezed to lower 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* [86] 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. 3.10(f), are learned. Note that, unlike FPN [29], 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 [121] employed by Kong et al. [85], 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) [122] 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. [85] and Li et al. [86], *Multi-Level FPN* [87] 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 (Fig. 3.10(g)). *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 [85], the authors employed squeeze and excitation networks [121] to combine different pyramidal shape features.

Rather than using hand-crafted architectures, *Neural Architecture Search FPN (NAS-FPN)* [88] aims to search for the best architecture to generate pyramidal features given the backbone features by using neural architecture search methods [123] – Fig. 3.10(h). This idea was also previously applied to the image classification task and showed to perform well [124, 125, 126]. *Auto-FPN* is also another example for using NAS while learning the connections from backbone features to pyramidal features and beyond. While 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. [127], however it is not within the scope of our review. Considering their performance in other tasks such as EfficientNet [126] and different definitions of search spaces may lead to better performance in NAS methods, more work is expected for FPN design using NAS.

3.3.3 Comparative Summary

SSD [24] 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 AP₅₀ on Pascal VOC 2007 test set [48] 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 [29] has a relative improvement of 3.7% on 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) [29]. 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% relative improvement is obtained on COCO testdev (from 40.1 to 44.2) by NAS-FPN [88] 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% more (92.1ms vs 73.0ms 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 3.3.1.4 are published last year. To illustrate the performance gains: Pang et al. [79] achieved 19.5% on 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 (AP^C s are 25.1 vs 30.0).

3.3.4 Open Issues

Here we discuss open problems concerning scale imbalance.

3.3.4.1 Characteristics of Different Layers of Feature Hierarchies

In feature-pyramid based methods (Section 3.3.2), 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 [32] addresses all three aspects, Path Aggregation Network [82] handles abstractness and cardinality, whereas FPN solves only abstractness to a certain extent), these aspects should be quantified and used for comparing different methods.

3.3.4.2 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 [30]) is still an open problem.

Explanation. The image pyramids (Fig. 3.8(d)) 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.

3.4 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 (localisation) 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.

3.4.1 Imbalance in Regression Loss

Definition. This imbalance problem is concerned with the uneven contributions of different individual examples to the regression loss. Fig. 3.11 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

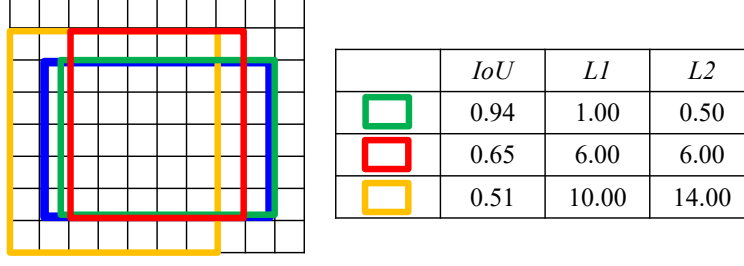


Figure 3.11: An illustration of imbalance in regression loss. Blue denotes the ground truth BB. There are three prediction boxes, marked with green, red and yellow colors. In the table on the right, $L1$ and $L2$ columns show the sum of $L1$ and $L2$ errors between the box corners of the associated prediction 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.

second one is the IoU-based loss functions. Table 3.3 presents a comparison of the widely used regression losses.

Replacing $L2$ regression loss [22, 128], *Smooth L1 Loss* [22] is the first loss function designed specifically for deep object detectors, and it has been widely adopted (e.g. [18, 19, 24]) since it reduces the effect of the outliers (compared to $L2$ loss) and it is more stable for small errors (compared to $L1$ loss⁸). Smooth $L1$ loss, a special case of Huber Loss [129], is defined as:

$$L1_{smooth}(b) = \begin{cases} 0.5(b)^2, & \text{if } |b| \leq \beta \\ \beta(|b| - 0.5\beta), & \text{otherwise.} \end{cases} \quad (3.6)$$

where $b = b_{ij} - \hat{b}_{ij}$ such that $\hat{b}_{ij}$ is the j th parameter of $\hat{b}_i \in \mathbb{R}^4$, the predicted parameters from the box regression network; and b_i , the ground truth corresponding to $\hat{b}_i$, is similarly represented. β is the cut-off parameter to indicate the coordinate to switch from $L2$ Loss to $L1$ Loss. Setting β also depends on the normalization of the ground truth parameters. For example, when they are normalized to unit Gaussian, then β is set to 0.11 in general [100].

Motivated by the fact that the gradients of the outliers still have a negative effect on

⁸ However, recently using $L1$ loss has become more common compared to Smooth $L1$ loss [14, 100]

Table 3.3: A list of widely used loss functions for the BB regression task.

Loss Function	Explanation
$L2$ Loss	Employed in earlier deep object detectors [21]. Stable for small errors but penalizes outliers heavily.
$L1$ Loss	Not stable for small errors. ⁷
Smooth $L1$ Loss [22]	Baseline regression loss function. More robust to outliers compared to $L1$ Loss.
Balanced $L1$ Loss [32]	Increases the contribution of the inliers compared to smooth $L1$ loss.
Kullback-Leibler Loss [90]	Predicts a confidence about the proposal box (i.e. anchor or RoI) based on KL divergence.
IoU Loss [91]	Uses an indirect calculation of IoU as the loss function.
Bounded IoU Loss [92]	Fixes all parameters of an input box in the IoU definition except the one whose gradient is estimated during backpropagation.
GIoU Loss [93]	Extends the definition of IoU based on the smallest enclosing rectangle of the inputs to the IoU, then uses directly IoU and the extended IoU, called GIoU, as the loss function.
DIoU Loss, CIoU Loss [94]	Extends the definition of IoU by adding additional penalty terms concerning aspect ratio difference and center distances between two boxes.

learning the inliers with smaller gradients in Smooth $L1$ loss, *Balanced $L1$ Loss* [32] 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:

$$\frac{\partial L1_{balanced}}{\partial b} = \begin{cases} \alpha \ln(\kappa|b| + 1), & \text{if } |b| < 1 \\ \theta, & \text{otherwise,} \end{cases} \quad (3.7)$$

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 Eq. 3.7, $L1_{balanced}$ is derived as follows:

$$L1_{balanced}(b) = \begin{cases} \frac{\alpha}{\kappa}(\kappa|b| + 1) \ln(\kappa|b| + 1) - \alpha|b|, & \text{if } |b| < 1 \\ \gamma|b| + Z, & \text{otherwise,} \end{cases} \quad (3.8)$$

where κ ensures $L1_{balanced}(b = 1)$ is a continuous function, Z is a constant and the

association between the hyper-paramaters is:

$$\alpha \ln(\kappa + 1) = \gamma. \quad (3.9)$$

Having put more emphasis on inliers, Balanced $L1$ Loss improves the performance especially for larger IoUs (namely, AP_{75} improves by %1.1).

Another approach, *Kullback-Leibler Loss* (KL Loss) [90], 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 localisation confidence associated prediction branch [96, 130, 131, 132, 133], 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 localisation compared to the baseline. The method assumes that each box coordinate is independent and follows a Gaussian distribution with mean b 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 modeled by the dirac delta distribution centered at the box coordinates. With these assumptions, KL Loss is proportional to:

$$L_{KL}(b, \sigma) \propto \frac{b^2}{2\sigma^2} + \frac{1}{2} \log \sigma^2. \quad (3.10)$$

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 review. Note that the choice of probability distribution for bounding boxes matters since the loss definition is affected by this choice. For example, Eq. 3.10 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* [91], where an object detector is successfully trained by directly formulating a loss based on the IoU as:

$$L_{IoU} = -\ln(\text{IoU}(\hat{b}_i, b_i)). \quad (3.11)$$

Another approach to exploit the metric-nature of $1 - \text{IoU}(\hat{b}_i, b_i)$ is the *Bounded IoU* loss [92]. This loss warps a modified version of $1 - \text{IoU}(\hat{b}_i, b_i)$ 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:

$$L_{B\text{IoU}}(x, \hat{x}) = 2L1_{\text{smooth}}(1 - \text{IoU}_B(\hat{b}_i, b_i)), \quad (3.12)$$

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 convention 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:

$$\text{IoU}_B(\hat{c}_x, c_x) = \max \left(0, \frac{w - 2|\hat{c}_x - c_x|}{w + 2|\hat{c}_x - c_x|} \right), \quad (3.13)$$

$$\text{IoU}_B(\hat{w}, w) = \min \left(\frac{\hat{w}}{w}, \frac{w}{\hat{w}} \right). \quad (3.14)$$

In such a setting, $\text{IoU}_B(\hat{b}_i, b_i) \geq \text{IoU}(\hat{b}_i, b_i)$. Also, an IoU based loss function is warped into the smooth $L1$ function in order to make the ranges of the classification and localisation 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) [93] showed that IoU can be directly optimised 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 $\text{IoU}=0$) by incorporating an additional smallest enclosing box of $\hat{b}_i$ and b_i as e_i . 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 $\text{IoU}(\hat{b}_i, b_i) > 0$. Unlike IoU, $\text{GIoU}(\hat{b}_i, b_i) \in [-1, 1]$. Having computed E , GIoU Loss is defined as:

$$\text{GIoU}(\hat{b}_i, b_i) = \text{IoU}(\hat{b}_i, b_i) - \frac{A(e_i) - A(\hat{b}_i \cup b_i)}{A(e_i)}, \quad (3.15)$$

where GIoU is a lower bound for IoU, and it converges to IoU when $A(\hat{b}_i, b_i) = A(e_i)$ and its loss form, GIoU Loss, is $L_{\text{GIoU}}(\hat{b}_i, b_i) = 1 - \text{GIoU}(\hat{b}_i, b_i)$. 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 0.50 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. [94] is to add penalty terms to the conventional IoU error (i.e. $1 - \text{IoU}(\hat{b}_i, b_i)$) 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 $\hat{b}_i$ and b_i is added as:

$$L_{\text{DIOU}}(\hat{b}_i, b_i) = 1 - \text{IoU}(\hat{b}_i, b_i) + \frac{d^2(\hat{c}_i, c_i)}{\bar{e}_i^2}, \quad (3.16)$$

where $d(\cdot, \cdot)$ is the Euclidean distance, c_i is the center point of box b_i and $\bar{e}_i$ is the diagonal length of e_i (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:

$$L_{\text{CIoU}}(\hat{b}_i, b_i) = 1 - \text{IoU}(\hat{b}_i, b_i) + \frac{d^2(\hat{c}_i, c_i)}{\bar{e}_i^2} + \alpha v, \quad (3.17)$$

such that

$$v = \frac{4}{\pi^2} \left(\arctan\left(\frac{w}{h}\right) - \arctan\left(\frac{\hat{w}}{\hat{h}}\right) \right)^2, \quad (3.18)$$

and

$$\alpha = \frac{v}{(1 - \text{IoU}(\hat{b}_i, b_i)) + v}. \quad (3.19)$$

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 analyse IoU-based loss functions (Section 3.4.5).

3.4.2 IoU Distribution Imbalance

Definition. IoU distribution imbalance is observed when proposals (i.e. anchor or RoI) have a skewed IoU distribution. The problem is illustrated in Fig. 3.12(a), where

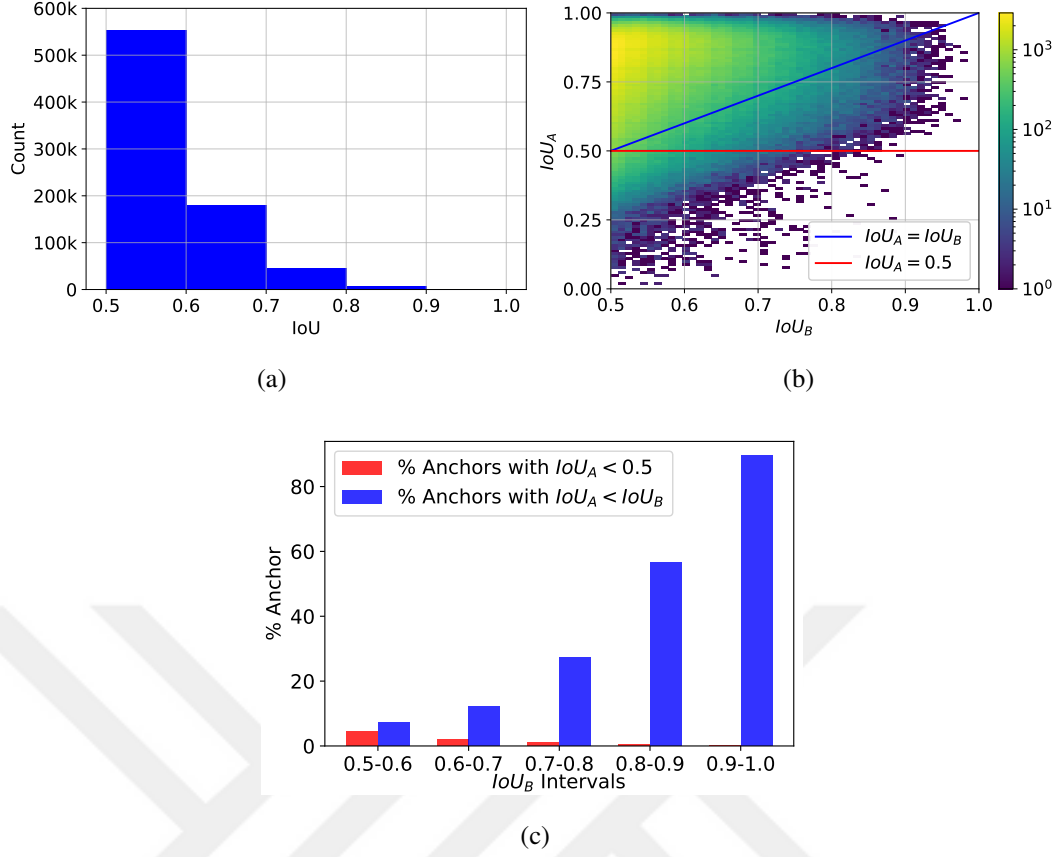


Figure 3.12: The IoU distribution of the positive anchors for a converged RetinaNet [18] with ResNet-50 [47] on COCO [13] before regression (a), and (b) the density of how the IoU values are affected by regression (IoU_B : before regression, IoU_A : after regression). For clarity, the density is plotted 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.

the IoU distribution of the anchors in RetinaNet [18] are observed to be skewed towards lower IoUs. It has been previously shown that this imbalance is also observed while training two-stage detectors [20]. Differently, we present in Fig. 3.12(b) 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. (Fig. 3.12(c)). 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% of the positive anchors are lost by the regressor (Fig. 3.12(c)). 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 $0.9 - 1.0$. These results suggest that there is still room to improve by observing the effect of the regressor on the IoUs of the proposals (i.e. anchor or RoI).

Solutions. The *Cascade R-CNN* method [20] 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 [134] and AttracNet [135], that iteratively apply the same network to the bounding boxes. Another cascade-based structure implicitly addressing the IoU imbalance is *Hierarchical Shot Detector* (HSD) [95]. 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 proposals (i.e. anchor or RoI) with balanced IoU distribution to the second stage of Faster R-CNN. *IoU-uniform R-CNN* [96] adds controllable jitters and in such a way provides approximately uniform positive proposals (i.e. anchor or RoI) to the regressor only (i.e. the classification branch still uses the RPN RoIs). Differently, *pRoI Generator* [74] 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. [74] 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 proposals (i.e. anchor or RoI) has an effect not only on the regression but also on the classification performance. (ii) Similar to the finding of Pang et al. [32], the IoU of the examples is related to their hardness. However, contrary to Cao et al. [33], who argued that OHEM [27] has an adverse effect when applied only to positive examples, Oksuz et al. [74] showed that the effect of OHEM depends on the IoU distribution of the positive proposals. 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.

3.4.3 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 (Fig. 3.13), 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. [75] 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

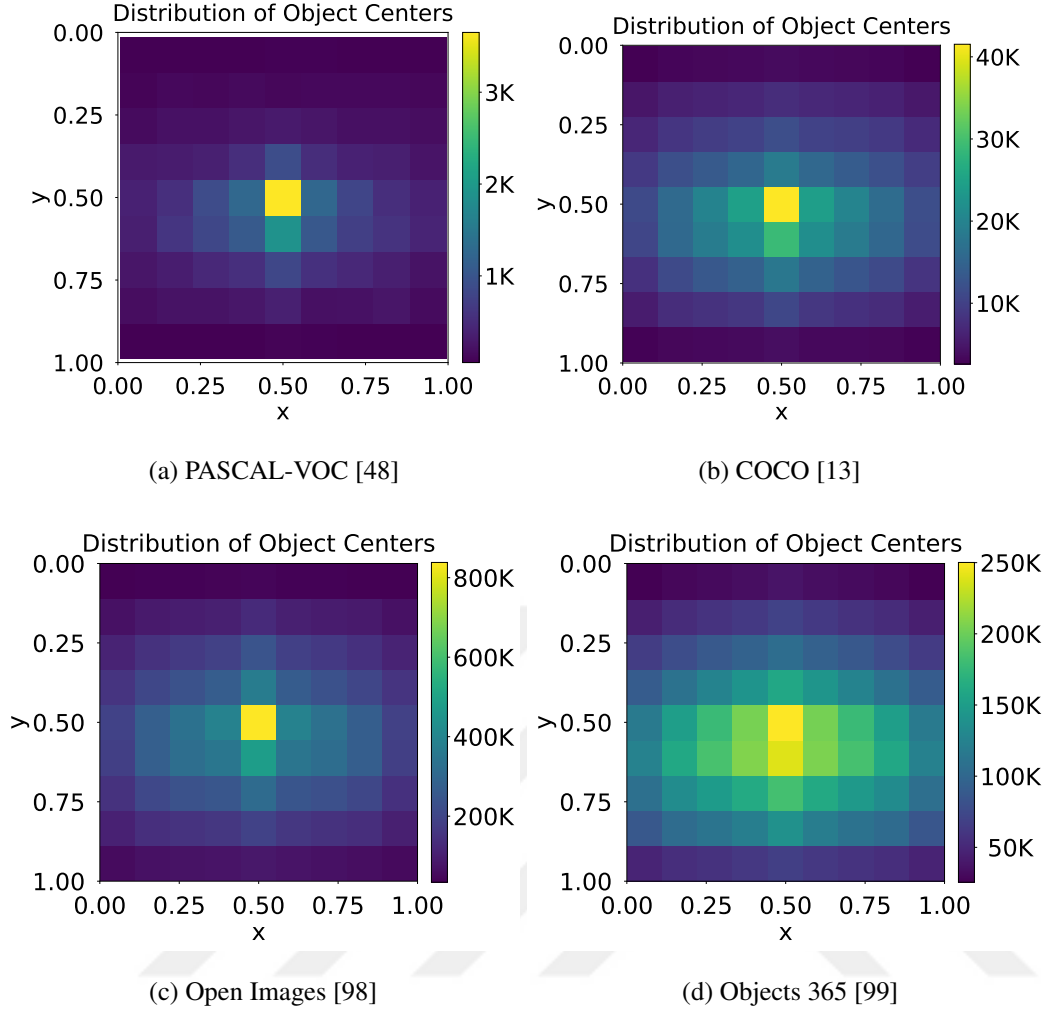


Figure 3.13: Distribution of the centers of the objects in the common datasets over the normalized image.

anchor-guided feature adaptation based on deformable convolutions [136] in order to have a balanced representation depending on the anchor size. Rather than learning the anchors, *free anchor* method [39] 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.

3.4.4 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. [100] 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% respectively. With the same configuration, DIOU and CIOU losses are reported to have a relative improvement of 0.2% and 1.7% against the GIoU baseline (without DIOU-NMS). Compared to the loss functions designed for the foreground-background imbalance problem (i.e. Focal Loss - Section 3.2.1.2), the relative improvement of the regression losses over Smooth $L1$ are smaller. We also analysed for which cases the baseline loss function fails in Fig. 3.12(b), 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% on 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% 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 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_{50} , it yields better performance for AP_{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 [75] increases average recall by 9.1% with 90% fewer anchors via learning the parameters

of the anchors. Compared to guided anchoring, free anchor [39] reports a lower relative improvement for average recall with 2.4% against RetinaNet with ResNet50, however it has a 8.4% relative improvement (from 35.7 to 38.7).

3.4.5 Open Issues

This section discusses the open issues related to the spatial properties of the proposals (i.e. anchor or RoI) and objects.

3.4.5.1 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 [32] increases the contribution of the inliers. (ii) KL Loss [90] 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.

3.4.5.2 Analyzing the Loss Functions

In order to analyse how outliers and inliers affect the regression loss, it is useful to analyse the loss function and its gradient with respect to the inputs. To illustrate such an analysis, in Focal Loss [18], 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 [32], 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 [37] 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 [93], which uses the ground truth box and the smallest enclosing box in addition to the detection box. Each box is represented by four parameters (Section 3.4.1), 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.

3.4.5.3 Designing Better Anchors

Designing an optimal anchor set with high recall has received limited attention. The Meta Anchor [137] method attempts to find an optimal set of aspect ratios and scales for anchors. More recently, Wang et al. [75] have improved recall more than 9% on COCO dataset [13] while using 90% less anchors than RPN [19]. Addressing the imbalanced nature of the locations and scales of the objects seems to be an open issue.

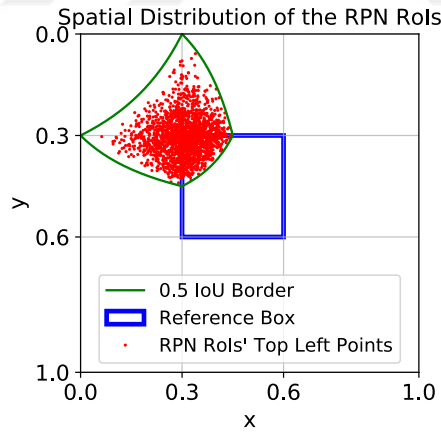


Figure 3.14: The (relative) spatial distributions of $1K$ RoIs with IoU between 0.5 to 0.6 from the RPN (of Faster R-CNN with ResNet101 backbone) collected from Pascal [48] during the last epoch of the training. A bias is observed around the top-left corners of ground truths such that RoIs are densely concentrated at the top-left corner of the normalized ground truth box.

3.4.5.4 Relative Spatial Distribution Imbalance

Open Issue. As we discussed in Section 3.4, 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 [74] 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, proposals (i.e. anchor or RoI) 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 proposals (i.e. anchor or RoI). The work by Oksuz et al. [74] 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 box representing the ground truth (Fig. 3.14). They reported that the resulting spatial distribution of the top-left points of the RPN RoIs 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.

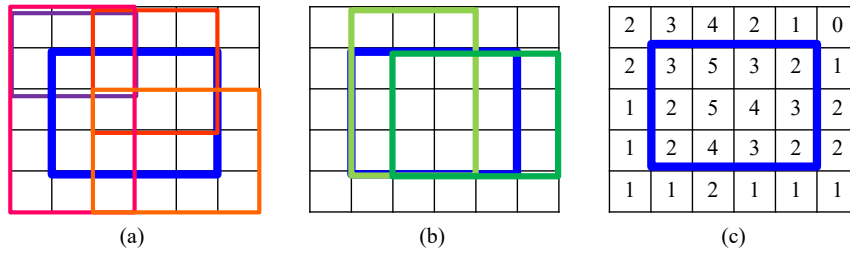


Figure 3.15: An illustration of the imbalance in overlapping BBs. The grid represents the pixels of the image/feature map, and blue bounding box is the ground-truth. **(a)** Four negative proposal boxes (i.e. anchor or RoI). **(b)** Two positive proposal boxes. **(c)** Per-pixel number-of-overlaps of the proposal boxes. Throughout the image, the number of sampling frequencies for the pixels vary due to the variation in the overlapping number of bounding boxes.

3.4.5.5 Imbalance in Overlapping BBs

Open Issue. Due to the dynamic nature of bounding box sampling methods (Section 3.2.1), 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 Fig. 3.15(a-c) on an example grid representing the image and six proposals (four negative and two positive). The number of overlapping BBs for each pixel is shown in Fig. 3.15(c); 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 [138].

3.4.5.6 Analysis of the Orientation Imbalance

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

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.

3.5 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

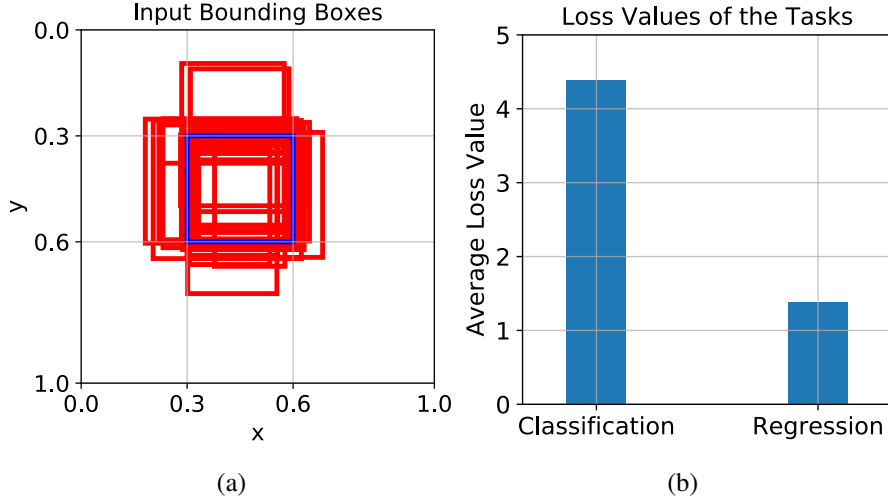


Figure 3.16: **(a)** Randomly sampled 32 positive RoIs using the pRoI Generator [74]. **(b)** Average classification and regression losses of these RoIs at the initialization of the object detector for COCO dataset [13] with 80 classes. We use cross entropy for the classification task assuming that initially 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.

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 (Fig. 3.16). (ii) The ranges of the loss functions from different tasks can be different, which hampers the consistent and balanced optimisation 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 [139].

Fig. 3.16 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 AP Loss is between 0 and 1. Another example is the GIoU Loss [93], which is in the $[0, 2]$ 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) [33], 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:

$$L_{CARL}(x) = c'_i L_{smooth}(x), \quad (3.20)$$

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 localisation 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 3.5.2, this correlation should be investigated and exploited more extensively.

Recently, Chen et al. [63] 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:

$$\frac{w_{reg} L_{Reg}}{L_{cls}}. \quad (3.21)$$

The motivation of the method is that regression loss consists of only foreground ex-

amples and is normalized only by number of foreground classes, therefore it can be used as a normalizer for the classification loss.

3.5.1 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 [63]. While many studies use equal weights for the regression and classification tasks [24, 18], it is also shown that appropriate linear weighting can lead to small improvements on the performance [100]. However, an in-depth analysis on objective imbalance is missing in the literature.

CARL [33] is a method to promote examples with higher IoUs, and in such a way, 1.6% relative improvement is achieved compared to prime sampling without CARL (i.e. from 37.9 to 38.5). Another method, Guided Loss [63], 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.

3.5.2 Open Issues

Open Issue. Currently, the most common approach is to linearly combine the loss functions of different tasks to obtain the overall loss function (except for classification-aware regression loss in [33]). However, as shown in Fig. 3.18(a-c), as the proposal box (i.e. anchor or RoI) 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* [37] did not modify the regression branch;

however, AP_{75} increased around 3%. This example shows that the loss functions for different branches (tasks) are not independent (see also Fig. 3.18). This interdependence of tasks has been explored in classification-aware regression loss by Cao et al. [33] (as discussed in Section 3.5) 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 [139] pointed out that learning pace affects performance. With this in mind, we plotted the regression and classification losses of the RPN [19] during training on the Pascal VOC dataset [48] in Fig. 3.17. 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.

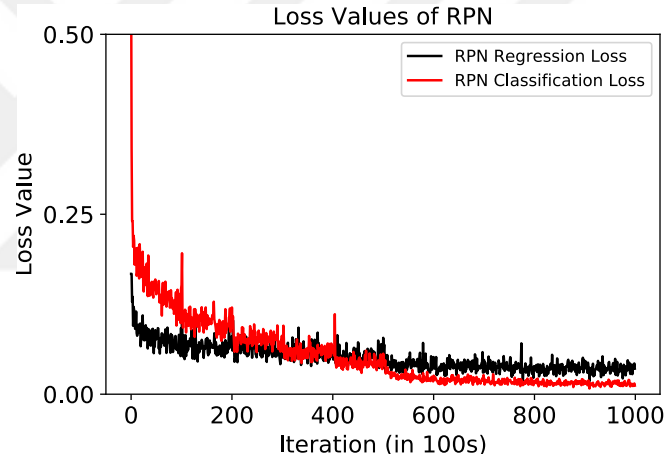


Figure 3.17: Paces of the tasks in RPN (with Resnet-101 [47] backbone) [19] on Pascal VOC 2007 training set [48].

3.6 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.

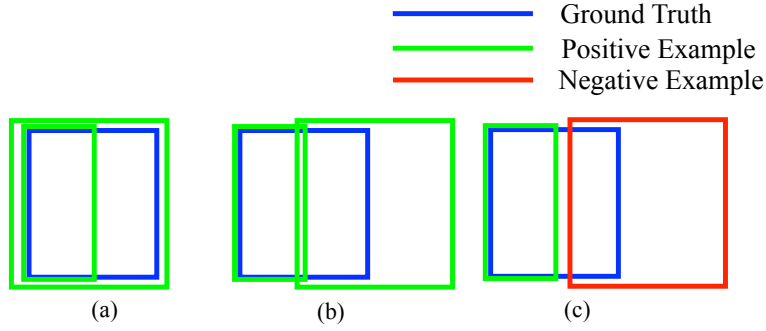


Figure 3.18: An example suggesting the necessity of considering different imbalance problems together in a unified manner. Blue, green and red colors indicate ground-truth, positive example (prediction) and negative examples, respectively. The larger example (i.e. the one with higher IoU with the blue 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.

3.6.1 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. 3.18. In this figure, we shift a proposal box (i.e. anchor or RoI) with a high IoU (Fig. 3.18(a)) to worse qualities in terms of IoU in two steps (Fig. 3.18(b-c)) and observe how this shift affects the different imbalance problems. For the base case in Fig. 3.18(a), there are two positive bounding boxes (relevant for class imbalance) with different scales (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. 3.18(b), 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 [27, 32]), and less prime samples [33]. 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. 3.18(c), 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 [27, 32], and the criterion that the prime sample attention [33] 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.

3.6.2 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 [27] favors a right-skewed IoU distribution towards 0.5 [32], whereas prime sample attention prefers the positive examples with larger IoUs [33] 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.

3.6.3 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 [22], Retina Net [18] and RPN [19] label proposals 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 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 proposals (i.e. anchor or RoI) 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. [68] showed that this scheme results in a large number of noisy examples. In Fig. 3.19, we illustrate two proposals 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.

3.6.4 Imbalance in Bottom-Up Object Detectors

Open Issue. Bottom-up detectors [26, 40, 97, 140] adopt a completely different approach to object detection than the one-stage and the two-stage detectors. Bottom-up detectors might share many of the imbalance problems seen in the top-down detec-

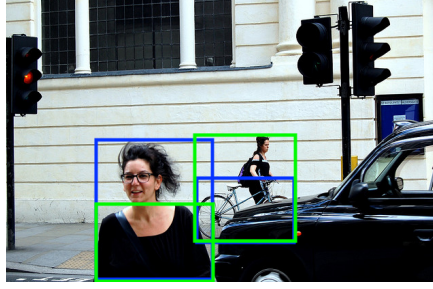


Figure 3.19: An illustration on ambiguities resulting from labeling examples. The blue boxes denote the ground truth. The green boxes are the estimated positive boxes with $IoU > 0.5$. The input image is from the COCO [13].

tors, 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 [26] and ExtremeNet [97] use focal loss [18] to address foreground-background class imbalance, and the hourglass network [109] 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. [27] show that the examples with larger losses are important; from the scale imbalance perspective, different architectures [32, 82, 88] and training methods [30, 31] involving feature and image pyramids are proven to be useful and finally from the objective imbalance perspective, Pang et al. [32] 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.

3.7 Conclusion

In this chapter, 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.





CHAPTER 4

BOUNDING BOX GENERATOR TO ANALYSE IMBALANCE PROBLEMS IN VISUAL DETECTION TASKS

In this chapter, we present our Bounding Box Generator as a basic operation to devise analysis tools for object detection. Then based on our Bounding Box Generator, we devise a Positive RoI Generator to analyse imbalance problems for the second stage of Faster R-CNN. This chapter is based on our work [74],

- Kemal Oksuz, Baris Can Cam, Emre Akbas* and Sinan Kalkan*, “Generating Positive Bounding Boxes for Balanced Training of Object Detectors”, IEEE Winter Conference on Applications of Computer Vision (WACV), 2020.

We only make minor changes to fit the text appropriately in the context of this thesis.

4.1 Introduction

An important challenge in object detection is class imbalance [18, 27, 32, 33, 68]: even from a single image, an infinite number of negative examples can be sampled, in contrast to only a limited set of positive RoIs (regions-of-interest). Naturally, this leads to significant imbalance between negatives and positives. Class imbalance also exists within foreground classes.

A prominent solution to the foreground-background class imbalance is to have two stages [19, 22, 23]: The first stage estimates regions (i.e., RoIs) that are likely to contain objects, significantly discarding background samples, and the second-stage

* Equal contribution for senior authorship.

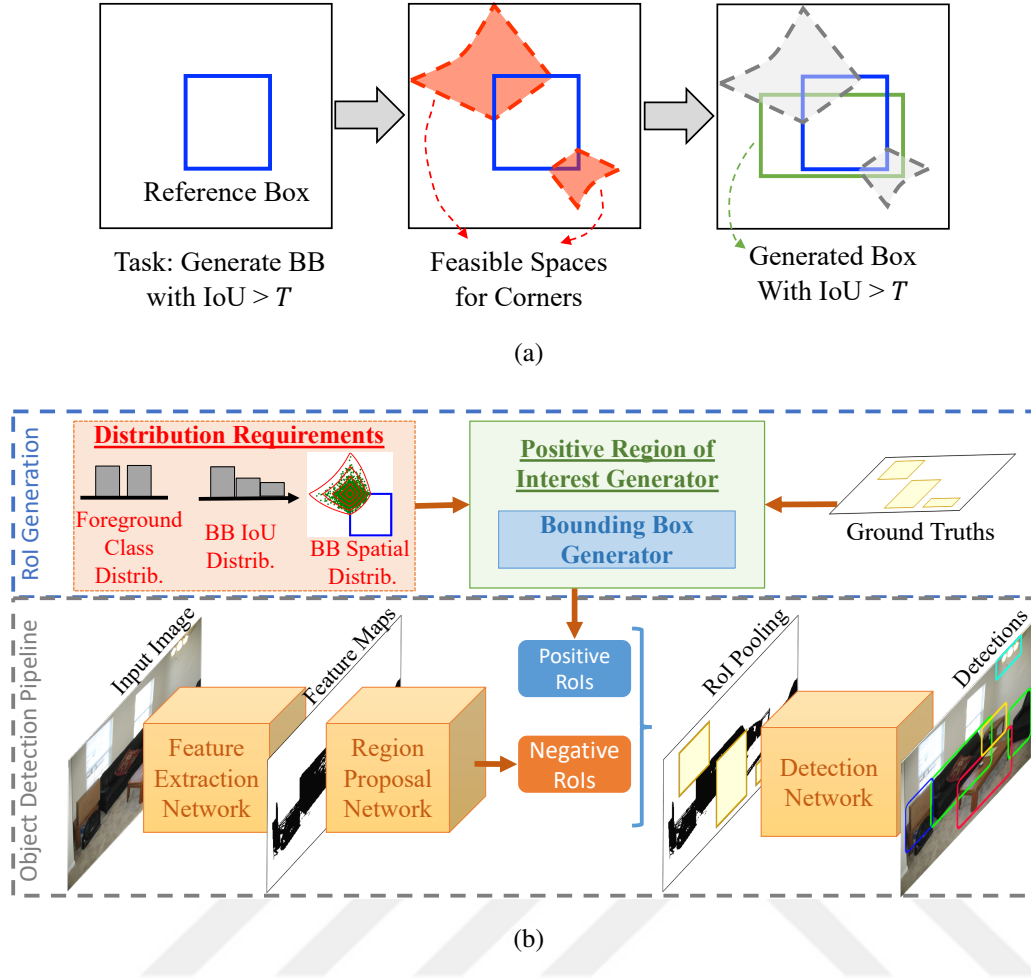


Figure 4.1: **(a)** An illustration of Bounding Box (BB) Generation. Given a reference box (in blue) and an IoU threshold T , a BB having at least T IoU is generated (drawn in green). **(b)** An illustration of training an object detector with positive region-of-interests. Given distribution requirements on foreground classes and BBs, we generate positive RoIs using the BB generator. Negative RoIs are still generated by the region proposal network.

classifies these regions into objects, and also fine-tunes the coordinates of the bounding boxes. Other solutions generally employ sampling with hard constraints (e.g., online hard example mining [27], Libra RCNN [32]) or soft constraints (e.g., focal loss [18], harmonizing gradients [68]).

The foreground-foreground class imbalance problem, i.e., the imbalance in the number of examples pertaining to different positive classes at the image, dataset or mini-

batch levels, has not attracted as much attention. In addition, the IoU distribution of the RoIs generated by the region proposal network (RPN) [19] is imbalanced [20], which biases the BB regressor in favor of the IoU that the distribution is skewed towards. We call this imbalance problem as *IoU distribution imbalance* (see also Section 3.4.2 for further discussion). Addressing these problems requires a careful analysis of the positive RoIs.

In this chapter, we analyse and address foreground-foreground class imbalance and IoU distribution imbalance by actively generating BBs. We first propose the “Bounding Box Generator”, a method that can generate an arbitrary BB overlapping with a reference box with an IoU larger than a given threshold (Fig. 4.1(a)). Using the BB generator, we develop a positive RoI (pRoI) generator that can produce RoIs conforming to desired foreground class, IoU and relative spatial distributions (Fig. 4.1(b)). Considering that there is a correlation between the hardness of an example and its IoU [32], the pRoI generator can *generate* (rather than sample) not only positive samples, but also samples with any desired property such as hard examples [27] or prime samples [33].

Devised based on Bounding Box Generator, our pRoI generator can perform several analyses and improvements. Specifically, we (i) show that IoU and foreground class distributions affect performance, (ii) make a comparative analysis for RPN RoIs and (iii) improve the performance of Faster RCNN for IoU intervals where RPN is not able to generate enough samples.

Finally, we devise an online, foreground-balanced (OFB) sampling method which considers the imbalance among the foreground classes dynamically within a training batch based on multinomial sampling.

Contributions. Overall, our main contributions in this chapter are as follows:

1. *Generators:* (i) A BB generator to generate BBs for a given IoU threshold and (ii) a positive RoI generator to generate RoIs with desired foreground class, IoU and relative spatial distributions.
2. *Imbalance Problems and Analysis:* Using our pRoI generator, we show that IoU distribution and foreground-foreground class imbalance within a training batch affect

the performance of the object detectors. We also provide an analysis of RPN RoIs and show that the effect of the hard examples depends on the IoU distribution of the BBs.

3. Practical Improvements: We train a detection network using our pRoI generator, which increases the amount and the diversity of the positive examples especially for the larger IoUs, and show that the performance improves compared to the standard training (e.g. for $IoU = 0.8$, $mAP@0.8$ improves by 10.9% for Pascal VOC). We also train the conventional detection pipeline by using the proposed OFB sampling, and improve the performance.

4.2 Related Work

Deep Object Detectors: We can group deep object detectors into two: One-stage methods and two-stage methods. While one-stage methods [17, 18, 24, 25] predict the object categories and their BBs directly from anchors, two-stage methods [19, 21, 22, 23] first estimate a set of RoIs from anchors and then predict objects from these RoIs in the second stage. Both approaches use a deep feature extractor [47, 141], optionally followed by steps like feature pyramid networks [29, 82, 85, 88].

Our BB sampling approach is more suitable for the second stage of the two-stage methods since one-stage detectors have structural constraints owing to the fact that each output of a one-stage detector corresponds to a predefined anchor having fixed location, shape and scale. For this reason, an additional module is required to employ our generator. However, having balanced IoU and foreground class distributions are relevant for any object detection pipeline since any object detector needs to deal with BBs even if they are estimated or fixed (in the case of anchors).

Class Imbalance in Object Detection: Following Oksuz et al. [142], we categorize the class imbalance problem for the deep object detectors into two: foreground-background and foreground-foreground class imbalance.

Foreground-background class imbalance has attracted more attention with *hard sampling*, *soft sampling* and *generative approaches*. In hard sampling methods, certain

samples are shown more to the network. This can be performed via random sampling [19, 23], or by relying on “sample usefulness” heuristics as in hard-example mining [24, 27, 32] and prime sampling [33]. Hard-example mining methods usually assume that examples with higher loss are more difficult to learn, and therefore, they train a network more with such examples. This approach is adopted for negative samples in SSD [24], while a more systematic approach considering both the positive and negative samples is proposed in online hard example mining (OHEM – [27]). An alternative hardness definition was proposed in Libra R-CNN [32] based on a sample’s IoU, and a solution was proposed using hard example mining using BB IoUs without computing the loss for the entire set. A recent interesting method, “prime sampling” [33], asserts that positive samples with higher IoUs are more representative and proposed ranking the positive samples based on its IoU with the ground truth, while still showing that hard example mining for the negative class works well. BB IoU imbalance is addressed by Cascade R-CNN [20] by employing cascaded detectors in such a way that a later-stage detector is trained by a distribution skewed towards higher IoU.

In *soft sampling*, a weight is assigned to each sample rather than performing a discrete (hard) selection of samples. Prominent examples include focal loss [18], which promotes hard examples; prime sampling [33], which assigns more weight to examples with higher IoUs; and finally gradient harmonizing mechanism [68], which assigns lower weights to easy negatives and suppresses the effect of the outliers.

The *generative methods* address imbalance with a different perspective by introducing generated samples. Example approaches include generating hard examples with various deformations and occlusion [71] and generating synthetic examples [143].

Foreground-foreground class imbalance is critical as well. Kuznetsova et al. [98] showed that object detection datasets are highly imbalanced also for foreground classes. The only method to consider the problem at the dataset level handcrafts a similarity measure, and based on the measure clusters the classes to have a more balanced training [28]. In the classification domain where there is no background class, this imbalance is studied more [61, 62] by, e.g., performing class-aware sampling [144]. However, these methods are not directly applicable for two-stage object detectors because the second stage’s input is very dynamic since it depends on RoIs estimated by

the first stage. Despite this difference, class-aware sampling is said to be adopted by [82], however no comparison is presented for balanced and imbalanced training from the object detection perspective.

Our ideas in this paper are relevant for both foreground-background and foreground-foreground class imbalance. One can generate any number of positive RoIs to address the foreground-background imbalance, and the generated set can also be chosen equally from each class to address the foreground-foreground imbalance. Among the three types of methods mentioned above, we classify our approach as a generative method. Since the end-to-end training pipeline is not disrupted (Fig. 4.1(b)), any hard sampling method [27, 32] can also be simulated. In addition, we directly address foreground-foreground class imbalance by online foreground balanced (OFB) sampling. Its main difference from the previously proposed class-aware sampling [144] is that while they use a static dataset, our OFB sampling is able to handle the dynamic nature of the RoIs (i.e. the batch depends on the sampled RoIs at each iteration) owing to the proposal network.

4.3 The Generators

In this section, we describe the methods for generating bounding boxes and balanced positive RoIs.

4.3.1 Bounding Box Generator

Given a reference bounding box $B = [x_1, y_1, x_2, y_2]$ and a threshold T , the goal of the BB generator is to determine a new box $\bar{B} = [\bar{x}_1, \bar{y}_1, \bar{x}_2, \bar{y}_2]$ such that $\text{IoU}(B, \bar{B}) \geq T$. To generate such a box, we propose a 2-step algorithm presented in Algorithm 1 and illustrated in Fig. 4.2. The first step (lines 3-4) finds the polygon¹ that computes the feasible space for top-left point of the generated BB ($\text{TL}(\bar{B}) = (\bar{x}_1, \bar{y}_1)$), which satisfies the desired IoU, and samples a point in this polygon. The second step (lines 6-7) takes into account the sampled $\text{TL}(\bar{B})$ and, similar to Step 1, determines a feasi-

¹ Note that the shape is not strictly a polygon; however, we approximate it as one at regular small intervals, and therefore, we call it a polygon for the sake of simplicity.

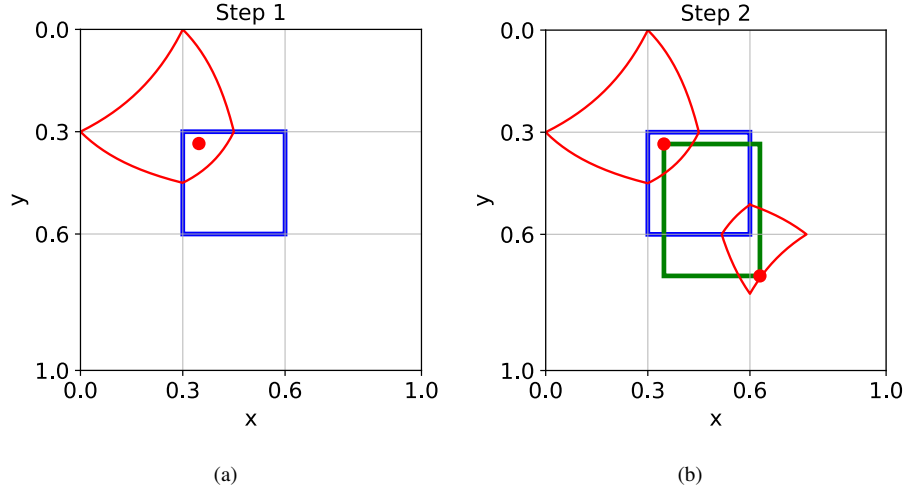


Figure 4.2: **(a,b)** Applying Algorithm 1 on the blue BB (B) with $T = 0.5$. Red polygons denote boundaries for top-left and bottom-right points that can be sampled with an IoU larger than $T = 0.5$. Red dots are sampled points, and green box is the generated box ($\bar{B}$) with $IoU = 0.5071$.

Algorithm 1 Bounding Box Generator. See Section 4.3.1 and the Appendix B for the definitions of the functions.

```

1: procedure GENERATEBB( $B, T$ )
2:   # Step-1: Find top-left corner
3:    $TLPoly \leftarrow \text{findTLFeasibleSpace}(B, T)$ 
4:    $TL(\bar{B}) \leftarrow \text{samplePolygon}(TLPoly)$ 
5:   # Step-2: Find bottom-right corner
6:    $BRPoly \leftarrow \text{findBRFeasibleSpace}(B, T, TL(\bar{B}))$ 
7:    $BR(\bar{B}) \leftarrow \text{samplePolygon}(BRPoly)$ 
8:   return  $[TL(\bar{B}), BR(\bar{B})]$ 
9: end procedure

```

ble space for bottom-right corner, then, samples bottom right-point of the generated bounding box ($BR(\bar{B})$). This order leads to a non-isotropic distribution with respect to the reference box. To make it isotropic, we can also sample in the reverse order: i.e. sample BR first then TL. We then randomly choose the order, before sampling. Fig. 4.3 superimposes 1000 generated boxes with $T = 0.6$.

The following two sections discuss how the feasible space is computed (i.e. find-

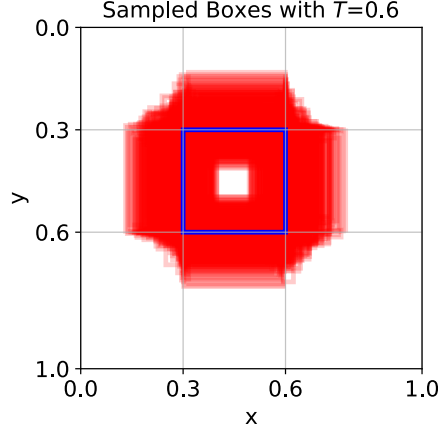


Figure 4.3: $1K$ generated boxes (shown with red) by Algorithm 1 for reference box drawn in blue (B) and IoU threshold $T = 0.6$.

ing top left feasible space) and how a point can be sampled within a polygon (i.e. sampling within a polygon). See the Appendix B for bottom-right point, $BR(\bar{B})$.

4.3.1.1 Determining Feasible Space for the Desired IoU

$\text{findTLFeasibleSpace}(B, T)$ is the function determining the feasible set of points that can be the top left point of a box ensuring the desired IoU. In order to find the set of these feasible points (i.e. $TL(\bar{B})$) that satisfy Eq. 2.2, we assume that $BR(\bar{B}) = BR(B)$ and manipulate Eq. 2.2, otherwise, some feasible points are excluded in the feasible top left space. Even though $BR(\bar{B})$ is fixed, there are still two unknown variables $\bar{x}_1$ and $\bar{y}_1$. That's why, we first bound one of these two variables and then find the value of the unbounded variable by moving within the limits of the bounded variable with some precision (we use 0.0001 as precision). Since the definition of the $\text{IoU}(B, \bar{B})$ is different in each of the four regions depicted in Fig. 4.4(a) due to the max and min operations, an equation is to be derived for each region.

Denoting the minimum and maximum bounds of $\bar{x}_1$ in Region I by x_{min}^I and x_{max}^I respectively, we bound the values in x axis. It is obvious that $x_{min}^I = x_1$ due to the boundary of Region I. To find x_{max}^I , we manipulate the definition of IoU (Eq. 2.2) by exploiting that $\bar{y}_1 = y_1$ for x_{max}^I , which yields:

$$x_{max}^I = x_2 - (x_2 - x_1) \times T. \quad (4.1)$$

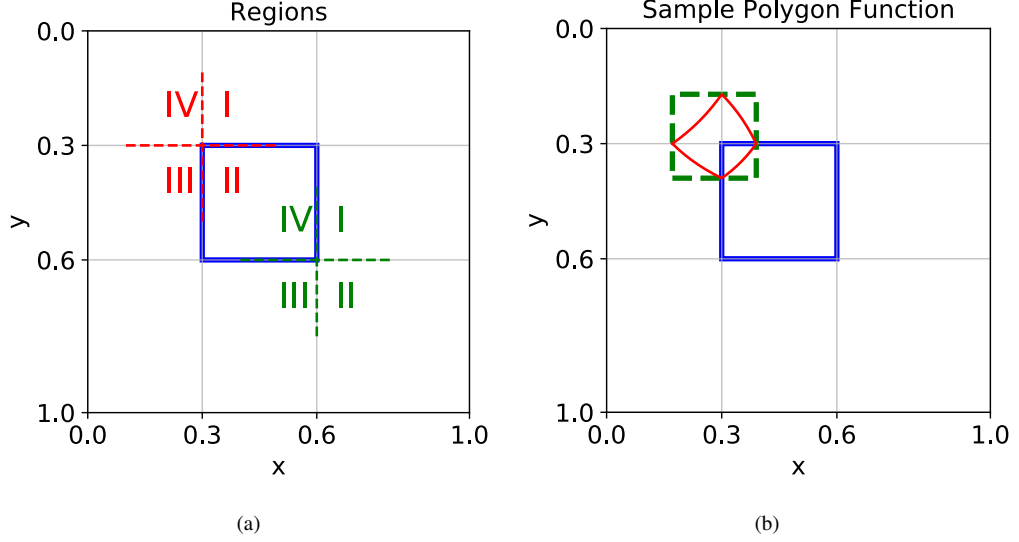


Figure 4.4: **(a)** The regions around $TL(B)$ and $BR(B)$ are splitted into four each. Red and green dashed lines split the top left and bottom right regions respectively. The numbers label the splitted regions.), **(b)** In the execution of the sample polygon function for $T = 0.75$, green dashed box is the enclosing box for the TL space polygon.

Having determined the boundaries for $\bar{x}_1$, now we derive a function that determines $\bar{y}_1$ given $\bar{x}_1$. Finally, moving within the bounds yields $\bar{x}_1, \bar{y}_1$ pairs satisfying $\text{IoU}(B, \bar{B}) = T$ when $BR(\bar{B}) = BR(B)$. In region I, note that $A(B \cap \bar{B})$ does not rely on $\bar{y}_1$ (i.e. $A(B \cap \bar{B}) = (x_2 - \bar{x}_1)(y_2 - y_1)$). Bringing these together, $\bar{y}_1$ can be defined as (see Appendix B for the entire derivation of x_{max}^I and $\bar{y}_1$):

$$\bar{y}_1 = y_2 - \frac{\frac{A(B \cap \bar{B})}{T} + A(B \cap \bar{B}) - A(B)}{(x_2 - \bar{x}_1)}. \quad (4.2)$$

Here, we only show the derivation steps for Region I and present the equations for all regions in Appendix B. Combining the points in all these regions yields the polygon limiting feasible region with $\text{IoU} \geq T$.

4.3.1.2 Controlling the Relative Spatial Distribution of the Boxes

`samplePolygon(TLPoly)` function determines the BB spatial distribution. We follow rejection sampling [145] in such a way that a point is proposed by the proposal

distribution until it hits the inside of the polygon. Accordingly, the proposal distribution determines the BB spatial distribution. Fig. 4.4(b) presents an example for spatial uniform distribution for the top-left space polygon with $T = 0.75$. We sample a point in the rectangle uniformly, which corresponds basically to generating two uniform numbers within a range. If the point is in the polygon, then it is accepted, else a new point is proposed until it is inside the polygon. Note that different proposal distributions lead to different relative spatial distributions for the generated BBs.

4.3.2 pRoI Generator: Training by Generated BBs

This section provides an application of our BB generator for generating positive RoIs for training a two-stage object detector. By applying our BB generator to the ground-truth boxes, we can generate positive RoIs with desired characteristics. This enables us to (i) analyse how the performance of Faster R-CNN is affected by the properties of the positive RoIs and (ii) improve the performance for IoU intervals where RPN is not able to generate enough samples.

Algorithm 2 Positive RoI Generator. See Section 4.3.2 and Appendix B for the definitions of functions `fgBalancedRoIAlloc` and `genRoIs`.

```

1: procedure GENERATEPROI( $GTs, \psi_{IoU}, W_{IoU}, RoINum$ )
2:    $perGtRoI = \text{fgBalancedRoIAlloc}(GTs, RoINum)$ 
3:    $RoIs = \text{genRoIs}(GTs, perGtRoI, \psi_{IoU}, W_{IoU}, RoINum)$ 
4:   return  $RoIs$ 
5: end procedure

```

The method, “Positive RoI Generator” (pRoI Generator), described in Algorithm 2, can control several different characteristics of the set of positive RoIs:

- `fgBalancedRoIAlloc()` first divides $RoINum$ by the number of different classes in the given ground truth set, GTs , to determine the allocated box number per class, and then shares this value among each example of the same class equally. As a result, `fgBalancedRoIAlloc()` determines the number of boxes to be generated for each ground truth box in GTs .

- Secondly, given the allocated number of boxes for each ground truth, `genRoIs()` iteratively uses BB generator as a subroutine to provide a set of RoI_{Num} RoIs. In this step, the IoU distribution requirement is determined by the inputs ψ_{IoU} , the base of the IoU bins and the weight of the each bin denoted by W_{IoU} . W_{IoU} is basically a multinomial distribution over the bins determined by ψ_{IoU} .

An important benefit of pRoI generator is that training with the generated RoIs has no impact on the gradient flow for the training process. At each training iteration, RPN generates a set of RoIs among which we discard the positive ones and use the positive RoIs generated by the proposed method (Fig. 4.1). Using our pRoI generator, we can address the imbalance problems regarding RoIs at three different levels:

(1) Foreground-foreground class imbalance, which occurs when a dataset or mini-batch (or batch) contains different numbers of positive examples from different classes. To illustrate on a batch, an image (used as a batch) from PASCAL dataset [48] includes 4 bottles, 2 persons, 2 dining tables and 1 chair. In such a case, having equal number of RoIs per instance may lead the model to be biased in favor of the bottle class while ignoring the chair class. In our pRoI Generator, `fgBalancedRoIAlloc()` function allocates the same number of RoIs for each class within the batch.

(2) IoU distribution imbalance, which occurs when the positive RoIs have a skewed IoU distribution (Fig. 4.5). It has been shown that the hardness of a RoI is related to its IoU [32] and also the regressor overfits to RoIs which has IoU around 0.5 when the distribution of the RPN proposals is concentrated towards 0.5 [20]. Thus, these recent findings imply that the IoU distribution has an important effect on training. As aforementioned, `genRoIs()` is able to control the IoU distribution of the BBs.

(3) Relative spatial imbalance, which occurs when the BBs intersect significantly and a diverse set of examples can not be provided to the detection network. This level of imbalance is controlled in our pRoI generator in the subroutine BB generator as discussed in Section 4.3.1.2.

4.4 Experimental Setup

Dataset and Implementation Details: We evaluate our generative methods on Faster R-CNN in two different settings: (i) on Pascal VOC 2007 [48] with backbone ResNet-101 following the implementation and training in [146] with batch size 1 image on 1 GPU, and (ii) on COCO [13] with backbone ResNet-50 following the implementation and training in [100] with batch size 2 images/GPU on 2 GPUs. During training, 32 positive, and 96 negative RoIs are used from each image in the batch.

Performance Measures: We exhaustively search for the best AP at IoU=0.50 (AP_{50}) and optimal LRP (oLRP) error [107] values over epochs and report them. oLRP is a recently introduced metric for object detection, which represents recall, precision and average tightness of the BBs. Note that AP is a higher-is-better measure, while oLRP is an error metric and thus, it is a lower-is-better measure.

RoI Sources: In addition to RoIs output by RPN, we use the RoIs generated by our pRoI generator, with a given distribution, during the analysis and training. The different distributions are obtained by controlling W_{IoU} in Algorithm 2. Unless otherwise stated, we set $\psi_{IoU} = [0.5, 0.6, 0.7, 0.8, 0.9]$ and $RoI_{Num} = 32$. We train these RoI sources with and without foreground balanced sampling in order to see the effects of different imbalance problems on different RoI sources. The results are presented in Table 4.1.

4.5 Imbalance Problems and Analysis of RPN RoIs

In this section, using our pRoI generator, we show that IoU and foreground class distributions affect performance, simulate a sampling method and analyse the relative spatial distribution of RPN RoIs.

4.5.1 IoU Distribution Imbalance

Our BB generator method (Algorithm 1) samples boxes for a given IoU threshold, spatially uniformly. It does not impose an upper bound for the IoUs of the sampled

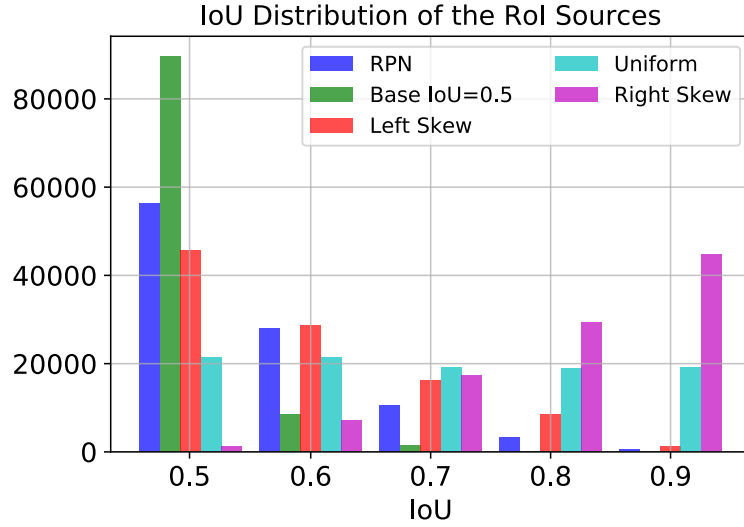


Figure 4.5: IoU distribution of different RoI Sources.

Table 4.1: Effect of the batch properties for generated positive samples (see Fig. 4.5 for different RoI sources) on Pascal VOC 2007. We trained each RoI source with balanced foreground-foreground distribution and simulating OHPM. RS, Unif, LS and Base respectively denote pRoI-Right Skew, pRoI-Uniform, pRoI-Left Skew and pRoI-Base IoU=0.5 distributions. FGB refers to foreground balanced generation of RoIs.

RoI Distrib.	<i>FGB?</i>	OHPM	oLRP ↓	oLRP _{Loc} ↓	oLRP _{FP} ↓	oLRP _{FN} ↓	AP ₅₀ ↑
RS	No	No	64.6	21.4	18.7	29.8	74.9
	Yes	No	64.5	21.5	18.7	29.5	75.3
	Yes	Yes	60.4	19.5	16.8	27.2	77.4
Unif.	No	No	61.3	19.5	17.9	28.5	76.3
	Yes	No	61.1	19.5	17.0	28.8	76.9
	Yes	Yes	59.9	19.2	16.0	27.6	77.8
LS	No	No	60.4	19.1	16.9	28.3	77.0
	Yes	No	60.3	19.0	17.3	28.2	77.2
	Yes	Yes	60.7	19.3	17.7	27.8	76.9
Base	No	No	61.5	19.7	17.2	28.8	76.6
	Yes	No	61.4	19.3	16.3	29.4	76.7
	Yes	Yes	61.2	19.7	16.6	28.6	76.7

boxes. Therefore, in order to analyse the density of the different IoUs for the positive samples, we uniformly generate $100K$ boxes for each IoU distribution type and plot the distribution of the generated boxes in Fig. 4.5. Note that training a detector with different IoU distributions of positive examples affects the resulting test performance (Table 4.1), which implies the effect of IoU distribution imbalance.

From Fig. 4.5, we observe the following: **(1)** The distribution of the boxes with $baseIoU = 0.5$ is highly biased towards 0.5 and includes very low samples with higher IoUs. This implies that the proportion of the boxes with $IoU > 0.9$ is far too low than that of the boxes with $0.6 > IoU > 0.5$ when $T = 0.5$. **(2)** RPN RoIs follow a similar tendency to the sampled boxes with $baseIoU = 0.5$ since the RoIs are based on anchors, which are uniformly distributed with a fixed set of boxes on the image. Thanks to the RPN regressor, the IoU distribution improves compared to the distribution of the sampled boxes with $baseIoU = 0.5$. On the other hand, this bias towards 0.5 is previously argued to make the regressor overfit for smaller IoUs [20]. **(3)** RPN is able to provide hard positive examples inherently; however, the number of prime samples (i.e. examples with larger IoUs) is quite low. This is critical since it is shown that prime sampling performs better than hard positive mining [33].

4.5.2 Foreground-Foreground Class Imbalance

We observe that, for each RoI source, addressing foreground-foreground imbalance ($FGB=Yes$) improves performance in terms of both AP and oLRP, especially for the right skew and uniform cases (Table 4.1). Moreover, addressing foreground-foreground class imbalance does not seem to affect the localisation error ($oLRP_{Loc}$) but improves the classification performance since AP_{50} , $oLRP_{FP}$ and $oLRP_{FN}$ get better (except for the left-skew case). Therefore, we conclude foreground-foreground class imbalance can also be alleviated by employing methods in the batch level.

4.5.3 Effect of Online Hard Positive Mining

Here we demonstrate another useful use-case of our pRoI generator by simulating OHEM [27] on positive examples. OHEM chooses the positive and negative exam-

ples with the highest loss values after applying NMS to the examples to preserve example diversity. A recent study [32] showed that the IoU and the hardness of an example are correlated. On the other hand, another study [33] proposed an opposite perspective to the OHEM based on prioritizing “prime samples”, i.e. samples with high IoUs. To be more clear, OHEM [27] implies preferring positive examples with IoUs just above 0.5, while prime sampling asserts that the higher the IoU, the better the example. To make an analysis on the positive examples, we simulate OHEM by (i) initially generating 128 BBs by pRoI generator, (ii) applying NMS using loss value of an example, (iii) finally selecting the ones with the larger loss values. We coin this as **online hard positive mining (OHPM)**.

In our experiments, we observe that the effect of the hard examples depends on the IoU distribution of the RoIs and high-quality samples are required during training: In Table 4.1, when OHPM is applied, uniform and right-skew distributions, which have more difficult examples due to their distribution (Fig. 4.5), have better performance compared to the left-skew and “Base IoU=0.5” cases. Moreover, while OHPM does not improve the performance of left-skew and “Base IoU=0.5” cases, it is crucial for the right-skew and uniform distributions (Table 4.1). Therefore, similar to prime sampling [33], we show that examples with higher IoUs are crucial during training, however, we also show that these examples should be supported by hard examples.

4.5.4 Relative Spatial Imbalance

We now analyse the relative spatial distribution of the RPN RoIs and how they fit within the theoretical IoU boundaries in Fig. 4.6. To be able to make such an analysis, we selected a reference box with $[x_1, y_1, x_2, y_2] = [0.3, 0.3, 0.6, 0.6]$. At the final epoch of the RPN training, we track positive RPN RoIs with associated ground truths. As discussed in Section 4.3.1, we scaled and shifted the ground truths to the reference box and applied the same transformations to their associated positive RPN RoIs. Among the positive RPN RoIs, top-left (TL) points of the 2, 500 RoIs are plotted with green dots in Fig. 4.6. Then, using `findTLFeasibleSpace()` function in Algorithm 1, we plot the theoretical limits for the top left points for RoIs with IoUs larger than 0.5, 0.6, 0.7, 0.8 and 0.9. Especially the last two observations may be critical for an object

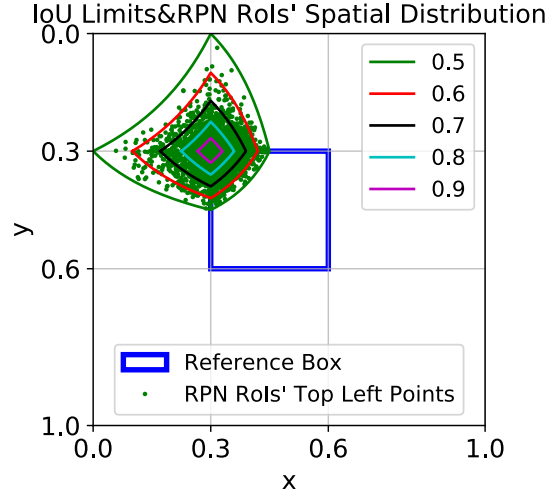


Figure 4.6: Relative spatial distribution of 2,500 RPN RoIs TL points and max. IoU limits from $IoU = 0.9$ to 0.5 (in-out direction)

detector since they may result in a positive bias towards specific RoIs and may make the generalization difficult over the entire spatial space. However, the effects of all these observations require experimental or theoretical validation that is not provided in this paper.

Fig. 4.6 leads to several key findings: **(1)** As expected, as the IoU decreases, the boundaries occupy a larger space around the TL point of the reference box. Hence, the sample space for 0.9 is very small, which makes it more difficult to have distinct RoIs with $IoU > 0.9$. **(2)** We observe that no TL point is outside of the 0.5 boundary, which is a sanity check for the boundaries since a RoI is labeled as positive if it has at least 0.5 IoU with a ground truth. **(3)** The TL points of the RPN RoIs are accumulated around the TL point of the reference box and they are not uniformly distributed within the 0.5 boundary. **(4)** The TL points of the RPN RoIs tend to be inside the reference box more than to be outside. Specifically, RPN RoIs between $x > 0.3, y > 0.3$ and $x < 0.3, y < 0.3$ are 28.2% and 21.0% of the all, respectively.

4.6 Practical Improvements

In this section, we propose OFB sampling and show the effect of employing pRoI generator for training the second-stage of Faster R-CNN.

Table 4.2: Average performance of 3 runs for Faster R-CNN with our OFB sampling on Pascal VOC. Lower is better for oLRP and its components, whereas higher is better for AP.

Sampling Method	oLRP ↓	oLRP _{Loc} ↓	oLRP _{FP} ↓	oLRP _{FN} ↓	AP ₅₀ ↑
Random	59.4	18.7	16.2	27.7	78.0
OFB	58.9	18.7	15.6	27.2	78.5

Table 4.3: Comparison of different sampling mechanisms on COCO using Faster R-CNN. Lower is better for oLRP and its components, whereas higher is better for AP. AP^C stands for COCO-style AP. R and H denote random and hard sampling respectively, and OFB is our sampling method for positive RoIs. The first block compares among different positive sampling schemes combined with random sampling, while the second block compares their combinations with hard example mining.

Sampling Method				
Positive	Negative	oLRP ↓	AP ^C ↑	AP ₅₀ ↑
R	R	72.4	34.1	55.2
H	R	75.3	31.0	51.7
OFB	R	72.1	34.7	55.8
R	H	71.9	35.3	54.6
H	H	74.6	31.1	50.0
OFB	H	70.9	35.6	55.3

4.6.1 Online Foreground Balanced Sampling

In the conventional training, the set of positive RoIs are limited and they are not generated as in pRoI generator. Motivated from the analysis using pRoI generator on the effect of foreground-foreground class imbalance (Section 4.5), we propose an online sampling method to be used in the conventional training pipeline. Denoting the total number of classes in a batch by C and the number of positive RoIs for class c by k_c , each RoI is assigned a probability $1/(Ck_c)$ and the subset of RoIs to train Faster R-CNN is sampled from this multinomial distribution. We call this sampling

scheme as Online Foreground Balanced (OFB) Sampling.

In order to see the effect, we train Faster R-CNN with and without OFB sampling and present results in Tables 4.2 and 4.3. For the Pascal VOC [48], we observe 0.5% improvement in AP_{50} and oLRP, with better performance in precision and recall components of oLRP and no impact on the regression branch. In our experiments with COCO (Table 4.3), we compared our results with hard example mining [24, 27]. Similar to the findings of Cao et al. [33] and our analysis in Section 4.5, while hard positive mining does not improve performance, our OFB sampling is beneficial for foreground examples. Moreover, the table shows that OFB sampler can be combined with sampling approaches for negative BBs. In any case, similar to our experiments for Pascal VOC, the best performance gain is in AP_{50} . This suggests that controlling RoIs to balance foreground classes has also a role during training of the object detectors and OFB, an efficient sampling algorithm, can be considered a basic solution for the problem.

4.6.2 Generating More Samples in Higher IoUs

Our approach can be integrated into an object detector without any hindrance on the gradient paths. In this section, we compare a detector trained with our pRoI Generator with a detector trained with the conventional method (i.e. using RPN RoIs) – Table 4.4. We use Uniform RoI source with foreground balance and OHPM since it performed the best in Table 4.1. For $IoU = \Theta$, we randomly sample negative samples from the output of the RPN in the range $[0.1, \Theta]$ and the positive samples are provided by the pRoI generator also using OHPM. To apply OHPM, we first generate $RoI\Num$ boxes, then select fg many from them. In IoUs 0.6 – 0.8, for which fewer RoIs are possible than 0.5, we initially train the models for 1 epoch by setting $fg = 32$ and $bg = 96$ and track “Mean RoI #” to see an upper bound for the models to generate RoIs and prevent class imbalance modelwise. In this run, Mean RoI # for IoUs 0.6, 0.7, 0.8 are 17.26, 7.60, 1.72 for RPN and 20.0, 11.41, 4.67 for pRoI-Uniform respectively. Then using $IoU = 0.5$ as an example, we multiply the resulting “Mean RoI #” by 1.5 and set fg approximately to it with $bg = 3 \times fg$ as in the conventional training. This approach makes training more stable and fair especially

Table 4.4: Performance Comparison with RPN on PASCAL VOC. $RoINum$ is the input of pRoI generator, fg/bg is the desired fg and bg RoI numbers during training, and Mean RoI # is the actual mean of number of positive RoIs. Note that fg/bg RoI numbers are set differently for pRoI and RPN so that the best performance is achieved for both of these RoI sources in order to provide a fair comparison especially in favor of RPN. We trained the models (except the one with the * mark) for 16 epochs with a learning rate decay at epochs 9 and 14 since our model provides more diverse data than RPN (see in Fig. 4.6 that the TL points of the RPN RoIs clusters around TL point of B) and there are fewer samples for training in higher IoUs (see Mean RoI # in Table 4.4)

RoI Source	IoU	$RoINum$	fg/bg	Mean RoI # $\uparrow$	oLRP $\downarrow$	oLRP _{Loc} $\downarrow$	oLRP _{FP} $\downarrow$	oLRP _{FN} $\downarrow$	AP _{IoU} $\uparrow$
RPN*	0.5	N/A	32/96	27.12	59.3	18.7	16.0	27.7	78.0
pRoI-Uniform	0.5	128	32/96	25.49	59.2	18.4	15.5	28.2	77.1
RPN	0.6	N/A	27/81	16.92	65.4	17.0	19.4	31.9	71.2
pRoI-Uniform	0.6	128	27/81	18.28	65.4	16.9	20.8	31.0	70.6
RPN	0.7	N/A	9/27	5.39	74.9	14.7	27.2	42.1	57.3
pRoI-Uniform	0.7	128	18/54	9.93	74.5	14.9	28.0	39.8	57.5
RPN	0.8	N/A	2/6	1.08	92.5	13.2	58.8	69.8	21.3
pRoI-Uniform	0.8	64	8/24	3.92	87.7	12.1	47.8	59.3	32.2
RPN	0.9	N/A	2/6	0.17	99.5	7.4	94.2	97.1	0.5
pRoI-Uniform	0.9	32	2/6	1.62	99.3	7.3	92.4	96.0	0.9

Table 4.5: Effect of $RoINum$ on PASCAL VOC. Speeds are reported on a single Geforce GTX 1080 Ti.

RoI Source	$RoINum$	oLRP ↓	AP ₅₀ ↑	Train Speed ↓	Mean RoI # ↑
pRoI-Uniform	32	60.3	77.5	0.41s	14.81
pRoI-Uniform	64	59.7	77.6	0.58s	21.32
pRoI-Uniform	128	59.9	77.8	0.97s	25.49

for the RPN (Table 4.4) by balancing foreground and background consistently.

Looking at Table 4.4 and comparing the methods in the IoUs that they are trained for, we observe the following: **(1)** For $IoU = 0.5, 0.6$ and $IoU = 0.7$ we get comparable results with the conventional training. **(2)** For $IoU = 0.8$, where RPN is not able to generate sufficient samples, the performance increases significantly in terms of both metrics since, at each iteration, generated positive boxes are provided consistently to the second stage. **(3)** Overall, the mean RoI # is approximately four times higher at $IoU = 0.8$; and, AP₈₀ and oLRP improve by 10.9% and 4.8% respectively. A similar trend is also achieved for $IoU = 0.9$.

In short, these results demonstrate that it is possible to train an object detector using BB generator with comparable results for lower IoUs and significantly better performance for higher IoUs. On par performance for low IoUs can be owing to the fact that there are sufficient amount of samples for these cases to see any imbalance effect.

Effect of $RoINum$: Apart from the input parameters to determine the nature of the RoI source, $RoINum$ is the only new hyperparameter in Algorithm 2. In Table 4.5, we observe that training improves (AP₅₀ increases) when $RoINum$ is increased because we have more positive samples at each iteration. However, more samples mean slower (yet still acceptable) training speed compared to conventional training having 0.23s training speed.

Preliminary Results on COCO: In order to back up our claims, we also conducted an experiment on COCO dataset using $IoU = 0.8$ with Faster R-CNN. Compared to the baseline achieving oLRP = 95.1 and AP₈₀ = 13.2, using pRoI generator the model has oLRP = 93.7 and AP₈₀ = 15.3. These results suggest that our model is

able to generate more diverse examples than the baseline in larger IoUs.

4.7 Conclusion

In this paper, we proposed a BB generator and a positive RoI generator. We showed that generated RoIs can be used both as an analysis tool (owing to its controllable nature) and a training method for the two-stage object detectors.

We showed that there is a bias in the RPN RoIs' IoU and spatial distribution with respect to the IoU boundaries that are physically possible and analysed the IoU distributions of RPN and other RoI sources.

Using our BB generator, we developed a pRoI generator that can generate RoIs overlapping with a GT box with a desired IoU or relative spatial distribution. Then, we trained Faster R-CNN's second-stage with the RoIs generated according to different distributions. We showed that, by producing more samples than RPN, we can achieve better or comparable performance to Faster R-CNN. Moreover, our results reconciliated two conflicting recent studies [27, 33] that both high-IoU and hard RoIs can have positive effect on the training if the IoU distribution is appropriate.

Our ideas can be used for analyzing the anchors of a one-stage detector (as well as those of a two-stage detector) in order to design a better anchor set. Furthermore, other applications, e.g. tracking, that require spatially distributed BBs with certain properties can also exploit our approach.



CHAPTER 5

LOCALISATION RECALL PRECISION (LRP) ERROR FOR EVALUATING VISUAL DETECTION TASKS

This chapter presents our novel performance metric, Localisation-Recall-Performance (LRP) Error, for evaluating visual detection tasks based on our works,

- Kemal Oksuz, Baris Can Cam, Emre Akbas* and Sinan Kalkan*, “Localization Recall Precision (LRP): A New Performance Metric for Object Detection”, European Conference on Computer Vision (ECCV), 2018.
- Kemal Oksuz, Baris Can Cam, Sinan Kalkan* and Emre Akbas*, “One Metric to Measure them All: Localisation Recall Precision (LRP) for Evaluating Visual Detection Tasks”, under review at IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI).

Here, we mainly exploit the text from our preprint (i.e. submission to TPAMI) [147] as the recent and extended version of our conference paper on LRP Error [107] by making minor changes mainly to ensure the notation to be consistent throughout the thesis.

5.1 Introduction

Many vision applications require identifying objects and object-related information from images. Such identification can be performed at different levels of detail, which are addressed by different detection tasks such as “object detection” for identifying

* Equal contribution for senior authorship.

labels of objects and boxes bounding them, “keypoint detection” for finding keypoints on objects, “instance segmentation” for identifying the classes of objects and localising them with masks, and “panoptic segmentation” for classifying both background classes and objects by providing detection ids and labels of pixels in an image. Accurately evaluating performances of these methods is crucial for developing better solutions.

Today “average precision” (AP), the area under the Precision-Recall (PR) curve, is the de facto standard for evaluating performance on many visual detection tasks and competitions [13, 48, 49, 98, 99, 148, 45]. AP not only enjoys vast acceptance but also appears to be unchallenged. There has been only a few attempts on developing an alternative to AP [107, 149, 150]. Despite its popularity, AP has many limitations as we discuss below.

5.1.1 Important features for a performance measure

To facilitate our analysis of AP and other performance measures, we define three important features:

Completeness. Arguably, three most important performance aspects that an evaluation measure should take into account in a visual detection task are false positive (FP) rate, false negative (FN) rate and localisation error. We call a performance measure “complete” if it precisely takes into account all three quantities.

Interpretability. Interpretability of a performance measure is related to its ability to provide insights on the strengths and weaknesses of the detector being evaluated. To provide such insight, the evaluation measure should ideally comprise interpretable components.

Practicality. Any issue that arises during practical use of a performance measure diminishes its practicality. This could be, for example, any discrepancy between the well-defined theoretical description of the evaluation measure and its actual application in practice, or any shortcoming that limits the applicability of the measure to certain scenarios.

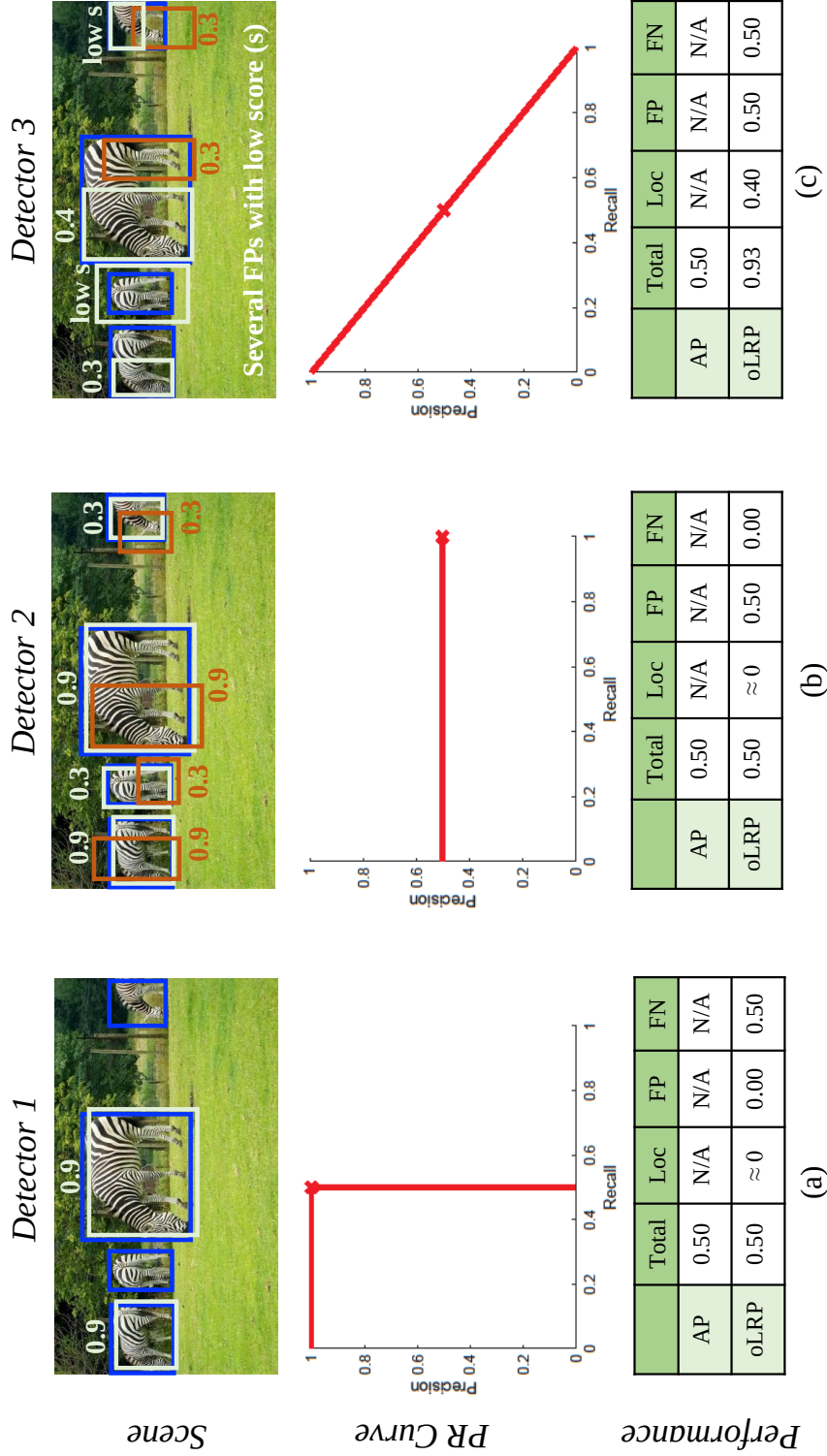


Figure 5.1: Three different object detection results (for an image from COCO [13]) with very different PR curves but the same AP. **First Row:** Blue, white and orange colors denote ground-truth, TPs and FPs respectively. Numbers are confidence scores, s , of the detections. **Second row:** PR curves for the corresponding detections. Red crosses indicate optimal points designated by oLRP. **Third row:** AP and oLRP results of the detection results. While the localisation quality of the TPs in (c) is worse than (a) and (b), AP does not penalize (c) more and cannot identify the difference between (a) and (b) despite they have very different problems.

5.1.2 Limitations of AP

Completeness. Localisation quality is only loosely taken into account in AP. Detections that meet a certain localisation criterion (e.g., intersection over union (IoU) over 0.50 in object detection) are treated equally regardless of their actual localisation quality. Further increase in localisation quality, for example, increasing the IoU of a true positive (TP) detection, does not change AP (Fig. 5.1).

Interpretability. The AP score itself does not provide any insight in terms of the important performance aspects, namely, FP rate, FN rate and localisation error. One needs to inspect the PR curve and make additional measurements (e.g. average-recall (AR) or some kind of localisation quality) in order to comment on the weaknesses or strengths of a detector in terms of these aspects. Therefore, being an approximation of the area under the PR curve, AP may fail to distinguish between the underlying issues of different detectors, as illustrated in Fig. 5.1.

Practicality. We identify three major issues related to the practical use of AP:

- (i) Using AP to evaluate hard-prediction tasks, i.e. tasks that involve outputs without confidence scores, such as panoptic segmentation [150], is problematic because hard predictions yield only a single point on the PR curve. Assumptions are needed to compute the area under the PR curve consisting of just one point.
- (ii) AP cannot be used for model selection. For example, when a detector is to be deployed for a certain problem, an optimal detection threshold is needed. AP does not offer any help in finding such optimal thresholds.
- (iii) Lastly, one needs to interpolate the PR curve before computing AP, which, as we will show, is a problem with classes with few examples.

We provide a detailed discussion on each limitation in Section 5.3 and provide an empirical analysis in Section 5.7.2.

5.1.3 Motivation for our Localisation Recall Precision (LRP) Error

LRP Error is a novel performance metric for visual detection tasks. We proposed LRP Error in our first work [107] for the object detection task, and then extended it for all visual detection task [147], where we showed that it alleviates all the aforementioned limitations of AP (Section 5.1.2): (i) LRP Error precisely combines the important performance aspects, therefore it is complete (compare AP and LRP in Fig. 5.1 - see Section 5.1.1 for completeness). (ii) LRP Error is easily interpretable by definition, and through its components, it provides insights regarding each performance aspect (compare AP and LRP in Fig. 5.1). (iii) Regarding the practicality issues of AP; with the Optimal LRP (oLRP) extension, LRP Error can evaluate both soft predictions (i.e. outputs with class labels and confidence scores, such as in object detection) and hard predictions (i.e. outputs with class labels only, such as in panoptic segmentation), can provide a class-specific optimal threshold and does not employ any interpolation for its computation. In addition, LRP is a metric, for which, however, we do not demonstrate any theoretical or practical benefits.

5.1.4 Other alternatives to AP

While AP is still de facto performance measure for many visual detection tasks, recently proposed visual detection tasks have preferred not employing AP, but instead introduced novel performance measures:

Panoptic Quality (PQ): Panoptic segmentation task [150] requires the background classes to be labeled and localised by masks in addition to the objects. Since this task is a combination of the instance segmentation and semantic segmentation tasks, AP can be used to evaluate performance. However, arguing the inconsistency between machines and humans in terms of perceiving the objects due to the confidence scores in the outputs, Kirillov et al. [150] preferred to discard these scores for evaluation. Considering the limitations of AP to evaluate hard predictions (Section 5.1.2), PQ was proposed as a new performance measure to evaluate the results of the panoptic segmentation task. Similar to LRP Error [107], PQ combines all important performance aspects of visual detection, however, its extension to other visual detection

tasks has not been explored. We provide a detailed analysis on PQ in Section 5.4 and discuss empirical results in Section 5.7.3. Note that PQ was proposed later than LRP.

Probability-based Detection Quality (PDQ): Unlike conventional object detection, probabilistic object detection (POD) [149] takes into account the spatial and semantic uncertainties of the objects, and accordingly for each detection, requires (i) a probability distribution over the class labels (i.e. instead of a single confidence score as in soft predictions) and (ii) a probabilistic bounding box represented by Gaussian distributions. Similar to Kirillov et al. [150], Hall et al. [149] also did not prefer an AP-based performance measure for POD, instead proposed a new performance measure called PDQ to evaluate probabilistic outputs. In this chapter, we limit our scope to deterministic approaches to visual detection tasks. Therefore, we do not delve into a detailed discussion on PDQ as we do for AP and PQ; instead, we provide a guidance on how LRP Error can be extended for different visual detection tasks in Section 5.5.4.

5.1.5 Contributions of the Chapter

Our contributions are as follows:

1. We thoroughly analyse Average Precision and Panoptic Quality.
2. We present LRP Error and describe its use for *all* visual detection tasks. LRP Error can evaluate all visual detection tasks in both output types (i.e. hard and soft predictions) by alleviating the drawbacks of AP and PQ. In particular, we empirically present the usage of LRP on four important visual detection tasks, namely, object detection, keypoint detection, instance segmentation, panoptic segmentation, and discuss its potential extensions to other tasks.
3. While LRP Error can directly be used for hard predictions, to evaluate soft predictions we propose Optimal LRP (oLRP) error as the minimum achievable LRP Error over the confidence scores.
4. We show that LRP Error is an upper bound for the error versions of precision, recall and PQ (Section 5.6). Therefore, minimizing LRP is guaranteed to mini-

mize the other measures.

5. We show that the performances of visual object detectors are sensitive to thresholding, and based on oLRP, we propose “LRP-Optimal Threshold” to reduce the number of detections in an optimal manner.

To demonstrate that LRP provides more insight than AP and PQ, we compare LRP with its counterparts on 35 state-of-the-art visual detectors. Using these detectors, we provide examples, observations and various analysis with LRP and oLRP at the detector- and class-level to present its evaluation capabilities. Our experiments also show that (i) LRP can unify the evaluation of all visual detection tasks in any desired output type (i.e. soft predictions or hard predictions), (ii) object detectors need to be thresholded in a class-specific manner, (iii) LRP-Optimal threshold is able to threshold object detectors a class-specific manner by considering all performance aspects, and (iv) the additional overhead of LRP computation to the COCO toolkit is negligible: It takes an additional 0.2 milliseconds per image on average (on COCO 2017 val) to output LRP, its components and LRP-Optimal thresholds on an eight-core standard CPU.

5.1.6 Outline of the Chapter

The chapter is organized as follows. Section 5.2 presents the related work. Sections 5.3 and 5.4 present a thorough analysis of Average Precision and Panoptic Quality respectively. Section 5.5 defines the LRP Error, oLRP Error and potential extensions of LRP. Section 5.6 compares LRP Error with AP and PQ. Section 5.7 presents several experiments to quantitatively analyse LRP. Finally, Section 5.8 concludes the chapter.

5.2 Related Work

Evaluation in visual detection tasks. As discussed in Section 5.1, except for the panoptic segmentation task which uses PQ [150], the performances of visual object detection methods are conventionally evaluated using AP. Sections 5.3 and 5.4 discuss and present an analysis of these performance measures.

Another measure, PDQ [149] has recently been proposed for evaluating the probabilistic object detection task, where the label of a detection is represented by a discrete probability distribution over classes and the bounding boxes are encoded by Gaussian distributions. To compute PDQ, first, pairwise PDQ (pPDQ) score is computed over all detection-ground truth pairs and the optimal matchings are identified following the Hungarian Algorithm [151]. Then, determining TPs, FPs and FNs, and using the pPDQs of optimal matchings, PDQ score of a detection set can be computed by normalizing the sum of these pPDQs by the total number of TPs, FPs and FNs. To evaluate each pair, pPDQ combines localisation and classification performances by its spatial quality and label quality components. The spatial quality evaluates a pair in a pixel-based manner (i.e. not box-based) by exploiting the segmentation mask of the ground truth. And, the label quality of a pair is the probability of the ground truth label in the label distribution of the detection. Therefore, computing PDQ requires (i) segmentation masks which are normally not provided for the conventional object detection task, and (ii) the outputs to be in the described probabilistic form. In this chapter, we demonstrate that LRP can be used for all common visual detection tasks having the conventional deterministic representation, and provide a guideline on how it can be employed by other tasks.

Analysis tools for visual detection tasks. Over the years, diagnostic tools have been proposed for providing detailed insights on the performances of detectors. For example, Hoiem et al. [152] selected the top-k FPs based on confidence scores and analysed them in terms of common error types (i.e. localisation error, confusion with similar objects, confusion with other objects and confusion with background). However, the tool of Hoiem et al. [152] requires additional analysis for FNs. Another toolkit, the COCO toolkit [13], is based on this analysis tool, but instead plots the considered error types on PR curves progressively to present how much AP difference is accounted by each error type. Recently, Bolya et al. [153] showed that the COCO toolkit can yield inconsistent outputs when the order of progressive contribution of the error types to the AP is interchanged. Moreover, this analysis by the COCO toolkit yields superimposed numerous PR curves which are time-consuming to examine and hard to digest. Based on these observations, Bolya et al. [153] proposed TIDE, a toolkit addressing the limitations of the previous analysis tools. TIDE introduces six

different error types, each of which is summarized by a single score in the analysis result. Although such tools are useful for providing detailed insights on the types of errors detectors are making, they are not performance measures, and as a result they do not yield a single performance value as the detection performance.

Point multi-target tracking performance metrics. The evaluation of the detection tasks is very similar to that of multi-target tracking in that there are multiple instances of objects/targets to detect, and the localisation, FP and FN errors are common criteria for success. Currently, component-based performance metrics are the accepted way of evaluating point multi-target tracking methods. One of the first metric to combine the localisation and cardinality (including both FP and FN) errors is the Optimal Subpattern Assignment (OSPA) [154]. Among the successors of OSPA, our LRP [107] was inspired by the Deficiency Aware Subpattern Assignment metric [155], which combines the three important performance aspects.

Summary. We observe that, with similar error definitions, point multi-target tracking literature utilizes component-based performance metrics commonly, which has not been explored thoroughly in the visual detection literature. While a recent attempt, panoptic quality, is an example of that kind, it is limited to panoptic segmentation (Section 5.4). The analysis tools also aim to provide insights on the detector, however, a single performance value for the detection performance is not provided by these methods. In this chapter, we propose a single metric that ensures important features (i.e. completeness, interpretability and practicality) while evaluating the performance of methods for visual detection tasks. We also show that our metric, considering all performance aspects, is able to pinpoint a class-wise optimal threshold for the visual detectors, from which several applications can benefit in practice.

5.3 Average Precision

This section provides a definition of AP and its detailed analysis.

5.3.1 Definition of AP

Computing AP for a class involves a set of detection results with confidence scores and a set of ground-truth items (e.g. bounding boxes in the case of object detection). First, detections are matched to ground-truth items (GT) based on a predefined spatial overlap criterion such as IoU¹ being larger than $\tau = 0.50$. Each GT can only match one detection and if there are multiple detections that satisfy the overlap criterion, the one with the highest confidence score is matched. A detection that is matched to a GT is counted as a TP. Unmatched detections are FPs and unmatched GTs are FNs. Given a specific confidence threshold s , detections with a lower confidence score than s are discarded, and TP, FP, FN values are calculated with the remaining detections as described. By systematically changing s , we obtain a precision-recall (PR) curve. This process usually results in a non-monotonic curve, that is, the precision may go up and down as recall is increased. Conventionally [13, 14, 48, 100, 45], in order to decrease these wiggles, the PR curve is interpolated as follows: denoting the precision at a recall r_i before and after interpolation by $p(r_i)$ and $\hat{p}(r_i)$ respectively, $\hat{p}(r_i) = \max_{r_j > r_i} p(r_j)$ [48]. Then, AP (for a class) is the area under this interpolated curve, or an approximation of this area by averaging it over evenly-spaced recall values [48, 13]. The detector’s performance over all classes is obtained simply by averaging over AP values per class. In order to include the localisation quality, which is somewhat ignored by AP, the COCO-style AP, denoted by AP^C , computes 10 AP_τ where the TP validation threshold, τ , is increased between 0.50 and 0.95 with a step size of 0.05, and these 10 AP_τ values are averaged. From the definition of AP, it follows that it is a ranking-based performance measure, and favors methods that have high precision over the entire recall domain.

5.3.2 An Analysis of AP

In the following, we provide an analysis of AP by discussing its limitations introduced in Section 5.1.2 in detail:

Completeness. *AP does not explicitly evaluate localisation performance (Fig. 5.1)*

¹ Note that while for the object detection task IoU is computed between bounding boxes, it is computed between the masks for segmentation tasks.

and therefore, violates completeness. To circumvent this issue, researchers typically use the following methods, neither of which ensures completeness:

- *Quantitatively using COCO-style AP (AP^C) or AP_τ with large τ* : AP variants do not include the precise localisation quality of a detection except for thresholding, hence the contribution of the localisation performance to these AP variants is always loose. Owing to this loose contribution, the localisation quality can not be quantified by AP. As a result, the methods specifically proposed to improve the localisation quality [20, 156, 90, 93, 157] have been struggling to present their contributions quantitatively. While some of them [20, 93] present only AP^C , AP_{75} and AP_{50} , a subset of these methods [156, 36, 90, 157] additionally resort to APs with larger τ values such as AP_{80} or AP_{90} . In any case, it is not clear or consistent to interpret any AP variant in terms of localisation.
- *Presenting qualitative examples [37, 19, 23, 24, 158, 159]*: In this case, note that it is very likely for the selected examples to be very limited and biased.

We empirically analyse this limitation in Section 5.7.2.1.

Interpretability. *The resulting AP value does not provide any insight on the strengths or weaknesses of the detector.* As illustrated in Fig. 5.1, different detectors may yield different PR curves, highlighting different types of performance issues. However, being an approximation of the area under the PR curve, AP fails to distinguish between the underlying issues of different detectors. This is mainly because both precision and recall performances of a detector are vaguely combined into a single performance value as an AP value. Besides, interpreting the COCO-style AP, AP^C , is more difficult since the localisation quality is also integrated in an indirect and loose manner, resulting in an ambiguous contribution of important performance aspects, where it is not clear how much each aspect affects the resulting single performance value. Similar to our previous work [107], Bolya et al. [153] also criticized AP since it does not isolate error types. To alleviate this, the COCO toolkit [13] can output PR curves with an error analysis, which requires manual inspection of several superimposed PR curves in order to understand the strengths and weaknesses. This is, however, time-consuming and impractical with large number of classes such as the LVIS dataset [45] with around 1000 classes (also see the discussion on analysis tools in Section 5.2).

We empirically analyse this limitation in Section 5.7.2.2.

Practicality. One can also face some practical challenges while employing AP for some use-cases:

- *Evaluation of hard predictions with AP, though possible, is problematic.* Note that a hard prediction (i.e. an output without confidence score) corresponds to a single point on the PR space, hence determines a step PR curve resulting in $AP = \text{Precision} \times \text{Recall}$. However, AP intends to prioritize and rank the detections with respect to their confidence scores, which are not included in hard predictions. As a result, in a recent study, Kirillov et al. [150] proposed a new performance measure called Panoptic Quality for the panoptic segmentation task (e.g. instead of using $\text{Precision} \times \text{Recall}$ as AP), which can evaluate hard predictions. Therefore, the usage of AP on hard predictions does not fit into its ranking-based definition.
- *AP does not offer an optimal threshold for a detector.* Being defined as the area under the PR curve, any thresholding on detections decreases this area. Hence, performance with respect to AP increases, when the confidence score threshold approaches to 0 (i.e. the case of “no-thresholding”). As a result, it is not clear how the large number of object hypotheses can be reduced properly with AP when a visual detector is to be deployed in a practical application.
- *Using AP for classes with few examples is problematic owing to the interpolation of the PR curve (Section 5.3.1 for how PR curve is interpolated in AP computation).* Having few examples causes the recall axis to have a sparse set of values, and in this case interpolating the line segments spanning larger recall intervals will change the AUC more, which can especially have an effect for long-tail visual detection challenges such as LVIS [45] with a median of only 9 instances per class in the validation set.

We empirically analyse these limitations in Section 5.7.2.3.

5.4 Panoptic Quality

Following a similar methodology with how we analysed AP, this section provides an analysis for PQ.

5.4.1 Definition of PQ

The PQ measure is proposed to evaluate the performance of panoptic segmentation methods [150]. Given hard predictions (i.e. outputs without confidence scores), first, the detections are labelled as TP, FP and FN using an IoU-based criterion, and then the numbers of TPs (N_{TP}), FPs (N_{FP}), FNs (N_{FN}) and the localisation quality of TP detection masks in terms of IoU (i.e. $\text{IoU}(g_i, d_{g_i})$ is the IoU between the mask of the ground truth g_i and the mask of the associated detection, d_{g_i} , with g_i) are computed. Based on these quantities, PQ between a ground truth set $\mathcal{G}$ and a detection set $\mathcal{D}$ is defined as:

$$\text{PQ}(\mathcal{G}, \mathcal{D}) = \frac{1}{N_{\text{TP}} + \frac{1}{2}N_{\text{FP}} + \frac{1}{2}N_{\text{FN}}} \left(\sum_{i=1}^{N_{\text{TP}}} \text{IoU}(g_i, d_{g_i}) \right). \quad (5.1)$$

PQ is a “higher is better” measure with a range between 0 and 1, and includes the localisation quality of TPs, the number of FPs and the number of FNs. To provide more insight on the segmentation performance, PQ is split into two components: (i) Segmentation Quality (SQ), defined as the average IoU of the TPs, is a measure of the localisation performance; (ii) Recognition Quality (RQ) is a measure of classification performance defined as the F-measure. Using SQ and RQ, PQ can equally be expressed as: $\text{PQ}(\mathcal{G}, \mathcal{D}) = \text{SQ}(\mathcal{G}, \mathcal{D})\text{RQ}(\mathcal{G}, \mathcal{D})$.

5.4.2 An Analysis of PQ

In the following, we analyse PQ in terms of the same criteria that we used for AP:

Completeness. In contrast to AP, PQ precisely takes into account all performance aspects (i.e. FP rate, FN rate and localisation error - see “completeness” in Section 5.1.1) that are critical for visual detectors.

Table 5.1: A comparison of LRP and PQ for the detectors (i.e. Detector 1 and Detector 2) in scenarios (a) and (b) in Fig. 5.1 (Since PQ and LRP do not need confidence scores, scores are simply ignored for computing PQ and LRP in these scenarios.) PQ cannot identify the difference between these two scenarios, and yields exactly the same results for both of its components (i.e. SQ and RQ). With a component for each performance aspect, LRP can discriminate between these results using FP and FN components. While for PQ and components higher is better; for LRP, lower is better.

Scenario	PQ	SQ	RQ
(a)	0.67	≈ 1	0.67
(b)	0.67	≈ 1	0.67

(a)

Scenario	LRP	Loc	FP	FN
(a)	0.50	≈ 0	0.50	0.00
(b)	0.50	≈ 0	0.00	0.50

(b)

Interpretability. Another advantage of PQ compared to AP is that PQ is relatively more interpretable than AP owing to its SQ and RQ components. On the other hand, while Kirillov et al. [150] proposed using these components (i.e. SQ and RQ) to provide insight on the detection performance, RQ, the F-measure, is limited in terms of discriminating recall and precision performances (Table 5.1(a)). This is because each performance aspect does not have a separate component in PQ (i.e. the error types are not isolated [153]), but instead, both precision error and recall errors are combined into a single component, RQ. Therefore, overall, PQ is superior than AP in terms of interpretability, but having a component for each performance aspect is better to provide more useful insights.

Practicality. Here we discuss the following issues, which mostly arise since PQ is designed only for panoptic segmentation, and omit a discussion on its generalizability over all detection tasks:

- Kirillov et al. [150] did not discuss how PQ can evaluate and threshold soft predictions (i.e. the outputs with confidence scores). Kirillov et al. preferred hard predictions for panoptic segmentation to eliminate the inconsistency between machines and humans in terms of perceiving the objects. Accordingly, proposed for being limited to panoptic segmentation, PQ is designed to evaluate hard predictions, and its possible extensions on soft predictions (and also

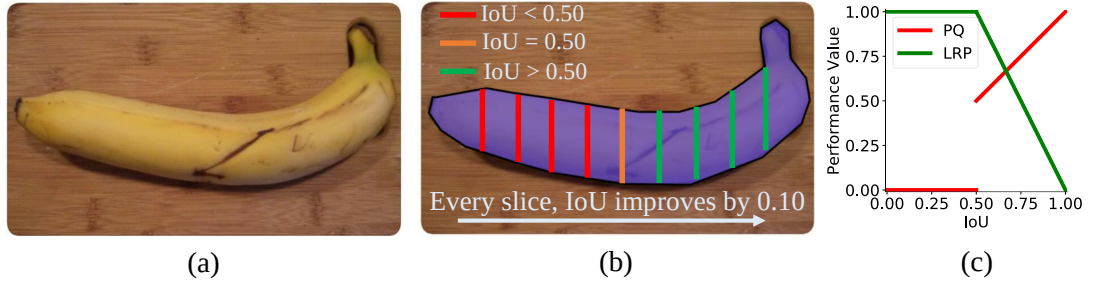


Figure 5.2: An illustration that shows how a transition from a FP to TP is handled differently by PQ and LRP. (a) An example image from COCO [13]. (b) Segmentation masks for the ground truth with different IoU. The ground truth is split into 10 approximately equal slices. Orange line is the threshold where the detection is still a FP, hence a single pixel added makes the detection a TP. (c) How PQ and LRP changes for different IoU. While LRP is zeroth-order continuous, PQ is a discontinuous function and allows large jumps.

other visual detection tasks) are not discussed and analysed by its authors.

- *PQ overpromotes classification performance compared to localisation performance inconsistently.* We observe the following for PQ: (i) Fig. 5.2 illustrates how small shifts, induced by a TP, can cause large changes in PQ. Due to this promotion of a TP via a jump in the performance value, the effect of the localisation quality is decreased since the localisation quality can contribute between $PQ \in [0.50, 1.00]$ (Fig. 5.2), (ii) While one can prefer classification error to have a larger effect on the overall performance, the formulation of PQ is inconsistent in terms of how localisation and classification performances are combined. In order to provide a comparative analysis with our performance metric, we discuss this inconsistency in Section 5.6. (iii) This inconsistent combination also makes PQ violate the triangle inequality property of metricity (see Appendix C for a proof).

5.5 Localisation-Recall-Precision (LRP) Error

In this section, we describe and analyse the LRP Error in Sections 5.5.1 and 5.5.2 respectively. Then, we present Optimal LRP (oLRP) as the extension of LRP for eval-

uating and thresholding soft-prediction-based visual object detectors (Section 5.5.3). We also discuss and present a guideline for other potential extensions of LRP Error (Section 5.5.4).

5.5.1 LRP: The Performance Metric

Definition: LRP is an error metric that considers both localisation and classification. To compute $\text{LRP}(\mathcal{G}, \mathcal{D})$ given a set of detections ($\mathcal{D}$ - each $d_i \in \mathcal{D}$ is a tuple of class-label and location information), and a set of ground truth items ($\mathcal{G}$), first, the detections are assigned to ground truth items based on the matching criterion (e.g. IoU) defined for the corresponding visual detection task. Once the assignments are made, the following values are computed: (i) N_{TP} , the number of true positives; (ii) N_{FP} , the number of false positives; (iii) N_{FN} , the number of false negatives and (iv) the localisation qualities of TP detections, i.e. $\text{lq}(g_i, d_{g_i})$ for all d_{g_i} where d_{g_i} is a TP matching with ground truth g_i . Using these quantities, the LRP error is defined as:

$$\text{LRP}(\mathcal{G}, \mathcal{D}) := \frac{1}{Z} \left(\sum_{i=1}^{N_{\text{TP}}} \frac{1 - \text{lq}(g_i, d_{g_i})}{1 - \tau} + N_{\text{FP}} + N_{\text{FN}} \right), \quad (5.2)$$

where $Z = N_{\text{TP}} + N_{\text{FP}} + N_{\text{FN}}$ is the normalisation constant and τ is the TP validation threshold ($\tau = 0.50$ unless otherwise stated). Eq. 5.2 can be interpreted as the “average matching error”, where the term in parentheses is the “total matching error”, and Z represents the “maximum possible value of the total matching error”. A TP contributes to the total matching error by its localisation error normalized by $1 - \tau$ to ensure that the value is in interval $[0, 1]$ and LRP is a zeroth-order continuous function(Fig. 5.2(c)). And, each FP or FN contributes to the total matching error by 1. Finally, normalisation by Z ensures $\text{LRP}(\mathcal{G}, \mathcal{D}) \in [0, 1]$. We prove that LRP is a metric if $1 - \text{lq}(x_i, y_{x_i})$ is a metric (Appendix D).

Components: In order to provide additional information on the characteristics of the detector, we show that LRP can be equivalently defined in a weighted form as:

$$\begin{aligned} \text{LRP}(\mathcal{G}, \mathcal{D}) := & \frac{1}{Z} (w_{\text{Loc}} \text{LRP}_{\text{Loc}}(\mathcal{G}, \mathcal{D}) + w_{\text{FP}} \text{LRP}_{\text{FP}}(\mathcal{G}, \mathcal{D}) \\ & + w_{\text{FN}} \text{LRP}_{\text{FN}}(\mathcal{G}, \mathcal{D})), \end{aligned} \quad (5.3)$$

with the weights $w_{\text{Loc}} = \frac{N_{\text{TP}}}{1 - \tau}$, $w_{\text{FP}} = |\mathcal{D}|$, and $w_{\text{FN}} = |\mathcal{G}|$ intuitively controlling

the contributions of the terms as the upper bound of the contribution of a component (or performance aspect) to the “total matching error”. These weights ensure that each component corresponding to a performance aspect (Section 5.1.1) is easy to interpret, intuitively balances the components to yield Eq. 5.2 and prevents the total error from being undefined whenever the denominator of a single component is 0. The first component, LRP_{Loc} , represents the localisation error of TPs as follows:

$$\text{LRP}_{\text{Loc}}(\mathcal{G}, \mathcal{D}) := \frac{1}{N_{\text{TP}}} \sum_{i=1}^{N_{\text{TP}}} (1 - \text{lq}(g_i, d_{g_i})). \quad (5.4)$$

The second component, LRP_{FP} , in Eq. 5.3 measures the FP rate:

$$\text{LRP}_{\text{FP}}(\mathcal{G}, \mathcal{D}) := 1 - \text{Precision} = 1 - \frac{N_{\text{TP}}}{|\mathcal{D}|} = \frac{N_{\text{FP}}}{|\mathcal{D}|}, \quad (5.5)$$

and the FN rate is measured by LRP_{FN} :

$$\text{LRP}_{\text{FN}}(\mathcal{G}, \mathcal{D}) := 1 - \text{Recall} = 1 - \frac{N_{\text{TP}}}{|\mathcal{G}|} = \frac{N_{\text{FN}}}{|\mathcal{G}|}. \quad (5.6)$$

When necessary, the individual importance of localisation, FP, FN errors can be changed for different applications (Section 5.6 and Appendix E).

5.5.2 An Analysis of LRP

As we did for AP (Section 5.3.2) and PQ (Section 5.4.2), in the following we analyse LRP in terms of important features for a performance measure.

Completeness: Both definitions of LRP Error above (which are equivalent to each other) clearly take into account all performance aspects precisely, and ensure completeness (Section 5.1.1).

Interpretability: The ranges of total error and the components are $[0, 1]$, and a lower value implies better performance. LRP Error describes the “average matching error” (see Definition), and each component summarizes the error for a single performance aspect, thereby providing insights on the strengths and weaknesses of a detector. Therefore, LRP Error ensures interpretability (Section 5.1.1). In the extreme cases; $\text{LRP} = 0$ means that each ground truth item is detected with perfect localisation, and if $\text{LRP} = 1$, then no detection matches any ground truth (i.e., $|\mathcal{D}| = N_{\text{FP}}$).

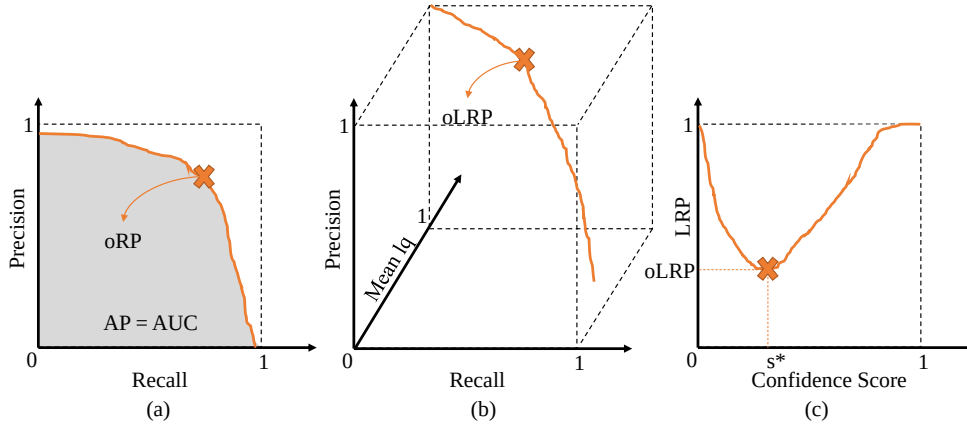


Figure 5.3: A visual comparison of AP and LRP. (a) A PR curve. The cross marks a hypothetical optimal-recall-precision point (oRP) (e.g. the point where F1-measure is maximized). (b) A localisation, recall and precision curve, where “Mean lq” is the average localisation quality of TPs. Unlike any performance measure obtained via a PR curve (e.g. AP), LRP Error intuitively combines these performance aspects (Eq. 5.2), and instead of area under the curve, uses the minimum of LRP values, defined as oLRP, as the performance metric. (c) An s-LRP curve to present the performance distribution of a detector for a class over confidence scores. Its minimum is oLRP.

Practicality: Since Eq. 5.2 requires a thresholded detection set (i.e. does not require confidence scores), LRP can directly be employed to evaluate hard predictions, and can be computed exactly without requiring any interpolations or approximations. In the next section, we discuss how LRP can be extended to evaluate soft predictions using Optimal LRP (oLRP) and show that it can also be computed exactly. Also, in order to prevent the over-represented classes in the dataset to dominate the performance, similar to AP and PQ, LRP is computed class-wise and then these class-wise LRP errors are averaged to assign the LRP Error of a detector. One practical issue of LRP is that localisation and FP components are undefined when there is no detection, and the FN component is undefined when there is no ground truth. However, even if some components (not all) are undefined, the LRP Error is still defined (Eq. 5.2). Namely, LRP is undefined only when the ground truth and detection sets are both empty (i.e., $N_{TP} + N_{FP} + N_{FN} = 0$), i.e., there is nothing to evaluate. When a component is undefined, we ignore the value while averaging it over classes.

5.5.3 Optimal LRP (oLRP): Evaluating and Thresholding Soft Predictions

Definition: Soft predictions (i.e. outputs with confidence scores) can be evaluated by, first, filtering the detections from a confidence score threshold and then, calculating LRP. We define Optimal LRP (oLRP) as the minimum achievable LRP error over the detection thresholds or equivalently, the confidence scores²:

$$\text{oLRP} := \min_{s \in \mathcal{S}} \text{LRP}(\mathcal{G}, \mathcal{D}_s), \quad (5.7)$$

where $\mathcal{D}_s$ is the set of detections thresholded at confidence score s (i.e. those detections with larger confidence scores than s are kept, and others are discarded). Eq. 5.7 implies searching over a set of confidence scores, $\mathcal{S}$, to find the best balance for competing precision, recall and localisation errors.

Components: The components of LRP for oLRP are coined as localisation@oLRP (oLRP_{Loc}), FP@oLRP (oLRP_{FP}), and FN@oLRP (oLRP_{FN}). oLRP_{Loc} describes the average localisation error of TPs, and oLRP_{FP} and oLRP_{FN} together indicate the point on the PR curve where optimal LRP is achieved. More specifically, one can infer the shape of the PR curve using the $(1 - \text{oLRP}_{\text{FP}}, 1 - \text{oLRP}_{\text{FN}})$ pair defining the optimal point on the PR curve.

Computation: Note that, theoretically, computing oLRP requires infinitely many thresholding operations since $\mathcal{S} = [0, 1]$. However, given that $\mathcal{S}$ is discretised by the scores of the detections, in order to compute oLRP exactly, it is sufficient to threshold the detection set only at the confidence scores of the detections. More formally, for two successive detections d_i and d_j (in terms of confidence scores) with confidence scores s_i and s_j where $s_i > s_j$, $\text{LRP}(\mathcal{G}, \mathcal{D}_s) = \text{LRP}(\mathcal{G}, \mathcal{D}_{s_j})$ if $s_j \leq s < s_i$. Then, oLRP for a class can be computed **exactly** by minimizing the finite LRP values on the detections, and one can average oLRP and its components over classes to obtain the performance of the detector.

oLRP as a Thresholder: Conventionally, visual object detectors yield numerous detections [18, 19, 160], most of which have smaller confidence scores. In order to deploy an object detector for a certain problem, the detections with “smaller” confi-

² Another way to evaluate soft predictions is the Average LRP (aLRP), the average of the LRP Errors over the confidence scores. While in a recent study [156], we showed that aLRP can be used as a loss function, we discuss in Appendix F why we preferred oLRP over aLRP as a performance measure.

dence scores need to be discarded to provide a clear output from the visual detector (i.e. model selection). While it is common to use a single class-independent threshold for the detector (e.g., Association-LSTM [159] uses SSD [24] detections for all classes with confidence score above 0.80), we show in Section 5.7.4 that (i) the performances of the detectors are sensitive to thresholding, and (ii) the thresholding needs to be handled in a class-specific manner. Note that balancing the competing performance aspects in an optimal manner, oLRP satisfies these requirements. In particular, we define the confidence score threshold corresponding to the oLRP Error as the “LRP-Optimal Threshold” (s^* -see Fig. 5.3). Different from the common approach, (i) s^* is a class-specific optimal threshold, and (ii) s^* considers all performance aspects of visual detection tasks (Fig. 5.3). See Appendices D and H for a further discussion on thresholding object detectors.

5.5.4 Potential Extensions of LRP

This section discusses potential extensions of LRP Error in three levels:

Extension to Other Localisation Quality Functions: Any localisation function that satisfies the following two constraints can be used within LRP: (i) $\text{lq}(\cdot, \cdot)$ should be a higher-better function, and (ii) $\text{lq}(\cdot, \cdot) \in [0, 1]$. In addition, choosing a $\text{lq}(\cdot, \cdot)$ such that $1 - \text{lq}(\cdot, \cdot)$ is a metric, guarantees the metricity of LRP Error. In case constraint (ii) is violated by a prospective $\text{lq}(\cdot, \cdot)$, then one can normalize the range of the function (and also TP validation threshold, τ) to satisfy this constraint. For example, as a recently proposed IoU variant to measure the spatial similarity between two bounding boxes, Generalized IoU (GIoU) [93] has a range of $[-1, 1]$. In this case, choosing $\text{lq}(\cdot, \cdot) = \text{GIoU}(\cdot, \cdot)/2 + 0.50$ will allow the use of GIoU within LRP.

Extension to New Detection Tasks: While adopting for new detection tasks, one should only consider the localisation quality function (see above). Following this, LRP can easily be adapted to new or existing detection tasks such as 3D object detection and rotated object detection.

Extension to Other Fields: LRP can be extended for any problem with the following two properties in terms of evaluation: (i) the similarity between a TP and its matched

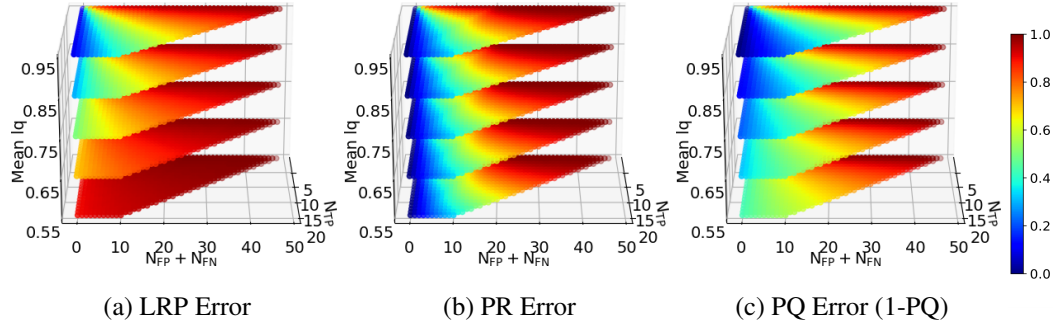


Figure 5.4: How LRP Error, PR Error (i.e. $1 - (\text{Precision} \times \text{Recall})$) and PQ Error (i.e. $1 - \text{PQ}$) behave over different inputs. Mean lq is the average localisation qualities of TPs. PR Error ignores localisation and PQ overpromotes classification compared to localisation. The space is uniformly discretized and error combinations of up to 20 ground truths and 50 detections are depicted.

ground truth can be measured by using a similarity function (preferably a metric to ensure the metricity of LRP), and (ii) at least one of the classification errors (i.e. FP error or FN error) matters for performance. Then, to use LRP Error, it is sufficient to ensure the similarity function satisfy the constraints for $\text{lq}(\cdot, \cdot)$ (see extension to other localisation quality functions). If either FP or FN error is not included in the task, then one can set the number of errors originating from the missing component (i.e. N_{FP} or N_{FN}) to 0 and proceed with Eq. 5.2.

5.6 A Comparison of LRP with AP and PQ

To better understand the behaviours of the studied performance measures (AP, PQ and LRP) and make comparisons, we plot them in the three dimensional space of mean localisation quality, N_{TP} and $N_{FP} + N_{FN}$ (Fig. 5.4). To facilitate comparison, we represent AP and PQ by their “error” versions, that is, for AP, we use “PR Error” which is $1 - \text{Precision} \times \text{Recall}$; and for PQ, we use “PQ Error” which is $1 - \text{PQ}$. Firstly, note that, PR Error stays the same as you move parallel to the “Mean lq” axis as expected (Fig. 5.4(b)). This is because PR Error, hence AP, uses the localisation quality just to validate TPs, and it does not take into account the quality above the TP-validation threshold. Secondly, PQ Error is lower than LRP Error at low “Mean

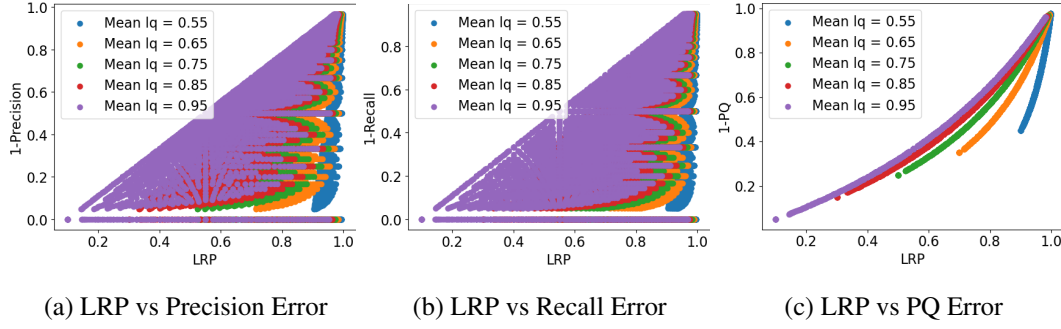


Figure 5.5: The relation of LRP with (a) precision error (1-precision), (b) recall error (1-recall) and (c) PQ error (1-PQ) using the examples from Fig. 5.4. LRP Error is an upper bound for precision, recall and PQ errors. Since LRP includes recall (precision) and localisation in addition to precision (recall) error, the correlation between precision (recall) error and LRP is not strong. On the other hand, with similar definitions LRP and PQ evaluate similarly, but still their difference increases when Mean lq decreases since PQ suppresses the effect of localisation by promoting classification more.

lq”, e.g. 0.55 and 0.65, low $N_{FP} + N_{FN}$ and large N_{TP} (Fig. 5.4(a) and (c)). This is due to the fact that PQ Error prefers to emphasize classification over localisation (as discussed in Section 5.4). On the other hand, as hypothesized in Section 5.4, the way how PQ overpromotes classification is inconsistent. To show this, we first demonstrate that LRP and PQ Errors have quite similar definitions. PQ Error can be written as (see Appendix G for the derivation):

$$1 - PQ = \frac{1}{\hat{Z}} \left(\sum_{i=1}^{N_{TP}} \frac{1 - lq(g_i, d_{g_i})}{1 - 0.50} + N_{FP} + N_{FN} \right), \quad (5.8)$$

where $\hat{Z} = 2N_{TP} + N_{FP} + N_{FN}$. Note that setting $\tau = 0.50$ and removing the coefficient of N_{TP} in $\hat{Z}$ (in red) results in $1 - PQ = LRP$ (Eq. 5.2), which implies very similar definitions for PQ and LRP Errors (and note that LRP was proposed before PQ): Eq. 5.8 presents that (i) the “total matching error” of PQ and LRP Errors are equal (Section 5.5.1 for total matching error), and (ii) PQ Error prefers doubling N_{TP} in the normalisation constant instead of normalizing the total matching error directly by its maximum value (i.e. $N_{TP} + N_{FP} + N_{FN}$) as done by LRP. Therefore, keeping the total matching error the same, the normalisation constant of PQ Error grows in-

consistently. In other words, the rates of the change of the total matching error and its maximum possible value are different. As suggested in our previous work [107], a consistent prioritization of a performance aspect can be achieved by including its coefficient to both total matching error (i.e. nominator) and its maximum value (i.e. denominator) as follows:

$$\frac{1}{Z} \left(\sum_{i=1}^{N_{TP}} \alpha_{TP} \frac{1 - \text{Iq}(g_i, d_{g_i})}{1 - \tau} + \alpha_{FP} N_{FP} + \alpha_{FN} N_{FN} \right) \quad (5.9)$$

where $Z = \alpha_{TP} N_{TP} + \alpha_{FP} N_{FP} + \alpha_{FN} N_{FN}$. Following the interpretation of LRP (Section 5.5.1), these coefficients imply duplicating each error source, hence the consistency between the total matching error and its maximum value is preserved (see Appendix E for more discussion).

In Fig. 5.5, we present the relationship of LRP with precision, recall and PQ Errors, which show that **LRP is an upper bound for all other error measures**. As a result, improving LRP can be considered a more challenging task than improving the other two error measures.

The comparison of LRP with AP and PQ in terms of the important features (Section 5.1.1) of a performance measure for visual object detectors is summarized in Table 5.2. Please refer to Sections 5.3, 5.4 and 5.5 for further discussion.

5.7 Experimental Evaluation

In this section, we first present the usage and discriminative abilities of the LRP Error on visual detection tasks in comparison to AP variants (Section 5.7.2) and PQ (Section 5.7.3). Then, we show that the performances of object detectors are sensitive to thresholding (Section 5.7.4) and provide a use-case of LRP-Optimal Thresholds (Appendix H). Also we analyse the additional overhead of LRP computation and the behaviour of LRP under different TP validation thresholds in Appendix H. We would like to note that our main motivation is to present insights on LRP and represent its evaluation capabilities rather than choosing which detection method is better.

Table 5.2: Comparison of AP, PQ and LRP in terms of desired properties. While LRP ensures all three, AP turns out to be limited in these properties.

Measure	Completeness	Interpretability	Practicality
AP	✗	✗	• limited to soft predictions • does not offer an optimal confidence score threshold • uses interpolation
PQ	✓	✓	• limited to panoptic segmentation • overpromotes the classification performance inconsistently
LRP	✓	✓	✓

5.7.1 Evaluated Models, Datasets and Performance Measures

Evaluated Models: We use LRP to evaluate 35 state of the art (SOTA) methods (i.e. 32 methods to evaluate soft predictions in Section 5.7.2 and 3 methods to evaluate hard predictions in Section 5.7.3) obtained from three widely-used repositories: mmdetection [100], detectron [161] and detectron2 [14]³. We provide the corresponding repository of each model in Table 5.3 to facilitate the usage and reproduction of our results. We do not retrain the models but use the already trained instances provided in their repositories. We use R50, R101 and X101 as abbreviations for ResNet-50, ResNet-101 and ResNext-101 backbones, respectively.

Datasets: We use the COCO [13] dataset, one of the most widely used detection datasets which provides ground-truth annotations for 80 object classes, for all four visual detection tasks that we are interested in. The training and testing sets of the used models are *COCO 2017 train* ($\sim 118k$ images) and *COCO 2017 val* (5k images). Note that the panoptic segmentation task additionally requires the annotation of the ‘stuff’ classes, which are not included in the 80 object classes. Hence, for panoptic segmentation, we follow the configuration of detectron2, where 53 additional stuff classes are acquired from the COCO-stuff dataset [46].

Performance Measures: On tasks with hard-predictions (i.e. panoptic segmentation), we compare LRP with PQ. On the remaining three tasks, namely object detection, keypoint detection and instance segmentation, all of which are soft-prediction tasks, we compare LRP with AP. Since AP does not explicitly have performance components, we include the following measures to facilitate comparison:

1. AP_{75} , in which τ , the TP validation threshold, is 0.75. AP_{75} is a popular measure to represent the localisation accuracy of a method.
2. AP_{50} , to represent the classification component.
3. AR_r^C (Average Recall) where r is the number of top-scoring detections to include in the computation of AR. Note that AR_r^C is also COCO-style (i.e. averaged over 10 τ thresholds - see the definition of COCO-style AP, denoted by AP^C , in Section 5.3).

³ One exception is aLRP Loss, for which we use our own implementation.

Table 5.3: The repositories of models that we downloaded, evaluated and utilized for comparison. The only exception is aLRP Loss [156], for which we used our own implementation [162]. The models are listed in alphabetical order.

Repository	Method
mmdetection [100]	ATSS [42]
	Cascade Mask R-CNN [20]
	Cascade R-CNN [20]
	FCOS [160]
	FreeAnchor [39]
	GHM [68]
	Grid R-CNN [163]
	Guided Anchoring [75]
	Hybrid Task Cascade [34]
	Libra R-CNN [32]
	Mask R-CNN [35]
	Mask Scoring R-CNN [41]
	NAS-FPN [88]
detectron [161]	RPDet [164]
	SSD [24]
detectron2 [14]	Faster R-CNN[19]
	RetinaNet [18]
	Keypoint R-CNN
	Panoptic FPN [115]

5.7.2 Evaluating Soft Predictions on Object Detection, Keypoint Detection and Instance Segmentation Tasks

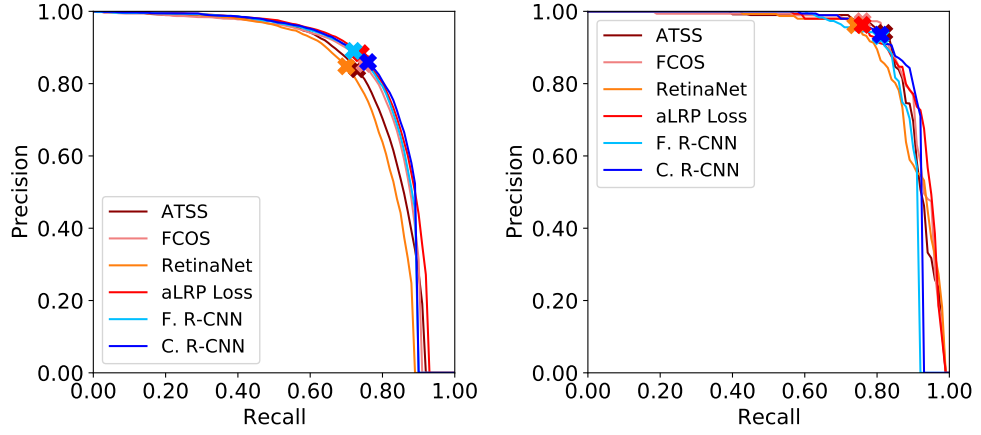
This section compares oLRP with AP&AR variants for soft predictions and shows that oLRP is more discriminative and interpretable. The structure of this section is based on the limitations (Section 5.1.2) and analysis of AP (Section 5.3.2) in terms of the important features (Section 5.1.1). While demonstrating these limitations and comparing with oLRP, we use both detector-level results (Table 5.4) and class-level

Table 5.4: AP and oLRP performances of several methods for soft-prediction tasks (i.e. object detection, keypoint detection and instance segmentation) on COCO 2017 val. For AR_r^C , $r = 100$ for object detection and instance segmentation, while it is 20 for keypoint detection.

Method	Backbone	Epoch	AP & AR			oLRP & Components				
			AP ^C ↑	AP ₅₀ ↑	AP ₇₅ ↑	AR _h ↑	oLRP ↓	oLRP _{Loc} ↓	oLRP _{FP} ↓	oLRP _{FN} ↓
Object Detection: <i>One Stage Methods:</i> SSD-300 [24] SSD-512 [24] RetinaNet [18] RetinaNet [18] RetinaNet [18] ATSS [42] RetinaNet [18] NAS-FPN [88] GHM [68] FreeAnchor [39] FCOS [160] RPDet [164] aLRP Loss [156] <i>Two Stage Methods:</i> Faster R-CNN [19] Faster R-CNN [19] Faster R-CNN [19] Faster R-CNN [19] Libra R-CNN [32] Grid R-CNN [163] Guided Anchoring [75] Cascade R-CNN [20] Cascade R-CNN [20]	VGG16	120	25.6	43.8	26.3	37.5	78.4	20.6	37.1	57.9
	VGG16	120	29.4	49.3	31.0	42.5	75.4	19.7	32.8	53.6
	R50	12	35.7	54.7	38.5	52.0	71.0	17.0	29.1	50.0
	R50	24	35.7	54.9	38.2	51.4	70.6	17.1	28.4	49.6
	X101	24	39.2	59.2	41.8	53.5	67.5	16.1	24.5	46.3
	R50	12	39.4	57.6	42.8	58.3	68.6	15.4	30.3	46.6
	X101	12	39.8	59.5	43.0	54.8	67.6	16.1	25.3	46.2
	R50	50	40.5	58.4	43.1	55.6	66.7	14.8	26.6	46.3
	X101	12	41.4	60.9	44.2	57.7	66.3	15.6	27.1	44.2
	X101	12	41.9	61.0	45.0	58.6	66.0	15.2	26.4	44.5
	X101	24	42.5	62.1	45.7	58.2	64.4	14.9	25.4	41.9
	X101	24	44.2	65.5	47.8	58.7	63.3	15.4	23.4	39.5
	X101	100	45.4	66.6	48.0	60.3	62.5	15.1	23.2	39.5
	R50	24	37.9	59.3	41.1	51.0	68.8	17.4	25.7	45.4
	R101	12	39.4	61.2	43.4	52.6	67.6	17.2	24.2	44.3
	R101	24	39.8	61.3	43.3	52.5	67.3	16.8	25.5	43.4
	X101	12	41.3	63.7	44.7	54.6	66.2	17.1	24.9	41.5
X101	12	42.7	63.7	46.9	56.0	65.1	15.8	24.3	41.6	
X101	24	43.0	61.6	46.7	56.7	64.2	14.4	24.7	42.3	
X101	12	43.9	63.7	48.3	59.9	64.4	14.8	25.6	41.8	
X101	20	44.5	63.2	48.5	56.9	63.3	14.3	25.4	41.0	
X101	12	44.7	63.6	48.9	57.4	63.2	14.4	23.9	40.9	
Keypoint Detection: Keypoint R-CNN Keypoint R-CNN Keypoint R-CNN Keypoint R-CNN	R50	12	64.0	86.4	69.3	71.0	44.8	12.8	10.8	18.6
	R50	37	65.5	87.2	71.1	72.4	43.0	12.3	10.4	17.3
	R101	37	66.1	87.4	72.0	73.1	42.0	11.9	9.0	17.8
	X101	37	66.0	87.3	72.2	73.2	41.9	11.7	8.8	18.1
Instance Segmentation: Mask R-CNN [35] Mask R-CNN [35] Mask R-CNN [35] Cascade Mask R-CNN [20] Mask Scoring R-CNN [41] Hybrid Task Cascade [34]	R50	24	35.4	56.4	37.9	48.1	70.7	18.5	28.5	47.0
	R101	24	36.6	57.9	39.1	48.8	69.4	18.2	25.9	46.3
	X101	12	38.4	60.6	41.3	50.3	67.8	18.3	24.9	43.5
	X101	20	39.5	61.3	42.5	50.5	66.8	18.0	24.3	42.1
	X101	12	39.5	60.5	42.6	50.1	67.5	17.9	24.5	43.3
	X101	20	43.8	66.8	47.1	57.4	63.6	17.0	23.4	37.9

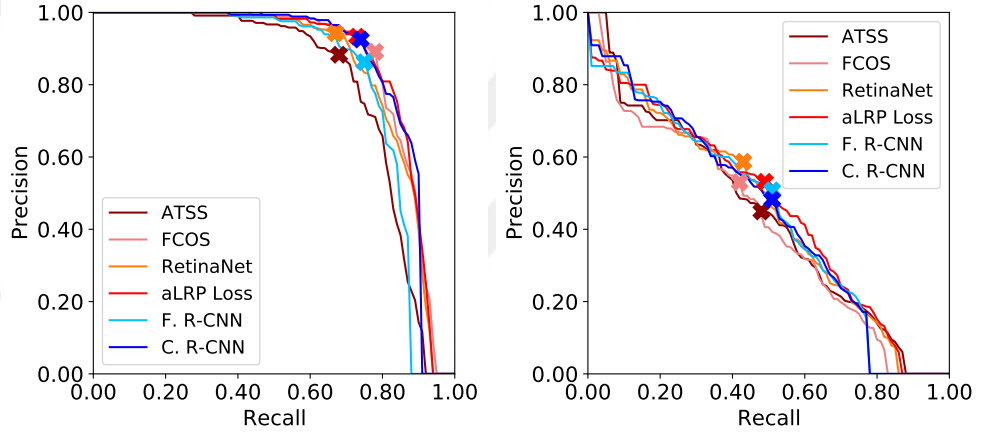
Table 5.5: AP and oLRP performances of several object detectors for two selected classes, namely, “person” and “broccoli”, on COCO 2017 val. The horizontal line splits one- and two-stage object detectors.

Class	Method	Backbone	Epoch	AP & AR				oLRP & Components				
				AP ^C ↑	AP ₅₀ ↑	AP ₇₅ ↑	AR ₁₀₀ ^C ↑	oLRP ↓	oLRP _{Loc} ↓	oLRP _{FP} ↓	oLRP _{FN} ↓	
Person	ATSS [42]	R50	12	54.7	81.4	59.3	64.8	55.6	15.4	14.8	27.8	
	FCOS [160]	X101	24	55.3	82.4	59.1	64.2	53.5	15.0	13.0	26.2	
	RetinaNet [18]	X101	12	51.1	79.0	54.6	60.1	58.3	16.3	14.1	31.0	
	aLRP Loss [156]	X101	100	57.7	84.3	61.7	66.6	52.1	14.5	11.2	26.3	
	Faster R-CNN [19]	R101	24	53.8	82.8	57.9	61.2	55.3	16.3	11.1	27.7	
	Cascade R-CNN [20]	X101	12	57.8	83.4	62.7	65.2	52.1	14.6	13.5	24.5	
Broccoli	ATSS [42]	R50	12	24.1	43.3	24.0	55.0	81.6	20.1	53.2	52.9	
	FCOS [160]	X101	24	21.9	41.5	21.6	48.4	82.4	21.4	45.1	58.7	
	RetinaNet [18]	X101	12	22.1	44.4	19.7	49.3	81.3	21.8	41.3	56.7	
	aLRP Loss [156]	X101	100	24.3	45.5	23.5	51.7	80.2	21.2	45.7	51.6	
	Faster R-CNN [19]	R101	24	22.0	43.6	19.3	43.8	81.2	22.7	49.2	48.4	
	Cascade R-CNN [20]	X101	12	24.3	43.9	25.5	47.2	80.2	20.1	51.8	48.7	



(a) Person

(b) Zebra



(c) Bus

(d) Broccoli

Figure 5.6: Example PR curves for four classes. The curves are drawn for $\tau = 0.50$. The lines with different red tones represent one-stage detectors whereas those with blue tones correspond to two-stage detectors. While AP_{50} considers the area under the curve, LRP combines localisation, recall and precision errors, and hence optimal LRP points, marked with crosses, are found in the top right part of the curves. Detailed results of “person” and “broccoli” can be found in Table 5.5. F. R-CNN: Faster R-CNN, C. R-CNN: Cascade R-CNN.

results (Table 5.5) of the SOTA methods. For the detector-level performance comparison, we present the results of 32 SOTA visual detectors on all three visual detection tasks. In order to provide insight on oLRP and its components and illustrate its usage at the class level, we select a subset of six object detectors by ensuring diversity (e.g. different backbones, one- and two-stage detectors, different assignment and sampling strategies etc.) and evaluate their performance on “person” and “broccoli” classes.

5.7.2.1 Analysis with respect to Completeness

AP loosely includes the localisation quality (Section 5.3.2). Here we discuss the benefits of directly using the localisation quality as an input to the performance measure.

Firstly, to see how AP_{50} fails to include localisation quality precisely, we consider the following three detectors with equal AP_{50} (63.7%) in Table 5.4: Faster R-CNN (X101-12), Libra R-CNN and Guided Anchoring. oLRP and AP^C , which take into account the localisation quality, rank these three detectors differently (Table 5.4). Therefore, AP_{50} should not be selected as the single performance measure for benchmarking since it neglects localisation.

To illustrate the drawback of AP^C or AP_{75} in terms of localisation, note that while neither AP^C nor AP_{75} assigns the largest performance value to NAS-FPN among one-stage object detectors, this detector has the least average localisation error (oLRP_{Loc} - see Table 5.4): e.g. GHM outperforms NAS-FPN by 1.1% in terms of AP_{75} , while its average localisation performance is 0.8% lower than NAS-FPN. Therefore, AP^C and AP_{75} , too, may fail to appropriately compare methods in terms of localisation quality.

Using oLRP is easier and more intuitive than using the AP variants mentioned above: (i) Unlike these AP variants, oLRP Error consistently and precisely (not loosely) combines localisation, FP and FN errors, and in this case, the performance gap between NAS-FPN and GHM reduces to 0.4% in terms of oLRP (i.e. while GHM outperforms NAS-FPN by 1.1% with respect to AP^C) thanks to the localisation performance of NAS-FPN. (ii) Different from all AP variants, oLRP_{Loc} quantifies the localisation error precisely and allows direct comparison among methods, classes,

etc.: e.g. NAS-FPN outperforms GHM by 0.8% in terms of oLRP_{Loc} . (iii) One can easily interpret oLRP_{Loc} both at the class- and detector-level: for example, for ATSS, IoU is $1 - 0.154 = 0.846$ and $1 - 0.201 = 0.799$ for the “person” and “broccoli” classes respectively (Table 5.5).

Finally, being an interpretable localisation measure, oLRP_{Loc} can facilitate analysis of detectors. For example, in Table 5.4, we can easily notice for oLRP_{Loc} that instance segmentation task has room for improvement in terms of localisation compared to other tasks. In particular, even the best performing instance segmentation method, Hybrid Task Cascade (HTC), yields 17.0% oLRP_{Loc} error which corresponds to a mediocre localisation error for the object detectors and keypoint detectors, typically achieving 14.3% and 11.7% oLRP_{Loc} errors, respectively. With this 17.0%, HTC has a similar localisation performance with RetinaNet (R50-24) in terms of oLRP_{Loc} . Again, HTC outperforms RetinaNet (R50-24) by around 10% AP_{75} , suggesting that the same deduction cannot be obtained by AP_{75} .

5.7.2.2 Analysis with respect to Interpretability

This section presents insights about the interpretability of oLRP Error compared to AP (see Section 5.3.2 for a discussion on the limited interpretability of AP).

While any AP variant does not provide insight on the performance of an object detector, the components of oLRP provide useful insight on the performance. To illustrate, we compare two object detectors with equal AP^{C} , ATSS and Faster R-CNN (R101-12) (see in Table 5.4 that both have 39.4% AP^{C}), using AP & AR based measures and oLRP components as follows:

- Faster R-CNN (R101-12) outperforms ATSS by 3.6% and 0.6% in terms of both AP_{50} and AP_{75} and ATSS outperforms Faster R-CNN (R101-12) by around 6% with respect to $\text{AR}_{100}^{\text{C}}$. Note that a clear conclusion (i.e. quantifying which detector is better on which performance aspect) is not possible with these AP & AR based measures since AP_{50} and AP_{75} are combinations of precision & recall and $\text{AR}_{100}^{\text{C}}$ a combination of recall & localisation quality.
- As for oLRP , Faster R-CNN (R101-12) outperforms ATSS by 6.1% and 2.3%

in terms of FP and FN Errors respectively, and ATSS has 1.8% better localisation performance than Faster R-CNN (R101-12). Since each component corresponds to one performance aspect, one can easily deduce that while Faster R-CNN has better classification performance (wrt. both precision and recall), ATSS localises objects better. Overall, combining these components consistently, in this case, oLRP prefers Faster R-CNN (R101-12) over ATSS by 1.1% while they have the same AP^C .

In addition, $oLRP_{FP}$ and $oLRP_{FN}$ provide insight on the structure of the PR curve by representing the point on the PR curve where the minimal LRP is achieved. To illustrate, for all methods, the “person” class has lower FP & FN error values than the “broccoli” class, implying the oLRP point of the “person” PR curve to be closer to the top-right corner. To see this, note that Faster R-CNN has 11.1% and 27.7% FP and FN error values, respectively for the “person” class (Table 5.5). Thus, without looking at the curve, one may conclude that the oLRP point resides at $1 - 0.111 = 0.889$ precision and $1 - 0.277 = 0.723$ recall. For the “broccoli” curve, the oLRP point is achieved at $1 - 0.492 = 0.508$ and $1 - 0.484 = 0.516$ as precision and recall, respectively. Unlike the “person” class, these values suggest that the optimal point of the “broccoli” class is around the center of the PR range (cf. Fig. 5.6(a) and (d)). Hence, $oLRP_{FP}$ and $oLRP_{FN}$ are also easily interpretable and in such a way, exhaustive examination of PR curves can be alleviated.

Similar to $oLRP_{loc}$, $oLRP_{FP}$ and $oLRP_{FN}$ facilitate analysis as well, which is not straightforward by using AP&AR based measures. Suppose that we want to compare precision and recall performances of the visual object detectors. Comparing $oLRP_{FP}$ and $oLRP_{FN}$, it is obvious that current object detectors have significantly lower precision error than recall error (i.e. $oLRP_{FP} - oLRP_{FN}$ is around 15% to 20 % for object detection and instance segmentation, 7% to 8 % for keypoint detection - see Table 5.4). Given AP&AR based measures, one alternative can be to compare AP_{50} with AR_{100}^C , which is again hampered by the loose and indirect combination of the performance aspects: Note that while $oLRP_{FP}$ and $oLRP_{FN}$, isolating errors with respect to performance aspects, assign significantly more recall error than precision error for ATSS ($oLRP_{FN} - oLRP_{FP} = 16.5\%$ - see Table 5.4), AP- and AR-based performance measures favor recall performance over precision performance

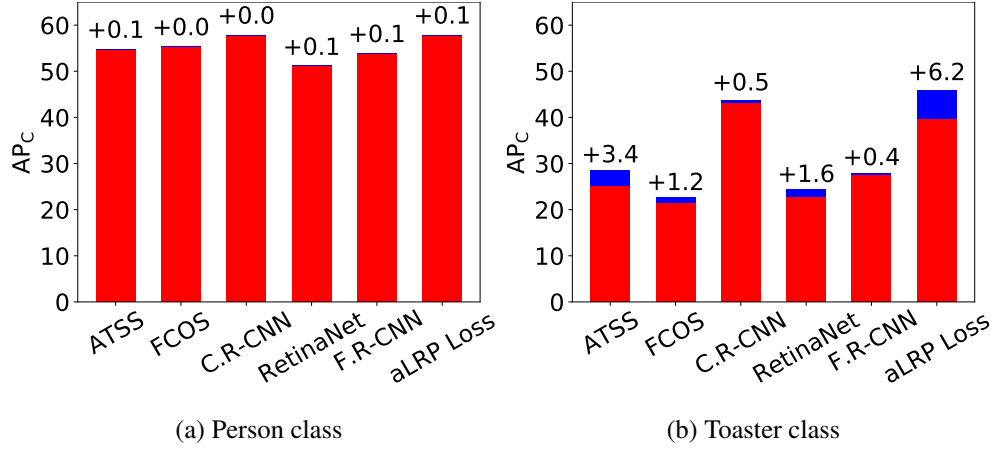


Figure 5.7: The effect of interpolating the PR curve on AP^C on (a) the “person” class, the class with the most number of examples, (b) the “toaster” class, the class with the least number of examples. Red: AP without interpolation. Blue: Additional AP^C after interpolation. The numbers on the bars indicate this additional AP^C points due to interpolation. While the effect of interpolation is negligible for the “person” class, there is a significant effect of interpolation (i.e. 2.2% AP^C on average, up to 6.2%) on the performance of the toaster class (i.e. the class with the minimum number of examples) for all the detectors. C.R-CNN: Cascade R-CNN, F.R-CNN: Faster R-CNN.

($AR_{100}^C > AP_{50}$). Therefore, indirect contribution of the performance aspects makes the analysis more difficult for AP- and AR-based measures, while oLRP and components are easier to interpret and compare.

5.7.2.3 Analysis with respect to Practicality

This section presents how LRP variants can handle the practical disadvantages of AP (see Section 5.3.2 for a discussion why AP is limited in these practical issues).

Evaluating Hard Predictions: We discuss how LRP evaluates hard predictions in Section 5.7.3.

Thresholding Visual Object Detectors: We discuss our class-specific thresholding approach using LRP-Optimal thresholds in Section 5.7.4.

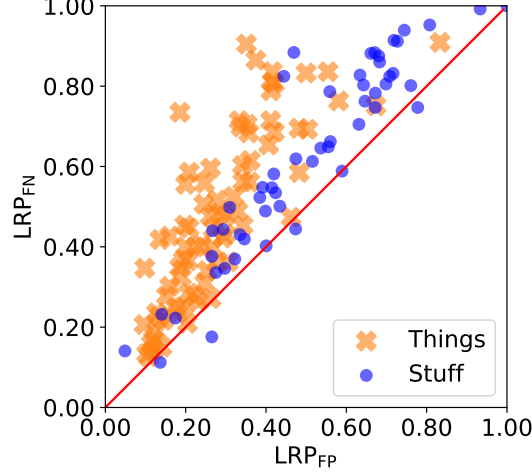


Figure 5.8: Class-level LRP_{FP} vs. LRP_{FN} comparison. Overall, there is a tendency of larger LRP_{FN} error than LRP_{FP} error. This is more obvious for things classes. It is not possible to make the same observation using RQ.

The Effect of Interpolation: In order to present the effect of interpolation on classes with relatively fewer number of examples, we compute AP^{C} of the same six detectors from class-level comparison table (Table 5.5) with and without interpolation on two classes: (i) the “person” class as the class with maximum number examples in COCO val 2017 (i.e. 21554 examples), and (ii) the “toaster” class, the class with the minimum number of examples (i.e. 17 examples). The AP^{C} difference is presented in Fig. 5.7, in which the significant effect of interpolation on the class with the less number of examples is clearly observed: (i) While the average AP^{C} difference over detectors between with and without interpolation is almost negligible for person class (i.e. less than 0.1% AP^{C}), it is around $35\times$ more for the toaster class (2.2% AP^{C}), (ii) There is even 6.2% jump for the aLRP Loss after interpolation. Note that this corresponds to around a superficial 20% relative performance improvement (from 33.4% to 39.6%) in terms of AP^{C} . Therefore, while the AP variants are sensitive to interpolation especially for the rare classes in the dataset, oLRP does not employ interpolation. This issue of AP should especially be taken care of for visual detection datasets with rare classes such as LVIS [45]. Note that unlike AP, oLRP computation is exact (Section 5.5.3).

Table 5.6: Detector-level performance results of panoptic segmentation methods as hard predictions. For each method, besides an overall average in All classes, we provide the performance of things and stuff as well.

Method	Backbone	Epoch	Type	PQ & Components			LRP & Components			
				PQ $\uparrow$	SQ $\uparrow$	RQ $\uparrow$	LRP $\downarrow$	LRP _{Loc} $\downarrow$	LRP _{FP} $\downarrow$	LRP _{FN} $\downarrow$
Panoptic FPN [115]	R50	12	All	39.4	77.8	48.3	77.5	22.2	40.2	57.2
			Things	45.9	80.9	55.3	72.7	18.1	29.4	51.7
			Stuff	29.6	73.3	37.7	84.6	25.3	54.3	65.5
Panoptic FPN [115]	R50	37	All	41.5	79.1	50.5	75.9	20.3	38.6	55.2
			Things	48.3	82.2	57.9	70.8	17.8	29.3	49.1
			Stuff	31.2	74.4	39.4	83.5	24.2	52.6	64.4
Panoptic FPN [115]	R101	37	All	43.0	80.0	52.1	74.6	19.4	37.0	53.6
			Things	49.7	82.9	59.2	69.4	17.1	28.4	47.6
			Stuff	32.9	75.6	41.3	82.3	22.9	50.2	62.7

5.7.3 Evaluating Hard Predictions on Panoptic Segmentation Task

In this section, we apply LRP Error to panoptic segmentation task to present its ability to evaluate hard predictions and also compare LRP with PQ. In particular, we evaluate three different variants of Panoptic FPN [115] using both LRP and PQ, and present the results in Table 5.6 in three groups: (i) “All” includes all 133 classes, (ii) “Things” includes 80 object classes, and (iii) “Stuff” includes the remaining 53 classes, normally counted as background by other detection tasks. Similar to oLRP, we follow our analysis on PQ (Section 5.4.2) except the superiority of LRP on evaluating and thresholding soft predictions, which we discuss in Sections 5.7.2 and 5.7.4 respectively.

5.7.3.1 Analysis with respect to Interpretability

The RQ component of PQ, the F-measure, does not provide discriminative information on precision and recall errors. On the other hand, LRP presents more insight on these errors with its FP and FN components. To illustrate, all Panoptic FPN variants suffer from the recall error more than the precision error, and this is more obvious for “things” classes: (i) Table 5.6 shows that $LRP_{FN} > LRP_{FP}$ for all methods in class groups. (ii) While the gap between LRP_{FP} and LRP_{FN} for “stuff” classes is around 10%, it is around 20% for “things” classes for all detectors. (iii) Finally, the same difference between “things” and “stuff” classes can easily be observed at the class-level in Fig. 5.8 where the error is skewed towards LRP_{FN} . Therefore, we argue that LRP FP and FN components present more insight than RQ.

5.7.3.2 Analysis with respect to Practicality

Since the definitions of LRP and PQ are similar (Eq. 5.8), LRP and PQ generally rank the detectors and classes similarly. However, we observed certain differences owing to the over-promotion of TPs by PQ with its discontinuous nature: (i) We observed that 205 pair of classes for which the evaluation results of LRP and PQ conflict (i.e. $(PQ_i < PQ_j)$ and $(LRP_j > LRP_i)$ where the subscript represents the class label). As expected (see also Fig. 5.4(a,c) and Fig. 5.5(c)), PQ favors classes with more TPs

compared to LRP, and LRP favors the classes with better localisation performance.

(ii) In some cases, the difference between the results of AP and PQ (i.e. $(PQ_i - PQ_j) - (LRP_j - LRP_i)$) can be large. For example, while the “bicycle” and “orange” classes have 40.6% and 34.1% PQ respectively (i.e. “bicycle” outperforms by 6.5), their LRP values are 82.4% and 81.5% (i.e. “orange” outperforms by 0.9%). Overall these imply a 7.4% difference between AP and LRP. The over-promotion of TPs by PQ can also be observed by examining its components: While the RQ of “bicycle” and “orange” are 55.9% and 38.4% respectively (i.e. “bicycle” outperforms by 17.5%), SQ are 72.7% and 88.9% (i.e. “orange” outperforms by 16.2%). These results suggest that while “bicycle” can be classified better than “orange”, the localisation performance of “bicycle” is poorer. As a result, while LRP results are similar, PQ promotes the class with better classification (i.e. “bicycle”) by 6.5% and assigns a lower priority to localisation.

5.7.4 Thresholding Visual Object Detectors

In this section, we show that (i) the performances of visual detectors is sensitive to thresholding, (ii) the thresholds need to be set in a class-specific manner and (iii) LRP-Optimal thresholds can be used to alleviate this sensitivity.

Firstly, to see why visual object detectors can be sensitive to thresholding, Fig. 5.9 shows on the “person” class how performance (in terms of LRP and its components) evolves with different score thresholds (s) on different detectors. We observe in Fig. 5.9(d) that the performances of some detectors (e.g ATSS, FCOS, aLRP Loss) improve and degrade rapidly around s^* , a situation which implies the sensitivity of these detectors with respect to the threshold choice (i.e. model selection). For example, for ATSS, choosing a threshold larger than 0.50 has a significant impact on the performance, and even a threshold larger than 0.75 results in a detector with no TPs. Therefore, model selection is important for practical usage of object detectors.

Secondly, for a given detector, the variance of the LRP-Optimal thresholds over classes can be large (Fig. 5.10- especially see RetinaNet in (b) and Cascade R-CNN in (d)). Thus, a general, fixed threshold for all classes can not provide optimal performance for all classes. Class-specific thresholding is required for optimal performance

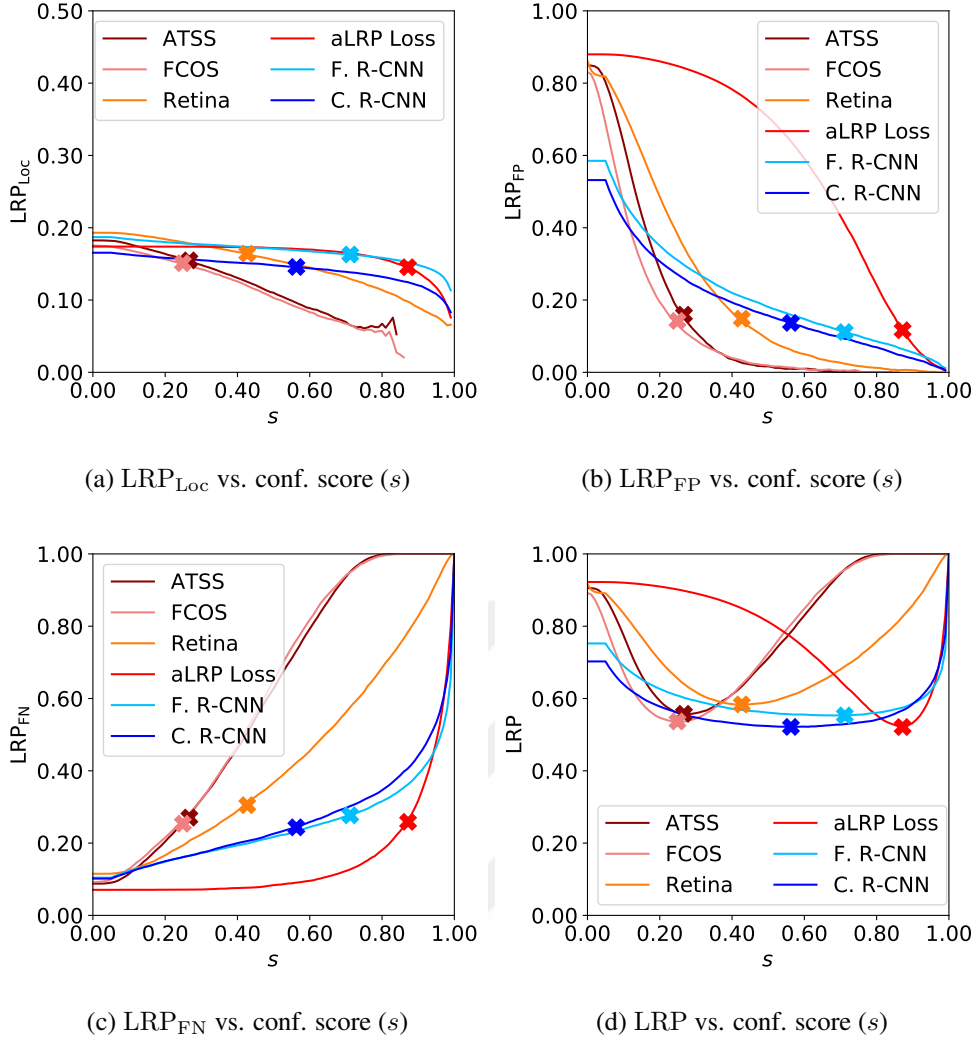


Figure 5.9: s-LRP curves of different object detectors for the “person” class. LRP combines localisation error (a), precision error (b) and recall error (c) over entire s domain in (d). The minimum-achievable LRP error in (d) is coined as oLRP (i.e. marked by “x”). Note that for some detectors (e.g. ATSS), the performance with respect to s changes abruptly in (d), hence, for some object detectors, the performance is very sensitive to thresholding. The lines with a different red tone represent a one-stage detector, while blue tones correspond to two-stage detector F. R-CNN: Faster R-CNN, C. R-CNN: Cascade R-CNN

of visual object detectors.

Based on these observations, in Appendix H, we present a use-case of LRP-Optimal

Thresholds on a video object detector which, first, collects thresholded detections from a conventional object detector, and then associates detection results between frames. On this use-case, we show that using class-specific LRP-Optimal thresholds significantly improves performance (up to around 9 points AP_{50} and 4 points oLRP) compared to using general, class-independent thresholding.

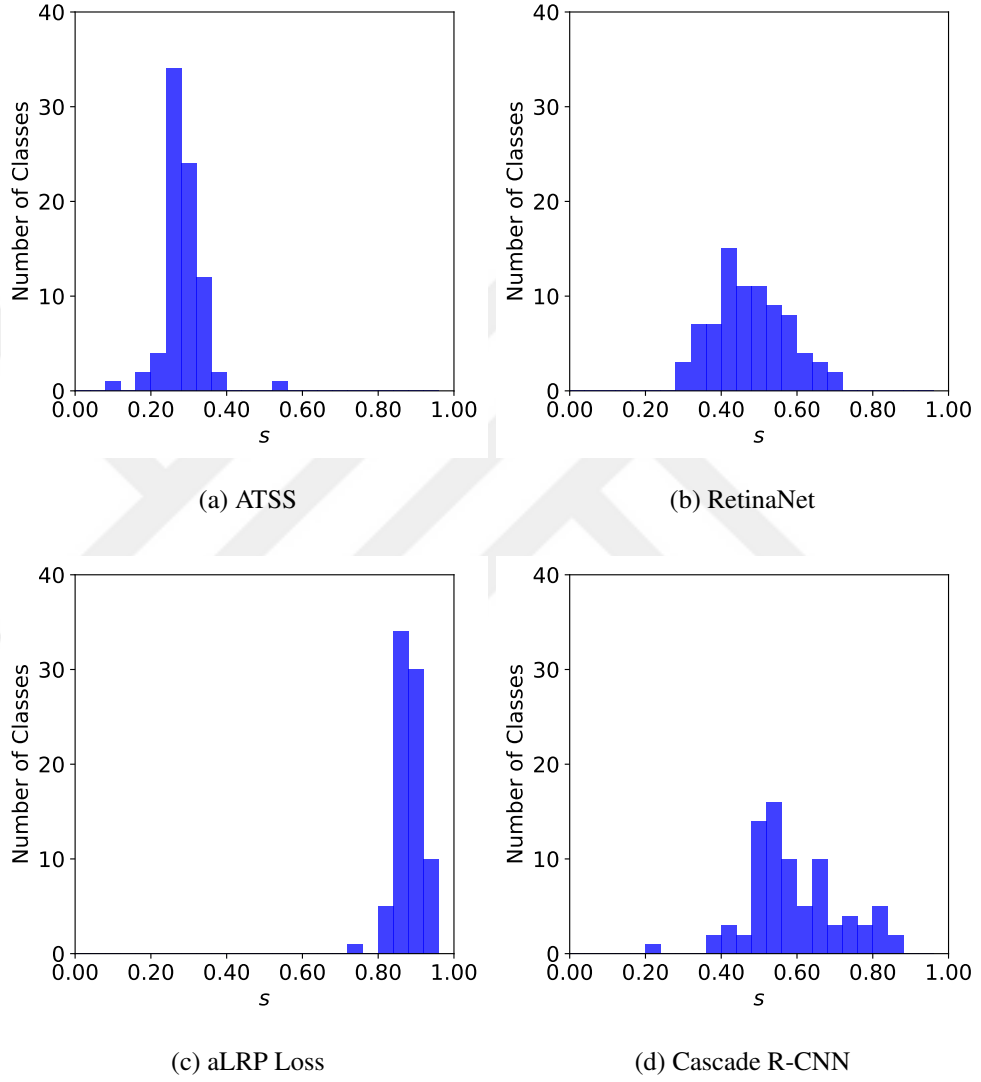


Figure 5.10: The distributions of the class-specific LRP-Optimal Thresholds (s^*) for different methods. The variance of the LRP-Optimal thresholds can be large among classes. Thus, using a single general threshold for all classes will provide sub-optimal results.

5.8 Conclusion

In this chapter, we introduced a novel performance metric, LRP Error, to evaluate *all* visual detection tasks as an alternative to the widely-used measures AP and PQ. LRP Error has a number of advantages which we demonstrated in this chapter: LRP Error (i) is “complete” in the sense that it precisely takes into account all important performance aspects (i.e. localisation quality, recall, precision) of visual object detectors, (ii) is easily interpretable through its components, and (iii) does not suffer from the practical drawbacks of AP and PQ.



CHAPTER 6

IDENTITY UPDATE: USING ERROR-DRIVEN UPDATE WITH BACKPROPAGATION TO OPTIMISE RANKING-BASED LOSS FUNCTIONS

Ranking two numbers, n_i and n_j , is an essential operation indicating whether n_j is larger than n_i or not:

$$\Psi(n_i, n_j) = \begin{cases} 1, & \text{if } n_j \geq n_i \\ 0, & \text{otherwise,} \end{cases} \quad (6.1)$$

and accordingly, we can classify the set of loss functions requiring this ranking operation to supervise deep learning models as *ranking-based loss functions*. The main characteristic of these loss functions is their ranking-based error definitions. In particular, assuming that a classifier generates two outputs, $\hat{s}_i$ and $\hat{s}_j$, as the predictions corresponding to different classes and $\hat{s}_i$ is the ground truth class; a ranking-based loss function supervises the model following a ranking error to increase $\hat{s}_i$ (i.e. promote i) and/or decrease $\hat{s}_j$ (i.e. promote j) when the ground class is not the predicted class (i.e. $\hat{s}_i \leq \hat{s}_j$) and assigns no error when the input is classified accurately (i.e. $\hat{s}_i > \hat{s}_j$).

Using the performance measures of visual detectors (e.g. AP and LRP) as ranking-based loss functions (see Chapter 5 for their definitions) is promising since (i) the network is, then, supervised as it is tested, (ii) the aforementioned ranking-based error definition has the potential to inherently handle severe class imbalance and (iii) the performance measures typically do not have hyper-parameters, thereby alleviating the tuning challenge of visual detectors (Chapter 1). On the other hand, due to its piecewise linearity, the derivative of Eq. 6.1 wrt. its inputs is either 0 or ∞ making direct incorporation of such losses in Stochastic Gradient Descent using backpropagation

infeasible.

Recently, Chen et al. [37] show that AP can be optimised by employing the error-driven optimisation from Perceptron Learning [44]. Their AP Loss is a ranking-based loss function to optimise the ranking of the classification outputs, and as expected, it provides balanced training between positives and negatives with less number of hyper-parameters than its score-based counterpart Focal Loss [18] (i.e. 1 vs. 2 hyper-parameters).

In this chapter, we first revisit how AP Loss is computed and optimised (Section 6.1) by Chen et al. [37]; then in Section 6.2, we identify the drawbacks of this computation and optimisation and propose our Identity Update as a more general and simple method to optimise ranking-based loss functions. We also prove that the loss functions optimised by our Identity Update has a natural balance between positives and negatives during training in terms of gradient magnitudes. Finally, in Section 6.3, we compute and obtain the gradients of AP Loss following our Identity Update as an example on how our method simplifies the optimisation of ranking-based loss functions. As for the generalization, we compute and optimise our aLRP and RS Losses respectively as more complex loss functions than AP Loss in Chapters 7 and 8 respectively. This chapter basically combines our findings in our two previous papers, aLRP Loss [156] and RS Loss [165].

6.1 Previous Work: Definition, Computation and Optimisation of AP Loss

Definition of AP Loss. AP Loss [37] directly optimises the following loss for AP with IoU thresholded at 0.50:

$$\mathcal{L}_{\text{AP}} = 1 - \text{AP}_{50} = 1 - \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \text{precision}(i) = 1 - \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \frac{\text{rank}^+(i)}{\text{rank}(i)}, \quad (6.2)$$

where $\mathcal{P}$ is the set of positive examples identified based on the assignments of the proposals (i.e. anchors/points/RoIs); $\text{rank}^+(i)$ and $\text{rank}(i)$ are respectively the ranking positions of the i th sample among positives and all samples. $\text{rank}^+(i)$ and $\text{rank}(i)$ can be easily defined using the ranking operation $\Psi(0, x_{ij})$ (Eq. 6.1), which compares

the difference transforms,

$$x_{ij} = \hat{s}_j - \hat{s}_i, \quad (6.3)$$

between the logit of i th prediction ($\hat{s}_i$) and the logit of each other sample ($j \in \mathcal{P} \cup \mathcal{N}$ for $\text{rank}(i)$ and $j \in \mathcal{P}$ for $\text{rank}^+(i)$) with 0. More particularly, $\text{rank}(i)$ ¹ is:

$$\text{rank}(i) = \sum_{j \in \mathcal{P} \cup \mathcal{N}} H(x_{ij}), \quad (6.4)$$

where, in practice, $H(x_{ij})$ is a smoothed version of $\Psi(0, x_{ij})$ around 0,

$$H(x_{ij}) = \begin{cases} 0, & x_{ij} < -\delta \\ \frac{x_{ij}}{2\delta} + 0.5, & -\delta \leq x_{ij} \leq \delta \\ 1, & \delta < x_{ij}. \end{cases} \quad (6.5)$$

such that δ aims to enforce a margin between $\hat{s}_i$ and $\hat{s}_j$ and $\mathcal{N}$ is the set of negatives. $\text{rank}^+(i)$ can be defined similarly over $j \in \mathcal{P}$:

$$\text{rank}(i) = \sum_{j \in \mathcal{P}} H(x_{ij}). \quad (6.6)$$

Computation of AP Loss (Forward Pass): Chen et al. [37] introduced “primary terms” (L_{ij}) of AP Loss as the “activation function” applied to each difference transform, x_{ij} , and then, aimed to rewrite the loss value, $\mathcal{L}_{\text{AP}}$, as the normalized summation over all primary terms. Following Eq. 6.2, the primary terms of AP loss can be

¹ While AP Loss [37] and our earlier work [156] added 1 for the example itself in rank-based function computations (e.g. $\text{rank}(i) = 1 + \sum_{j \in \mathcal{P} \cup \mathcal{N}, i \neq j} H(x_{ij})$); for clarity and consistency in terms of the contribution of examples, we included the contribution of the example itself similar to all other examples by applying $H(\cdot)$ in our most recent paper, RS Loss [165]. Eq. 6.4 sets an example on $\text{rank}(i)$.

obtained using algebraic manipulations as follows:

$$\mathcal{L}_{\text{AP}} = 1 - \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \frac{\text{rank}^+(i)}{\text{rank}(i)} \quad (6.7)$$

$$= \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \frac{\text{rank}(i) - \text{rank}^+(i)}{\text{rank}(i)} \quad (6.8)$$

$$= \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \frac{\sum_{j \in \mathcal{P} \cup \mathcal{N}} \text{H}(x_{ij}) - \sum_{j \in \mathcal{P}} \text{H}(x_{ij})}{\text{rank}(i)} \quad (6.9)$$

$$= \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \frac{\sum_{j \in \mathcal{N}} \text{H}(x_{ij})}{\text{rank}(i)} \quad (6.10)$$

$$= \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} \frac{\text{H}(x_{ij})}{\text{rank}(i)} \quad (6.11)$$

$$= \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} L_{ij}, \quad (6.12)$$

where the resulting primary term turns out to be,

$$L_{ij} = \frac{\text{H}(x_{ij})}{\text{rank}(i)}. \quad (6.13)$$

Note that when $i \notin \mathcal{P}$ or $j \notin \mathcal{N}$, no error needs to be added to $\mathcal{L}_{\text{AP}}$ due to the indices of the summations (Eq. 6.12). This is imposed by an additional variable, α_{ij} ², and then AP Loss can be expressed in the desired form, that is the normalized summation over all pairs, as follows:

$$\mathcal{L}_{\text{AP}} = \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P} \cup \mathcal{N}} \sum_{j \in \mathcal{P} \cup \mathcal{N}} L_{ij} \alpha_{ij}. \quad (6.14)$$

To provide a more general overview, we can summarize the computation of AP Loss during training (i.e. the forward pass) in three steps (Fig. 6.1 green arrows): (1) Given model outputs, the difference transforms are obtained; (2) given difference transforms the primary terms are obtained and (3) given primary terms the loss value is obtained.

Optimisation of AP Loss (Backward Pass): Here, the aim is to find updates $\frac{\partial \mathcal{L}}{\partial \hat{s}_i}$, and then proceed with backpropagation through model parameters. Among the three computation steps, Step 1 and Step 3 are differentiable, whereas a primary term L_{ij} is not a differentiable function of difference transforms (Fig. 6.1 orange arrows).

² this is denoted by y_{ij} by Chen et al. [37]

Denoting this update in x_{ij} by Δx_{ij} and using the chain rule, $\frac{\partial \mathcal{L}}{\partial \hat{s}_i}$ can be expressed as:

$$\frac{\partial \mathcal{L}}{\partial \hat{s}_i} = \sum_{j,k} \frac{\partial \mathcal{L}}{\partial L_{jk}} \Delta x_{jk} \frac{\partial x_{jk}}{\partial \hat{s}_i} = \frac{1}{Z} \left(\sum_{j \in \mathcal{P} \cup \mathcal{N}} \Delta x_{ji} - \sum_{j \in \mathcal{P} \cup \mathcal{N}} \Delta x_{ij} \right). \quad (6.15)$$

Inspired by error-driven update [44]³, Chen et al. [37] showed that the following update can be used to optimise AP Loss,

$$\Delta x_{ij} = -(L_{ij}^* \alpha_{ij} - L_{ij} \alpha_{ij}), \quad (6.16)$$

where L_{ij}^* is the target primary term indicating the desired/target error for pair (i, j) . Note that the target primary term of AP Loss is always $L_{ij}^* = 0$ since there is no error when a positive is ranked above negative for AP Loss.

6.2 Identity Update to Compute and Optimise Ranking-based Loss Functions

While the formulation to compute and optimise AP Loss enabled a ranking-based loss function to achieve similar performance with its score-based counterpart (i.e. Focal Loss) for the first time; it can not be generalized over different ranking-based loss functions easily. In particular, it has the following drawbacks: (D1) The resulting loss value ($\mathcal{L}$) does not consider the target L_{ij}^* , and thus, is not easily interpretable when $L_{ij}^* \neq 0$ (cf. our aLRP Loss [156] and RS Loss [165]). These errors become especially important with continuous labels as in our RS Loss: The larger the label of $i \in \mathcal{P}$, the larger should $\hat{s}_i$ be, and (D2) identifying the primary terms is especially a challenge when there are intra-class errors (e.g. the sorting error of our RS Loss [165]) and also requires a careful derivation (Eq. 6.7-6.14), which may also requires additional parameter (e.g. α_{ij} in Eq. 6.14). Both (D1) and (D2) imply that the optimisation method is specifically designed for AP Loss and its generalization over different ranking-based losses is not straightforward. Having observed these limitations, we address (D1) and (D2) by our Identity Update, a general and simple framework to compute and optimise ranking-based loss functions.

Definition of the Ranking-based Loss Function. We define a ranking-based loss

³ Note that the perceptron update rule is $w^{new} = w^{old} + (e^* - e)x$ where e is the error, e^* is the target error and x is the input. Then assuming the input, $x = 1$, and reorganising the update rule to align it with that of SGD update, we have $w^{new} = w^{old} - (-(e - e^*))$ and $-(e - e^*)$ can be exploited as the update. Also see the work of Chen et al. [37].

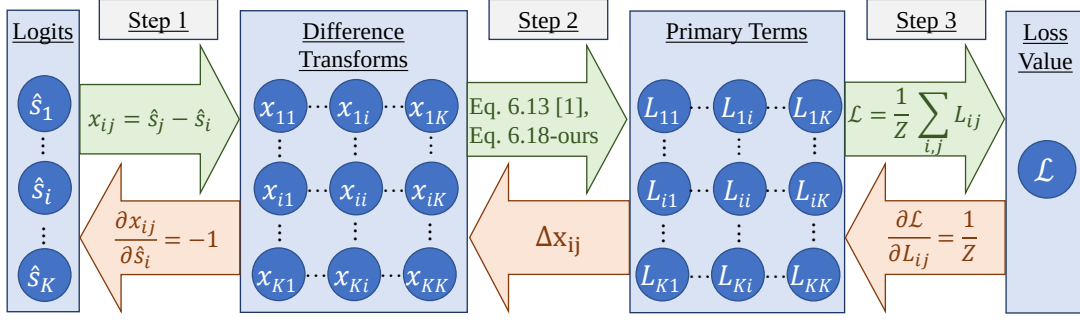


Figure 6.1: Three-step computation (green arrows) and optimisation (orange arrows) algorithms of ranking-based loss functions. Our identity update (i) yields interpretable loss values (see Appendix M for an example on our RS Loss) since target primary term is guaranteed to be 0, thus the primary terms corresponds to the error between current value and target value (see our loss definition in Eq. 6.17, and the resulting primary term definition in Eq. 6.18), (ii) introduces a definition of primary terms (Eq. 6.18—green arrow in Step 2), instead of cumbersome derivation (e.g. the derivation of the primary terms of AP Loss [37] is presented in Eq. 6.7-6.14). Furthermore, our primary term definition does not require additional parameters such as α_{ij} in Eq. 6.14, and also allows intra-class errors, crucial to model our RS Loss, and (iii) results in a simple “Identity Update” rule (orange arrow in Step 2): $\Delta x_{ij} = L_{ij}$. To sum up, “Identity Update” only requires identifying primary terms, which are also explicitly defined (Eq. 6.18), both to compute and to optimise a ranking-based loss function.

function as:

$$\mathcal{L} = \frac{1}{Z} \sum_{i \in \mathcal{P} \cup \mathcal{N}} (\ell(i) - \ell^*(i)), \quad (6.17)$$

where Z is a problem specific normalization constant, and $\ell(i)$ and $\ell^*(i)$ are the error term and desired error term computed on i respectively. Our loss definition has two benefits: (i) $\mathcal{L}$ directly measures the difference between the target and the desired errors, yielding an interpretable loss value to address (D1), and (ii) we do not constrain $\mathcal{L}$ to be defined only on positives and replace “ $i \in \mathcal{P}$ ” with “ $i \in \mathcal{P} \cup \mathcal{N}$ ”. Although we do not use “ $i \in \mathcal{P} \cup \mathcal{N}$ ” to model our loss functions (aLRP and RS Losses), it makes the definition of $\mathcal{L}$ complete in the sense that, if necessary to obtain $\mathcal{L}$, individual errors ($\ell(i)$) can be computed on each output, and hence, a larger set of ranking-based

loss functions can be represented.

Computation of the Ranking-based Loss Function (Forward Pass). Similar to Chen et al. [37], we follow the same three-step computation algorithm, but differently our goal is to express $\mathcal{L}$ as a sum of *primary terms* in a more general form than Eq. 6.12 given a ranking-based loss function following Eq. 6.17.

Definition 2. The *primary term* L_{ij} concerning examples $i \in \mathcal{P} \cup \mathcal{N}$ and $j \in \mathcal{P} \cup \mathcal{N}$ is the loss originating from i and distributed over j via a probability mass function $p(j|i)$. Formally,

$$L_{ij} = (\ell(i) - \ell^*(i)) p(j|i). \quad (6.18)$$

Then, as desired, we can express $\mathcal{L}$ as a summation of L_{ij} s:

Theorem 1. $\mathcal{L} = \frac{1}{Z} \sum_{i \in \mathcal{P} \cup \mathcal{N}} \sum_{j \in \mathcal{P} \cup \mathcal{N}} L_{ij}.$

Proof. The definition of the loss function $\mathcal{L}$,

$$\mathcal{L} = \frac{1}{Z} \sum_{i \in \mathcal{P} \cup \mathcal{N}} (\ell(i) - \ell^*(i)), \quad (6.19)$$

can be expressed by using the fact that $p(j|i)$ is a pmf, and thus $\sum_{j \in \mathcal{P} \cup \mathcal{N}} p(j|i) = 1$ as:

$$\mathcal{L} = \frac{1}{Z} \sum_{i \in \mathcal{P} \cup \mathcal{N}} (\ell(i) - \ell^*(i)) \sum_{j \in \mathcal{P} \cup \mathcal{N}} p(j|i). \quad (6.20)$$

Reorganizing the terms and using the definition of the primary term (Eq. 6.18) conclude the proof,

$$\mathcal{L} = \frac{1}{Z} \sum_{i \in \mathcal{P} \cup \mathcal{N}} \sum_{j \in \mathcal{P} \cup \mathcal{N}} (\ell(i) - \ell^*(i)) p(j|i) \quad (6.21)$$

$$= \frac{1}{Z} \sum_{i \in \mathcal{P} \cup \mathcal{N}} \sum_{j \in \mathcal{P} \cup \mathcal{N}} L_{ij}. \quad (6.22)$$

□

Note that our primary term definition in Eq. 6.18 makes it easier to obtain primary terms addressing (D2) and free from cumbersome derivations (e.g. Eq. 6.7-6.14) and additional variables (e.g. α_{ij} in Eq. 6.14). As a result, using 6.18 in three-step algorithm (Fig. 6.1 green arrows), one can easily compute the loss value during training.

Furthermore, with this definition, we add more flexibility on the error distribution: e.g., AP Loss takes $p(j|i) = H(x_{ij})/N_{\text{FP}}(i)$, which distributes error uniformly (since it is reduced to $1/N_{\text{FP}}(i)$) over $j \in \mathcal{N}$ with $\hat{s}_j \geq \hat{s}_i$; though, a skewed $p(j|i)$ can be used to promote harder examples (i.e. larger x_{ij}). Here, $N_{\text{FP}}(i) = \sum_{j \in \mathcal{N}} H(x_{ij})$ is the number of false positives for $i \in \mathcal{P}$.

Optimisation of the Loss (Backward Pass). Since the error of a pair, L_{ij} , is minimized when $\ell(i) = \ell^*(i)$, Eq. 6.18 has a target of $L_{ij}^* = 0$ regardless of $\mathcal{L}$. Thus, Δx_{ij} in Eq. 6.15 is simply the primary term itself:

$$\Delta x_{ij} = -(L_{ij}^* - L_{ij}) = -(0 - L_{ij}) = L_{ij}. \quad (6.23)$$

Therefore, replacing Δx_{ij} by L_{ij} in Eq. 6.15,

$$\frac{\partial \mathcal{L}}{\partial \hat{s}_i} = \frac{1}{Z} \left(\sum_{j \in \mathcal{P} \cup \mathcal{N}} L_{ji} - \sum_{j \in \mathcal{P} \cup \mathcal{N}} L_{ij} \right), \quad (6.24)$$

concluding the derivation of our *Identity Update*. The steps for obtaining the gradients of $\mathcal{L}$ are summarized in Algorithm 3.

Algorithm 3 Obtaining the gradients of a ranking-based function with Identity Update.

Input: A ranking-based function $\mathcal{L} = (\ell(i), \ell^*(i), Z)$, and a probability mass function $p(j|i)$

Output: The gradient of $\mathcal{L}$ with respect to model output $\hat{s}$

- 1: $\forall i, j$ find primary term, L_{ij} , using Eq. 6.18.
 - 2: **return** $\frac{1}{Z} \left(\sum_{j \in \mathcal{P} \cup \mathcal{N}} L_{ji} - \sum_{j \in \mathcal{P} \cup \mathcal{N}} L_{ij} \right)$ for each $\hat{s}_i \in \hat{s}$ (c.f. Eq. 6.24).
-

Finally, we show that Identity Update provides balanced training for ranking-based losses conforming to Theorem 1:

Theorem 2. *Training is balanced between positive and negative examples at each iteration; i.e. the magnitude of the total gradients of positives and negatives are equal:*

$$\left| \sum_{i \in \mathcal{P}} \frac{\partial \mathcal{L}}{\partial \hat{s}_i} \right| = \left| \sum_{i \in \mathcal{N}} \frac{\partial \mathcal{L}}{\partial \hat{s}_i} \right|. \quad (6.25)$$

Proof. We first obtain the total gradients of positives and negatives, and then finally it turns out that the magnitude of the total gradients of positives has the same magnitude with that of negatives, but in opposite directions.

The total gradients of positives. By algebraically manipulating Eq. 6.24, we can obtain the following simplified form for the total gradients of positive examples.

$$\sum_{i \in \mathcal{P}} \frac{\partial \mathcal{L}}{\partial \hat{s}_i} = \sum_{i \in \mathcal{P}} \frac{1}{Z} \left(\sum_{j \in \mathcal{P} \cup \mathcal{N}} L_{ji} - \sum_{j \in \mathcal{P} \cup \mathcal{N}} L_{ij} \right) \quad (6.26)$$

$$= \frac{1}{Z} \left(\sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{P} \cup \mathcal{N}} L_{ji} - \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{P} \cup \mathcal{N}} L_{ij} \right) \quad (6.27)$$

$$= \frac{1}{Z} \left(\sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{P}} L_{ji} + \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} L_{ji} - \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{P}} L_{ij} - \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} L_{ij} \right) \quad (6.28)$$

$$= \frac{1}{Z} \left(\sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} L_{ji} - \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} L_{ij} \right) \quad (6.29)$$

The total gradients of negatives. Similar to positives, we obtain the total gradients of positives as follows:

$$\sum_{i \in \mathcal{N}} \frac{\partial \mathcal{L}}{\partial \hat{s}_i} = \sum_{i \in \mathcal{N}} \frac{1}{Z} \left(\sum_{j \in \mathcal{P} \cup \mathcal{N}} L_{ji} - \sum_{j \in \mathcal{P} \cup \mathcal{N}} L_{ij} \right) \quad (6.30)$$

$$= \frac{1}{Z} \left(\sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{P} \cup \mathcal{N}} L_{ji} - \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{P} \cup \mathcal{N}} L_{ij} \right) \quad (6.31)$$

$$= \frac{1}{Z} \left(\sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{P}} L_{ji} + \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} L_{ji} - \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{P}} L_{ij} - \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} L_{ij} \right) \quad (6.32)$$

$$= \frac{1}{Z} \left(\sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{P}} L_{ji} - \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{P}} L_{ij} \right). \quad (6.33)$$

Interchanging i by j and reorganizing summations, we have:

$$\sum_{i \in \mathcal{N}} \frac{\partial \mathcal{L}}{\partial \hat{s}_i} = \frac{1}{Z} \left(\sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} L_{ij} - \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} L_{ji} \right). \quad (6.34)$$

Concluding the proof. Considering Eq. 6.29 and Eq. 6.34, one can note that the magnitude of the total gradients of positives has the same magnitude with that of negatives, but in opposite directions:

$$\sum_{i \in \mathcal{P}} \frac{\partial \mathcal{L}}{\partial \hat{s}_i} = - \sum_{i \in \mathcal{N}} \frac{\partial \mathcal{L}}{\partial \hat{s}_i}, \quad (6.35)$$

which implies their magnitudes to be equal, concluding the proof.

$$\left| \sum_{i \in \mathcal{P}} \frac{\partial \mathcal{L}}{\partial \hat{s}_i} \right| = \left| \sum_{i \in \mathcal{N}} \frac{\partial \mathcal{L}}{\partial \hat{s}_i} \right|, \quad (6.36)$$

□

Furthermore, note that when the errors are computed only on positives (i.e. the loss function has the form of $\mathcal{L} = \frac{1}{Z} \sum_{i \in \mathcal{P}} (\ell(i) - \ell^*(i))$ as our aLRP Loss and RS Loss in this thesis, and also AP Loss), $\sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} L_{ji} = 0$ in Eq. 6.29 and Eq. 6.34. Also noting that, $L_{ij} \geq 0$ (Eq. 6.18), we can present a stricter equality than Theorem 2, indicating that the summed gradient magnitudes for positives and negatives are equal in that case as follows:

$$\sum_{i \in \mathcal{P}} \left| \frac{\partial \mathcal{L}}{\partial \hat{s}_i} \right| = \sum_{i \in \mathcal{N}} \left| \frac{\partial \mathcal{L}}{\partial \hat{s}_i} \right|. \quad (6.37)$$

6.3 A Case Study: Optimising AP Loss by Identity Update

Let us derive AP Loss as a case example for our Identity Update. To begin with the definition of the loss function (Eq. 6.17), $\ell_{\text{AP}}(i)$ is simply $1 - \text{precision}(i) = N_{\text{FP}}(i)/\text{rank}(i)$, and $Z = |\mathcal{P}|$. The target of AP Loss is to rank a positive above all negative examples, implying $\ell_{\text{AP}}(i)^* = 0$.

Then, we obtain the primary terms (Eq. 6.18) simply based on the definition of the loss function by assuming $p(j|i)$ to be uniform, i.e. $p(j|i) = H(x_{ij})/N_{\text{FP}}(i)$. These give us $L_{ij} = \frac{N_{\text{FP}}(i)}{\text{rank}(i)} \frac{H(x_{ij})}{N_{\text{FP}}(i)} = \frac{H(x_{ij})}{\text{rank}(i)}$ (c.f. L_{ij} in Eq. 6.12), which is directly used both to compute and to optimise AP Loss, thereby simply completing the derivation.

CHAPTER 7

AVERAGE LOCALISATION-RECALL-PRECISION LOSS: A RANKING-BASED, BALANCED LOSS FUNCTION UNIFYING CLASSIFICATION AND LOCALISATION IN OBJECT DETECTION

In this chapter, we present our average Localisation-Recall-Precision Loss. This chapter is based on our work,

- Kemal Oksuz, Baris Can Cam, Emre Akbas* and Sinan Kalkan*, “A Ranking-based, Balanced Loss Function Unifying Classification and Localisation in Object Detection”, Advances on Neural Information and Processing Systems (NeurIPS), 2020 (Spotlight paper).

Different from the paper, we exclude the section on background on LRP and AP Loss, which are comprehensively discussed in Chapters 5 and 6 of this thesis respectively. When needed, we provide cross-references to the relevant material in the previous chapters. Additionally, for clarity, the optimisation algorithm used to optimise aLRP Loss, “A Generalisation of Error-Driven Optimisation for Ranking-Based Losses”, is discussed in Appendix I since a more general form of optimisation ranking-based loss functions is introduced in Chapter 6 of this thesis as “Identity Update”. Note that both of our optimisation algorithms (a generalization of error-driven optimisation and Identity Update) provide the same gradient values, hence the results remain the same in this chapter regardless of the optimisation algorithm. We also demonstrate in Appendix J to define and obtain the gradients of aLRP Loss using our Identity Update as an additional use-case to AP Loss (Section 6.3) and our RS Loss (Chapter 8). Apart from these modifications, we make minor changes to fit the text appropriately in the

* Equal contribution for senior authorship.

context of this thesis. Finally, within the scope of this project, the usage of robust box regression losses with aLRP Loss (e.g. GIoU Loss [93], DIoU Loss [94]) is contributed by Baris Can Cam, and accordingly included in his MS thesis [166] (see the discussion after Eq. 7.3).

7.1 Introduction

Object detection requires jointly optimising a classification objective ($\mathcal{L}_c$) and a localisation objective ($\mathcal{L}_r$) combined conventionally with a balancing hyperparameter (w_r) as follows:

$$\mathcal{L} = \mathcal{L}_c + w_r \mathcal{L}_r. \quad (7.1)$$

Optimising $\mathcal{L}$ in this manner has three critical drawbacks: (D1) It does not correlate the two tasks, and hence, does not guarantee high-quality localisation for high-precision examples (Fig. 7.1). (D2) It requires a careful tuning of w_r [39, 40, 93], which is prohibitive since a single training may last on the order of days, and ends up with a sub-optimal constant w_r [167, 168]. (D3) It is adversely impeded by the positive-negative imbalance in $\mathcal{L}_c$ and inlier-outlier imbalance in $\mathcal{L}_r$, thus it requires sampling strategies [18, 68] or specialized loss functions [22, 32], introducing more hyperparameters (Table 1.1).

A recent solution for D3 is to directly maximize Average Precision (AP) with a loss function called AP Loss [37]. AP Loss is a ranking-based loss function to optimise the ranking of the classification outputs and provides balanced training between positives and negatives.

In this chapter, we extend AP Loss to address all three drawbacks (D1-D3) with one, unified loss function called average Localisation Recall Precision (aLRP) Loss. In analogy with the link between precision and AP Loss, we formulate aLRP Loss as the average of LRP values [107] over the positive examples on the Recall-Precision (RP) curve. aLRP has the following benefits: (i) It exploits ranking for both classification and localisation, enforcing high-precision detections to have high-quality localisation (Fig. 7.1). (ii) aLRP has a single hyperparameter (which we did not need to tune) as opposed to ~ 7 in state-of-the-art loss functions (Table 1.1). (iii) The network is

Input Anchors	Classifier Output (C)		Three Possible Localization Outputs					
			Pos. Correlated with C (R_1)		Uncorrelated with C (R_2)		Neg. Correlated with C (R_3)	
	Score	Rank	IoU	Rank	IoU	Rank	IoU	Rank
a_1	1.00	1	0.95	1	0.80	2	0.50	4
a_2	0.90	—	—	—	—	—	—	—
a_3	0.80	2	0.80	2	0.65	3	0.65	3
a_4	0.70	—	—	—	—	—	—	—
a_5	0.60	—	—	—	—	—	—	—
a_6	0.50	3	0.65	3	0.50	4	0.80	2
a_7	0.40	—	—	—	—	—	—	—
a_8	0.30	—	—	—	—	—	—	—
a_9	0.20	—	—	—	—	—	—	—
a_{10}	0.10	4	0.50	4	0.95	1	0.95	1

(a) 3 possible localization outputs (R_1 - R_3) for the same classifier output (C)
(Orange: Positive anchors, Gray: Negative anchors)

Detector Output	AP ₅₀	AP ₆₅	AP ₈₀	AP ₉₅	AP
(C & R_1)	0.51	0.43	0.33	0.20	0.37
(C & R_2)	0.51	0.39	0.24	0.02	0.29
(C & R_3)	0.51	0.19	0.08	0.02	0.20

(b) Performance in AP = (AP₅₀+AP₆₅+AP₈₀+AP₉₅)/4

	$\mathcal{L}_c$		$\mathcal{L}_r$		Ours
Detector Output	Cross Entropy	AP Loss	L1 Loss	IoU Loss	aLRP Loss
(C & R_1)	0.87	0.36	0.29	0.28	0.53
(C & R_2)	0.87	0.36	0.29	0.28	0.69
(C & R_3)	0.87	0.36	0.29	0.28	0.89

(c) Comparison of different loss functions
(Red: Improper ordering, Green: Proper ordering)

Figure 7.1: aLRP Loss enforces high-precision detections to have high-IoUs, while others do not. (a) Classification and three possible localisation outputs for 10 anchors and the rankings of the positive anchors with respect to (wrt) the scores (for C) and IoUs (for R_1 , R_2 and R_3). Since the regressor is only trained by positive anchors, “—” is assigned for negative anchors. (b,c) Performance and loss assignment comparison of R_1 , R_2 and R_3 when combined with C . When correlation between the rankings of classifier and regressor outputs decreases, performance degrades up to 17 AP (b). While any combination of $\mathcal{L}_c$ and $\mathcal{L}_r$ cannot distinguish them, aLRP Loss penalizes the outputs accordingly (c). The details of the calculations are presented in the Section 1 of Supp.Mat. in our paper [156] and excluded in this thesis.

trained by a single loss function that provides provable balance between positives and negatives. Replacing AP and SmoothL1 losses by aLRP Loss for training RetinaNet improves the performance by up to 5.4AP, and our best model reaches 48.9AP without test time augmentation, outperforming all existing one-stage detectors with significant margin.

7.2 Related Work

Balancing $\mathcal{L}_c$ and $\mathcal{L}_r$ in Eq. 7.1, an open problem in object detection (OD) [142], bears important challenges: Disposing w_r , and correlating $\mathcal{L}_c$ and $\mathcal{L}_r$. *Classification-aware regression loss* [33] links the branches by weighing $\mathcal{L}_r$ of an anchor using its

classification score. Following Kendall et al. [168], *LapNet* [167] tackled the challenge by making w_r a learnable parameter based on homoscedastic uncertainty of the tasks. Other approaches [92, 130] combine the outputs of two branches during non-maximum suppression (NMS) at inference. Unlike these methods, aLRP Loss considers the ranking wrt scores for both branches and addresses the imbalance problem naturally.

Ranking-based objectives in OD: An inspiring solution for balancing classes is to optimise a ranking-based objective. However, such objectives are discrete wrt the scores, rendering their direct incorporation challenging. A solution is to use black-box solvers for an interpolated AP loss surface [169], which, however, provided only little gain in performance. AP Loss [37] takes a different approach by using an error-driven update mechanism to calculate gradients (Sec. 6.2). An alternative, DR Loss [70], employs Hinge Loss to enforce a margin between the scores of the positives and negatives. Despite promising results, these methods are limited to classification and leave localisation as it is. In contrast, we propose a single, balanced, ranking-based loss to train both branches.

7.3 Average Localisation-Recall-Precision (aLRP) Loss

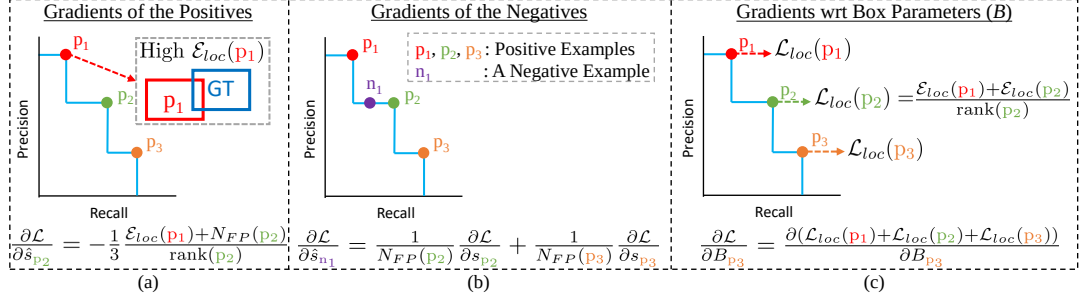
Similar to the relation between precision and AP Loss, aLRP Loss is defined as the average of LRP values ($\ell_{\text{LRP}}(i)$) of positive examples:

$$\mathcal{L}^{\text{aLRP}} := \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \ell_{\text{LRP}}(i). \quad (7.2)$$

For LRP, we assume that anchors are dense enough to cover all ground-truths, i.e. $N_{\text{FN}} = 0$. Also, since a detection is enforced to follow the label of its anchor during training, TP and FP sets are replaced by the thresholded subsets of $\mathcal{P}$ and $\mathcal{N}$, respectively. This is applied by $H(\cdot)$, and $\text{rank}(i) = N_{\text{TP}} + N_{\text{FP}}$ from Eq. 5.2. Then, following the definitions in Sec. 6.1, $\ell_{\text{LRP}}(i)$ is:

$$\ell_{\text{LRP}}(i) = \frac{1}{\text{rank}(i)} \left(N_{\text{FP}}(i) + \mathcal{E}_{\text{loc}}(i) + \sum_{k \in \mathcal{P}, k \neq i} \mathcal{E}_{\text{loc}}(k) H(x_{ik}) \right), \quad (7.3)$$

where $\mathcal{E}_{\text{loc}}(i) = \frac{1 - \text{IoU}(i)}{1 - \tau}$ and τ is the positive-negative assignment threshold following the definition of LRP. Note that Eq. 7.3 allows using robust forms of IoU-based losses



(e.g. generalized IoU (GIoU) [93]) only by replacing IoU Loss (i.e. $1 - \text{IoU}(i)$) in $\mathcal{E}_{loc}(i)$ and normalizing the range to $[0, 1]$ [166].

In order to provide more insight and facilitate gradient derivation, we split Eq. 7.2 into two as localisation and classification components such that $\mathcal{L}^{\text{aLRP}} = \mathcal{L}_{cls}^{\text{aLRP}} + \mathcal{L}_{loc}^{\text{aLRP}}$, where

$$\mathcal{L}_{cls}^{\text{aLRP}} = \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \frac{N_{FP}(i)}{\text{rank}(i)}, \text{ and} \quad (7.4)$$

$$\mathcal{L}_{loc}^{\text{aLRP}} = \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \frac{1}{\text{rank}(i)} \left(\mathcal{E}_{loc}(i) + \sum_{k \in \mathcal{P}, k \neq i} \mathcal{E}_{loc}(k)H(x_{ik}) \right). \quad (7.5)$$

7.3.1 Optimisation of the aLRP Loss

$\mathcal{L}^{\text{aLRP}}$ is differentiable wrt the estimated box parameters, B , since $\mathcal{E}_{loc}$ is differentiable [91, 93] (i.e. the derivatives of $\mathcal{L}_{cls}^{\text{aLRP}}$ and $\text{rank}(\cdot)$ wrt B are 0). However,

$\mathcal{L}_{cls}^{aLRP}$ and $\mathcal{L}_{loc}^{aLRP}$ are not differentiable wrt the classification scores, and therefore, we use error-driven update similar to AP Loss¹.

Using the same error distribution, $p(j|i)$, from AP Loss, the primary terms of aLRP Loss can be defined as $L_{ij}^{aLRP} = \ell^{LRP}(i)p(j|i)$. As for the target primary terms, we use the following desired LRP Error:

$$\ell^{LRP}(i)^* = \frac{1}{\text{rank}(i)} \left(\cancel{N_{FP}(i)}^0 + \mathcal{E}_{loc}(i) + \sum_{k \in \mathcal{P}, k \neq i} \mathcal{E}_{loc}(k) \cancel{H(x_{ik})}^0 \right)^0 = \frac{\mathcal{E}_{loc}(i)}{\text{rank}(i)}, \quad (7.6)$$

yielding a target primary term, $L_{ij}^{aLRP*} = \ell^{LRP}(i)^* p(j|i)$, which includes localisation error and can be non-zero when $\hat{s}_i < \hat{s}_j$, unlike AP Loss. Then, the resulting update for x_{ij} is (Eq. 6.15)²:

$$\Delta x_{ij} = - \left(\ell^{LRP}(i)^* - \ell^{LRP}(i) \right) p(j|i) = \frac{1}{\text{rank}(i)} \left(N_{FP}(i) + \sum_{k \in \mathcal{P}, k \neq i} \mathcal{E}_{loc}(k) H(x_{ik}) \right) \frac{H(x_{ij})}{N_{FP}(i)}. \quad (7.7)$$

Finally, $\partial \mathcal{L}^{aLRP} / \partial \hat{s}_i$ can be obtained with Eq. 6.15. Computation and optimisation of aLRP Loss has the same quadratic time&space complexity with those of AP Loss. We provide further details of aLRP Loss in Appendix K.

Interpretation of the Components: A distinctive property of aLRP Loss is that classification and localisation errors are handled in a unified manner: i.e. with aLRP, both classification and localisation branches use the entire output of the detector, instead of working in their separate domains as conventionally done. As shown in Fig. 7.2(a,b), $\mathcal{L}_{cls}^{aLRP}$ takes into account localisation errors of detections with larger scores ($\hat{s}$) and promotes the detections with larger IoUs to have higher $\hat{s}$, or suppresses the detections with high- $\hat{s}$ &low-IoU. Similarly, $\mathcal{L}_{loc}^{aLRP}$ inherently weighs each positive based on its classification rank (see Appendix K for the weights): the contribution of a positive increases if it has a larger $\hat{s}$. To illustrate, in Fig. 7.2(c), while $\mathcal{E}_{loc}(p_1)$ (i.e. p_1 is the positive with the largest $\hat{s}$) contributes to each $\mathcal{L}_{loc}(i)$; $\mathcal{E}_{loc}(p_3)$ (i.e. p_3 is the positive with the largest $\hat{s}$) only contributes once with a very low weight due to

¹ In this chapter, we use ‘A Generalisation of Error-Driven Optimisation for Ranking-Based Losses’ (Appendix I) to obtain the gradients of aLRP Loss as in our original paper and demonstrate in Appendix J how to define and obtain the gradients of aLRP Loss using our Identity Update.

² Note that different from our paper, in this thesis we use Δx_{ij} to denote directly the update not the error-driven update.

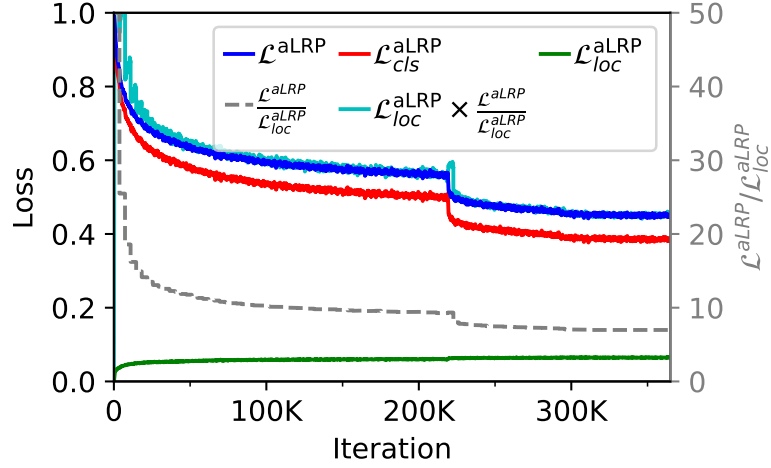


Figure 7.3: aLRP Loss and its components. The localisation component is self-balanced.

its rank normalizing $\mathcal{L}_{loc}(p_3)$. Hence, the localisation branch effectively focuses on detections ranked higher wrt $\hat{s}$.

7.3.2 A Self-Balancing Extension for the Localisation Task

LRP metric yields localisation error only if a detection is classified correctly (Sec. 5.5.1, Eq. 5.2). Hence, when the classification performance is poor (e.g. especially at the beginning of training), aLRP Loss is dominated by classification error ($N_{FP}(i)/\text{rank}(i) \approx 1$ and $\ell^{LRP}(i) \in [0, 1]$ in Eq. 7.3). As a result, the localisation head is hardly trained at the beginning (Fig. 7.3). Moreover, Fig. 7.3 also shows that $\mathcal{L}_{cls}^{aLRP}/\mathcal{L}_{loc}^{aLRP}$ varies significantly throughout training. To alleviate this, we propose a simple and dynamic *self-balancing* (SB) strategy using gradient magnitudes: note that $\sum_{i \in \mathcal{P}} |\partial \mathcal{L}^{aLRP} / \partial \hat{s}_i| = \sum_{i \in \mathcal{N}} |\partial \mathcal{L}^{aLRP} / \partial \hat{s}_i| \approx \mathcal{L}^{aLRP}$ (Appendix K). Then, assuming that the gradients wrt scores and boxes are proportional to their contributions to the aLRP Loss, we multiply $\partial \mathcal{L}^{aLRP} / \partial B$ by the average $\mathcal{L}^{aLRP} / \mathcal{L}_{loc}^{aLRP}$ of the previous epoch.

7.4 Experiments

Dataset: We train all our models on COCO *trainval35K* set [13] (115K images), test on *minival* set (5k images) and compare with the state-of-the-art (SOTA) on *test-dev* set (20K images).

Performance Measures: COCO-style AP [13], denoted by AP^C , and when possible optimal LRP [107] (Sec. 5.5.1) are used for comparison. For more insight into aLRP Loss, we use Pearson correlation coefficient (ρ) to measure correlation between the rankings of classification and localisation, averaged over classes.

Implementation Details: For training, we use 4 v100 GPUs. The batch size is 32 for training with 512×512 images (aLRPLoss500), whereas it is 16 for 800×800 images (aLRPLoss800). Following AP Loss, our models are trained for 100 epochs using stochastic gradient descent with a momentum factor of 0.9. We use a learning rate of 0.008 for aLRPLoss500 and 0.004 for aLRPLoss800, each decreased by factor 0.1 at epochs 60 and 80. Similar to previous work [37, 40], standard data augmentation methods from SSD [24] are used. At test time, we rescale shorter sides of images to 500 (aLRPLoss500) or 800 (aLRPLoss800) pixels by ensuring that the longer side does not exceed $1.66\times$ of the shorter side. NMS is applied to 1000 top-scoring detections using 0.50 as IoU threshold.

7.4.1 Ablation Study

In this section, in order to provide a fair comparison, we build upon the official implementation of our baseline, AP Loss [170]. Keeping all design choices fixed, otherwise stated, we just replace AP & Smooth L1 losses by aLRP Loss to optimise RetinaNet [18]. We conduct ablation analysis using aLRPLoss500 on ResNet-50 backbone (more ablation experiments are presented in Appendix L).

Effect of using ranking for localisation: Table 7.1 shows that using a ranking loss for localisation improves AP^C (from 35.5 to 36.9). For better insight, AP_{90} is also included in Table 7.1, which shows ~ 5 points increase despite similar AP_{50} values. This confirms that aLRP Loss does produce high-quality outputs for both branches,

Table 7.1: Ablation analysis on COCO *minival*. For optimal LRP (oLRP), lower is better. AP^C denotes COCO-style AP.

Method	Rank-Based $\mathcal{L}_c$	Rank-Based $\mathcal{L}_r$	SB	ATSS	AP^C	AP_{50}	AP_{75}	AP_{90}	oLRP	ρ
AP Loss [37]	✓				35.5	58.0	37.0	9.0	71.0	0.45
aLRP Loss	✓	✓ (w IoU)			36.9	57.7	38.4	13.9	69.9	0.49
	✓	✓ (w IoU)	✓		38.7	58.1	40.6	17.4	68.5	0.48
	✓	✓ (w GIoU)	✓		38.9	58.5	40.5	17.4	68.4	0.48
	✓	✓ (w GIoU)	✓	✓	40.2	60.3	42.3	18.1	67.3	0.48

and boosts the performance for larger IoUs.

Effect of Self-Balancing (SB): Section 7.3.2 and Fig. 7.3 discussed how $\mathcal{L}_{cls}^{aLRP}$ and $\mathcal{L}_{loc}^{aLRP}$ behave during training and introduced self-balancing to improve training of the localisation branch. Table 7.1 shows that SB provides +1.8 AP^C gain, similar AP_{50} and +8.4 points in AP_{90} against AP Loss. Comparing SB with constant weighting in Table 7.2, our SB approach provides slightly better performance than constant weighting, which requires extensive tuning and end up with different w_r constants for IoU and GIoU. Finally, Table 7.3 presents that initialization of SB (i.e. its value for the first epoch) has a negligible effect on the performance even with very large values. We use 50 for initialization.

Using GIoU: Table 7.1 suggests robust IoU-based regression (GIoU) improves performance slightly.

Using ATSS: Finally, we replace the standard IoU-based assignment by ATSS [42], which uses less anchors and decreases training time notably for aLRP Loss: One iteration drops from 0.80s to 0.53s with ATSS (34% more efficient with ATSS) – this time is 0.71s and 0.28s for AP Loss and Focal Loss respectively. With ATSS, we also observe +1.3 AP^C improvement (Table 7.1). See Appendix L for details.

Hence, we use GIoU [93] as part of aLRP Loss, and employ ATSS [42] when training RetinaNet.

Table 7.2: SB does not require tuning and slightly outperforms constant weighting for both IoU types. COCO-style AP, AP^C , is reported.

w_r	1	2	5	10	15	20	25	SB
w IoU	36.9	37.8	38.5	38.6	38.3	37.1	36.0	38.7
w GIoU	36.0	37.0	37.9	38.7	38.8	38.7	38.8	38.9

Table 7.3: SB is not affected significantly by the initial weight in the first epoch (w_r) even for large values.

w_r	1	50	100	500
AP^C	38.8	38.9	38.7	38.5

7.4.2 More insight on aLRP Loss

Potential of Correlating Classification and Localisation. We analyse two bounds:

(i) A *Lower Bound* where localisation provides an inverse ranking compared to classification. (ii) An *Upper Bound* where localisation provides exactly the same ranking as classification. Table 7.4 shows that correlating ranking can have a significant effect (up to 20 AP^C) on the performance especially for larger IoUs. Therefore, correlating rankings promises significant improvement (up to $\sim 10AP^C$). Moreover, while ρ is 0.44 and 0.45 for Focal Loss (results not provided in the table) and AP Loss (Table 7.1), respectively, aLRP Loss yields higher correlation (0.48, 0.49).

Analysing Balance Between Positives and Negatives. For this analysis, we compare Cross Entropy Loss (CE), Focal Loss (FL) and aLRP Loss on RetinaNet trained for 12 epochs and average results over 10 runs. Fig. 7.4 experimentally confirms Theorem 2 for aLRP Loss ($\mathcal{L}_{cls}^{aLRP}$), as it exhibits perfect balance between the gradi-

Table 7.4: Effect of correlating rankings. AP^C denotes COCO-style AP.

$\mathcal{L}$	ρ	AP^C	AP_{50}	AP_{75}	AP_{90}
aLRP Loss	0.48	38.7	58.1	40.6	17.4
Lower Bound	-1.00	28.6	58.1	23.6	5.6
Upper Bound	1.00	48.1	58.1	51.9	33.9

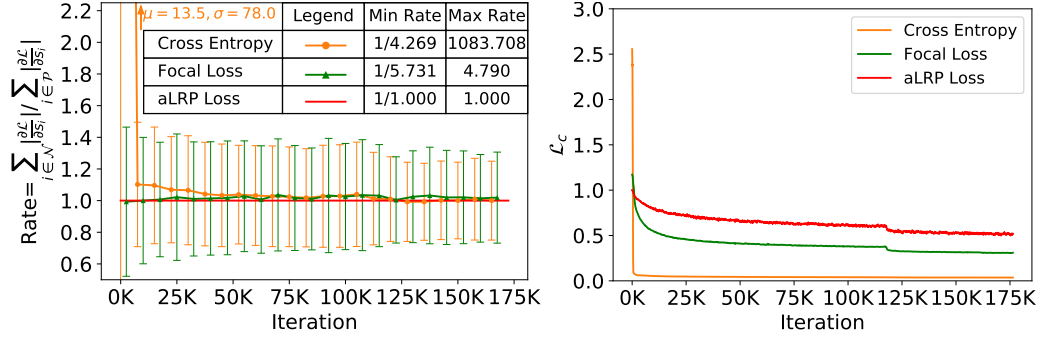


Figure 7.4: **(left)** The rate of the total gradient magnitudes of negatives to positives. **(right)** Loss values.

ents throughout training. However, we see large fluctuations in derivatives of CE and FL (left), which biases training towards positives or negatives alternately across iterations. As expected, imbalance impacts CE more as it quickly drops (right), overfitting in favor of negatives since it is dominated by the error and gradients of these large amount of negatives.

7.4.3 Comparison with State of the Art (SOTA)

Different from the ablation analysis, we find it useful to decrease the learning rate of aLRPLoss500 at epochs 75 and 95. For SOTA comparison, we use the mmdetection framework [100] for efficiency (we reproduced Table 7.1 using our mmdetection implementation, yielding similar results - see our repository). Table 7.5 presents the results, which are discussed below:

Ranking-based Losses. aLRP Loss yields significant gains over other ranking-based solutions: e.g., compared with AP Loss, aLRP Loss provides +5.4AP^C for scale 500 and +5.1AP^C for scale 800. Similarly, for scale 800, aLRP Loss performs 4.7AP^C better than DR Loss with ResNeXt-101.

Methods combining branches. Although a direct comparison is not fair since different conditions are used, we observe a significant margin (around 3-5AP^C in scale 800) compared to other approaches that combine localisation and classification.

Comparison on scale 500. We see that, even with ResNet-101, aLRPLoss500 out-

performs all other methods with 500 test scale. With ResNext-101, aLRP Loss outperforms its closest counterpart (HSD) by $2.7AP^C$ and also in all sizes (AP_S-AP_L).

Comparison on scale 800. For 800 scale, aLRP Loss achieves $45.9AP^C$ and $47.8AP^C$ on ResNet-101 and ResNeXt-101 backbones respectively. Also in this scale, aLRP Loss consistently outperforms its closest counterparts (i.e. FreeAnchor and Center-Net) by $2.9AP^C$ and reaches the highest results wrt all performance measures. With DCN [171], aLRP Loss reaches $48.9AP^C$, outperforming ATSS by $1.2AP^C$.

7.4.4 Using aLRP Loss with Different Object Detectors

Here, we use aLRP Loss to train FoveaBox [174] as an anchor-free detector, and Faster R-CNN [19] as a two-stage detector. All models use 500 scale setting, have a ResNet-50 backbone and follow our mmdetection implementation [100]. Further implementation details are presented in Appendix L.

Results on FoveaBox: To train FoveaBox, we keep the learning rate same with RetinaNet (i.e. 0.008) and only replace the loss function by aLRP Loss. Table 7.6 shows that aLRP Loss outperforms Focal Loss and AP Loss, each combined by Smooth L1 (SL1 in Table 7.6), by 1.4 and $3.2AP^C$ points (and similar oLRP points) respectively. Note that aLRP Loss also simplifies tuning hyperparameters of Focal Loss, which are set in FoveaBox to different values from RetinaNet. One training iteration of Focal Loss, AP Loss and aLRP Loss take 0.34, 0.47 and 0.54 sec respectively.

Results on Faster R-CNN: To train Faster R-CNN, we remove sampling, use aLRP Loss to train both stages (i.e. RPN and Fast R-CNN) and reweigh aLRP Loss of RPN by 0.20. Thus, the number of hyperparameters is reduced from nine (Table 1.1) to three (two δ s for step function, and a weight for RPN). We validated the learning rate of aLRP Loss as 0.012, and train baseline Faster R-CNN by both L1 Loss and GIoU Loss for fair comparison. aLRP Loss outperforms these baselines by more than $2.5AP^C$ and 2oLRP points while simplifying the training pipeline (Table 7.7). One training iteration of Cross Entropy Loss (with L1) and aLRP Loss take 0.38 and 0.85 sec respectively.

Table 7.5: Comparison with the SOTA detectors on COCO *test-dev*. $S, \times 1.66$ implies that the image is rescaled such that its longer side cannot exceed $1.66 \times S$ where S is the size of the shorter side. R:ResNet, X:ResNeXt, H:HourglassNet, D:DarkNet, De:DeNet. We use ResNeXt101 64x4d. AP^C denotes COCO-style AP.

Method	Backbone	Training Size	Test Size	AP^C	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
<i>One-Stage Methods</i>									
RefineDet [66] [‡]	R-101	512×512	512×512	36.4	57.5	39.5	16.6	39.9	51.4
EFGRNet [67] [‡]	R-101	512×512	512×512	39.0	58.8	42.3	17.8	43.6	54.5
ExtremeNet [172] ^{*‡}	H-104	511×511	original	40.2	55.5	43.2	20.4	43.2	53.1
RetinaNet [18]	X-101	$800, \times 1.66$	$800, \times 1.66$	40.8	61.1	44.1	24.1	44.2	51.2
HSD [95] [‡]	X-101	512×512	512×512	41.9	61.1	46.2	21.8	46.6	57.0
FCOS [160] [†]	X-101	$(640, 800), \times 1.66$	$800, \times 1.66$	44.7	64.1	48.4	27.6	47.5	55.6
CenterNet [40] ^{*‡}	H-104	511×511	original	44.9	62.4	48.1	25.6	47.4	57.4
ATSS [42] [†]	X-101-DCN	$(640, 800), \times 1.66$	$800, \times 1.66$	47.7	66.5	51.9	29.7	50.8	59.4
<i>Ranking Losses</i>									
AP Loss500 [37] [‡]	R-101	512×512	$500, \times 1.66$	37.4	58.6	40.5	17.3	40.8	51.9
AP Loss800 [37] [‡]	R-101	800×800	$800, \times 1.66$	40.8	63.7	43.7	25.4	43.9	50.6
DR Loss [70] [†]	X-101	$(640, 800), \times 1.66$	$800, \times 1.66$	43.1	62.8	46.4	25.6	46.2	54.0
<i>Combining Branches</i>									
LapNet [167]	D-53	512×512	512×512	37.6	55.5	40.4	17.6	40.5	49.9
Fitness NMS [92]	De-101	$512, \times 1.66$	$768, \times 1.66$	39.5	58.0	42.6	18.9	43.5	54.1
Retina+PISA [33]	R-101	$800, \times 1.66$	$800, \times 1.66$	40.8	60.5	44.2	23.0	44.2	51.4
FreeAnchor [39] [†]	X-101	$(640, 800), \times 1.66$	$800, \times 1.66$	44.9	64.3	48.5	26.8	48.3	55.9
<i>Ours</i>									
aLRP Loss500 [‡]	R-50	512×512	$500, \times 1.66$	41.3	61.5	43.7	21.9	44.2	54.0
aLRP Loss500 [‡]	R-101	512×512	$500, \times 1.66$	42.8	62.9	45.5	22.4	46.2	56.8
aLRP Loss500 [‡]	X-101	512×512	$500, \times 1.66$	44.6	65.0	47.5	24.6	48.1	58.3
aLRP Loss800 [‡]	R-101	800×800	$800, \times 1.66$	45.9	66.4	49.1	28.5	48.9	56.7
aLRP Loss800 [‡]	X-101	800×800	$800, \times 1.66$	47.8	68.4	51.1	30.2	50.8	59.1
aLRP Loss800 [‡]	X-101-DCN	800×800	$800, \times 1.66$	48.9	69.3	52.5	30.8	51.5	62.1
<i>Multi-Scale Test</i>									
aLRP Loss800 [‡]	X-101-DCN	800×800	$800, \times 1.66$	50.2	70.3	53.9	32.0	53.1	63.0

[†]: multiscale training, [‡]: SSD-like augmentation, *: Soft NMS [173] and flip augmentation at test time

Table 7.6: Comparison on FoveaBox [174]. AP^C denotes COCO-style AP.

$\mathcal{L}$	AP^C	AP_{50}	AP_{75}	AP_{90}	oLRP
Focal Loss+SL1	38.3	57.8	40.7	15.7	68.8
AP Loss+SL1	36.5	58.3	38.2	11.3	69.8
aLRP Loss (Ours)	39.7	58.8	41.5	18.2	67.2

Table 7.7: Comparison on Faster R-CNN [19]. AP^C denotes COCO-style AP.

$\mathcal{L}$	AP^C	AP_{50}	AP_{75}	AP_{90}	oLRP
Cross Entropy+L1	37.8	58.1	41.0	12.2	69.3
Cross Entropy+GIoU	38.2	58.2	41.3	13.7	69.0
aLRP Loss (Ours)	40.7	60.7	43.3	18.0	66.7

7.5 Conclusion

In this chapter, we introduced aLRP Loss, a ranking-based, balanced loss function which handles the classification and localisation errors in a unified manner. aLRP Loss has only one hyperparameter which we did not need to tune, as opposed to around 7 in SOTA loss functions. We showed that using aLRP improves its baselines significantly over different detectors by simplifying parameter tuning, and outperforms all one-stage detectors.

CHAPTER 8

RANK & SORT LOSS FOR OBJECT DETECTION AND INSTANCE SEGMENTATION

In this chapter, we present our RS Loss based on our work,

- Kemal Oksuz, Baris Can Cam, Emre Akbas* and Sinan Kalkan*, “Rank & Sort Loss for Object Detection and Instance Segmentation”, under review for International Conference on Computer Vision (ICCV), 2021.

We note that an extended version of our Identity Update, originally introduced in this work [165], is discussed in Chapter 6, and hence, this chapter is only reserved to Rank & Sort Loss. Similar to other chapters, we make minor changes to fit the text appropriately in the context of this thesis and to provide a consistent notation.

8.1 Introduction

Owing to their multi-task (e.g. classification, box regression, mask prediction) nature, object detection and instance segmentation methods rely on loss functions of the form:

$$\mathcal{L}_{VD} = \sum_{k \in \mathcal{K}} \sum_{t \in \mathcal{T}} \lambda_t^k \mathcal{L}_t^k, \quad (8.1)$$

which combines $\mathcal{L}_t^k$, the loss function for task t on stage k (e.g. $|\mathcal{K}| = 2$ for Faster R-CNN [19] with RPN and R-CNN), weighted by a hyper-parameter λ_t^k . In such formulations, the number of hyper-parameters can easily exceed 10 [156], with additional hyper-parameters arising from task-specific imbalance problems [142], e.g. the

* Equal contribution for senior authorship.

yielded impressive performances, they require longer training and more augmentation.

Broadly speaking, the ranking-based losses (AP Loss and aLRP Loss) focus on ranking positive examples over negatives, and they do not explicitly model positive-to-positive interactions. However, prioritizing predictions wrt. their localisation qualities by using an auxiliary (e.g. IoU, centerness) head has been a common approach that improves performance [160, 42, 43, 90, 130]. Besides, as recently shown by Li et al. [175] (in Quality Focal Loss - QFL), when the classifier is directly supervised to regress IoUs of the predictions, one can remove the auxiliary head and further improve the performance.

In this chapter, we propose Rank & Sort (RS) Loss as a ranking-based loss function to train visual detection (VD – i.e. object detection and instance segmentation) methods. RS Loss not only ranks positives above negatives (Fig. 8.1(a)) but also sorts positives among themselves with respect to their continuous IoU values (Fig. 8.1(b)). This approach brings in several crucial benefits. Due to the prioritization of positives during training, detectors trained with RS Loss do not need an auxiliary head, and due to its ranking-based nature, RS Loss can handle extremely imbalanced data (e.g. object detection [142]) without any sampling heuristics. Besides, except for the learning rate, RS Loss does not need any hyper-parameter tuning thanks to our tuning-free task-balancing coefficients. Owing to this significant simplification of training, we can apply RS Loss to different methods (i.e. multi-stage, one-stage, anchor-based, anchor-free) easily (i.e. *only by tuning the learning rate*) and demonstrate that RS Loss consistently outperforms baselines.

Our contributions in this chapter can be summarized as follows:

- (1) We propose *Rank & Sort Loss* that defines a ranking objective between positives and negatives as well as a sorting objective to prioritize positives wrt. their continuous IoUs. Due to this ranking-based nature, RS Loss can train models in the presence of highly imbalanced data.
- (2) We present the effectiveness of RS Loss on a diverse set of four object detectors and three instance segmentation methods only by tuning the learning rate and without

any auxiliary heads or sampling heuristics: E.g. (i) Our RS-R-CNN improves Faster-CNN by ~ 3 box AP, (ii) our RS-Mask R-CNN improves Mask R-CNN by ~ 2 mask AP and box AP, (iii) our RS-YOLACT improves YOLACT by more than 3 box AP and ~ 1.5 mask AP.

8.2 Related Work

Auxiliary heads and continuous labels. Predicting the localisation quality of a detection with an auxiliary centerness [160, 42], IoU [43, 130], mask-IoU [41] or uncertainty (i.e. variance) head [90] and combining these predictions with the classification scores for NMS are shown to improve detection performance. Lin et al. [175] discovered that using continuous IoUs of predictions to supervise the classifier outperforms using an auxiliary head. Currently, Lin et al.’s “Quality Focal Loss” [175] is the only method that is robust to class imbalance [142] and uses continuous labels to train the classifier. In this work, we investigate the generalizability of this idea on different networks (e.g. multi-stage networks [19, 20]) and on a different task (i.e. instance segmentation) by using our ranking-based RS Loss.

Ranking-based losses in VD. Despite their advantages, ranking-based losses are non-differentiable and difficult to optimise. To address this challenge, black-box solvers [176] use an interpolated AP surface, though yielding little gain in object detection. DR Loss [70] achieves ranking between positives and negatives by enforcing a margin with Hinge Loss, which is differentiable. Finally, AP Loss [37] and aLRP Loss [156] optimise the performance metrics, AP and LRP [107] respectively, by using the error-driven update of perceptron learning [44] for the non-differentiable parts. The main difference of RS Loss is that it also considers continuous localisation qualities as labels.

Objective imbalance in VD. The common strategy in VD is to use λ_t^k (Eq. 8.1), a scalar multiplier, on each task and tune them by grid search [38, 43]. Recently, Oksuz et al. [156] employed a self-balancing strategy to balance classification and box regression heads, both of which compete for the bounded range of aLRP Loss. Similarly, Chen et al. [63] use the ratio of classification and regression losses to bal-

ance these tasks. In our design, each loss $\mathcal{L}_t^k$ for a specific head has its own bounded range and thus, no competition ensues among heads. Besides, we use $\mathcal{L}_t^k$ s with similar ranges, and show that our RS Loss can simply be combined with a simple task balancing strategy based on loss values, and hence does not require any tuning except the learning rate.

8.3 Rank & Sort Loss

In order to supervise the classifier of visual detectors by considering the localisation qualities of the predictions (e.g. IoU), RS Loss decomposes the problem into two tasks: (i) *Ranking task*, which aims to rank each positive higher than all negatives, and (ii) *sorting task*, which aims to sort the logits $\hat{s}_i$ in descending order wrt. continuous ground-truth labels y_i (e.g. IoUs). We define RS Loss and compute its gradients using our Identity Update (Section 6.2 – Fig. 6.1).

Definition. Given logits $\hat{s}_i$ and their continuous ground-truth labels $y_i \in [0, 1]$ (e.g. IoU), we define RS Loss as the average of the differences between the current ($\ell_{\text{RS}}(i)$) and target ($\ell_{\text{RS}}^*(i)$) RS errors over positives (i.e. $y_i > 0$):

$$\mathcal{L}_{\text{RS}} := \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} (\ell_{\text{RS}}(i) - \ell_{\text{RS}}^*(i)), \quad (8.2)$$

where $\ell_{\text{RS}}(i)$ is a summation of the current ranking error and current sorting error:

$$\ell_{\text{RS}}(i) := \underbrace{\frac{N_{\text{FP}}(i)}{\text{rank}(i)}}_{\ell_{\text{R}}(i): \text{Current Ranking Error}} + \underbrace{\frac{\sum_{j \in \mathcal{P}} H(x_{ij})(1 - y_j)}{\text{rank}^+(i)}}_{\ell_{\text{S}}(i): \text{Current Sorting Error}}. \quad (8.3)$$

For $i \in \mathcal{P}$, while the “current ranking error” is simply the precision error, the “current sorting error” penalizes the positives with logits larger than $\hat{s}_i$ by the average of their inverted labels, $1 - y_j$. Note that when $i \in \mathcal{P}$ is ranked above all $j \in \mathcal{N}$, $N_{\text{FP}}(i) = 0$ and target ranking error, $\ell_{\text{R}}^*(i)$, is 0. For target sorting error, we average over the inverted labels of $j \in \mathcal{P}$ with larger logits ($H(x_{ij})$) and labels ($y_j \geq y_i$) than $i \in \mathcal{P}$

corresponding to the desired sorted order,

$$\ell_{\text{RS}}^*(i) = \cancel{\ell_{\text{R}}^*(i)} + \underbrace{\frac{\sum_{j \in \mathcal{P}} \text{H}(x_{ij})[y_j \geq y_i](1 - y_j)}{\sum_{j \in \mathcal{P}} \text{H}(x_{ij})[y_j \geq y_i]}}_{\ell_{\text{S}}^*(i): \text{Target Sorting Error}}, \quad (8.4)$$

where $[P]$ is the Iverson Bracket (i.e. 1 if predicate P is True; else 0), and similar to previous work [37], $\text{H}(x_{ij})$ is smoothed in the interval $[-\delta_{\text{RS}}, \delta_{\text{RS}}]$ as $x_{ij}/2\delta_{\text{RS}} + 0.5$.

Computation. We follow the three-step algorithm (Section 6.2, Fig. 6.1) and define primary terms, L_{ij} , using Eq. 6.18, which allows us to express the errors among positives as:

$$L_{ij} = \begin{cases} (\ell_{\text{R}}(i) - \ell_{\text{R}}^*(i)) p_{\text{R}}(j|i), & \text{for } i \in \mathcal{P}, j \in \mathcal{N} \\ (\ell_{\text{S}}(i) - \ell_{\text{S}}^*(i)) p_{\text{S}}(j|i), & \text{for } i \in \mathcal{P}, j \in \mathcal{P}, \\ 0, & \text{otherwise,} \end{cases} \quad (8.5)$$

where ranking ($p_{\text{R}}(j|i)$) and sorting pmfs ($p_{\text{S}}(j|i)$) uniformly distribute ranking and sorting errors on i respectively over examples causing error (i.e. for ranking, $j \in \mathcal{N}$ with $\hat{s}_j > \hat{s}_i$; for sorting, $j \in \mathcal{P}$ with $\hat{s}_j > \hat{s}_i$ but $y_j < y_i$):

$$p_{\text{R}}(j|i) = \frac{\text{H}(x_{ij})}{\sum_{k \in \mathcal{N}} \text{H}(x_{ik})}; p_{\text{S}}(j|i) = \frac{\text{H}(x_{ij})[y_j < y_i]}{\sum_{k \in \mathcal{P}} \text{H}(x_{ik})[y_k < y_i]}, \quad (8.6)$$

Optimisation. To obtain $\frac{\partial \mathcal{L}_{\text{RS}}}{\partial \hat{s}_i}$, we simply replace Δx_{ij} (Eq. 6.15) by the primary terms of RS Loss, L_{ij} (Eq. 8.5), following Identity Update (Section 6.2). The resulting $\frac{\partial \mathcal{L}_{\text{RS}}}{\partial \hat{s}_i}$ for $i \in \mathcal{N}$ then becomes (see Appendix M for derivations):

$$\frac{\partial \mathcal{L}_{\text{RS}}}{\partial \hat{s}_i} = \frac{1}{|\mathcal{P}|} \sum_{j \in \mathcal{P}} \ell_{\text{R}}(j) p_{\text{R}}(i|j). \quad (8.7)$$

Owing to the additional sorting error (Eq. 8.3, 8.4), $\frac{\partial \mathcal{L}_{\text{RS}}}{\partial \hat{s}_i}$ for $i \in \mathcal{P}$ includes update signals for both promotion and demotion to sort the positives accordingly:

$$\frac{1}{|\mathcal{P}|} \left(\underbrace{\ell_{\text{RS}}^*(i) - \ell_{\text{RS}}(i)}_{\text{Update signal to promote } i} + \underbrace{\sum_{j \in \mathcal{P}} (\ell_{\text{S}}(j) - \ell_{\text{S}}^*(j)) p_{\text{S}}(i|j)}_{\text{Update signal to demote } i} \right). \quad (8.8)$$

Note that the directions of the first and second part of Eq. 8.8 are different. To place $i \in \mathcal{P}$ in the desired ranking, $\ell_{\text{RS}}^*(i) - \ell_{\text{RS}}(i) \leq 0$ promotes i based on the

error computed on itself, whereas $(\ell_S(j) - \ell_S^*(j)) p_S(i|j) \geq 0$ demotes i based on the signal from $j \in \mathcal{P}$. We provide more insight for RS Loss and its gradients on an example in Appendix M.

8.4 Using RS Loss to Train Visual Detectors

In this section, we develop an overall loss function to train visual detectors with RS Loss, in which only the learning rate needs tuning. To do so, as commonly performed in the literature [175, 43], we analyse different design choices on ATSS [42], a state-of-the-art (SOTA) one-stage object detector (i.e. $k = 1$ in Eq. 8.1) in Section 8.4.2; and in Section 8.4.3, we extend our design to other architectures and tasks.

8.4.1 Dataset and Implementation Details

Unless explicitly specified, we use (i) standard configuration of each detector and only replace the loss function, (ii) mmdetection framework [100], (iii) 16 images with a size of 1333×800 in a single batch (4 images/GPU, Tesla V100) during training, (iv) $1 \times$ training schedule (12 epochs), (v) single-scale test with images with a size of 1333×800 , (vi) ResNet-50 backbone with FPN [29], (vii) COCO *trainval35K* (115K images) and *minival* (5k images) sets [13] to train and test our models, (viii) report COCO-style AP, denoted as AP^C .

8.4.2 Analysis and Tuning-Free Design Choices

ATSS [42] with its classification, box regression and centerness heads is originally trained by minimizing:

$$\mathcal{L}_{ATSS} = \mathcal{L}_{cls} + \lambda_{box} \mathcal{L}_{box} + \lambda_{ctr} \mathcal{L}_{ctr}, \quad (8.9)$$

where $\mathcal{L}_{cls}$ is Focal Loss [18]; $\mathcal{L}_{box}$ is GIoU Loss [93]; $\mathcal{L}_{ctr}$ is Cross-entropy Loss with continuous labels to supervise centerness prediction; and $\lambda_{box} = 2$ and $\lambda_{ctr} = 1$. We first remove the centerness head and replace $\mathcal{L}_{cls}$ by our RS Loss (Section 8.3), $\mathcal{L}_{RS}$, using $\text{IoU}(\hat{b}_i, b_i)$ between a prediction box ($\hat{b}_i$) and ground truth box (b_i) as the

soft labels:

$$\mathcal{L}_{RS-ATSS} = \mathcal{L}_{RS} + \lambda_{box} \mathcal{L}_{box}, \quad (8.10)$$

where λ_{box} , the *task-level balancing coefficient*, is generally set to a constant scalar by grid search.

Inspired by recent work [156, 63], we investigate two tuning-free heuristics to determine λ_{box} every iteration: (i) value-based: $\lambda_{box} = \mathcal{L}_{box} / \mathcal{L}_{RS}$, and (ii) magnitude-based: $\lambda_{box} = \left| \frac{\partial \mathcal{L}_{box}}{\partial \hat{\mathbf{b}}} \right| / \left| \frac{\partial \mathcal{L}_{RS}}{\partial \mathbf{s}} \right|$ where $|\cdot|$ is L1 norm, $\hat{\mathbf{b}}$ and $\mathbf{s}$ are box regression and classification head outputs respectively.

Next we delve into $\mathcal{L}_{box}$, which is defined as the weighted average of the individual losses of examples:

$$\mathcal{L}_{box} = \sum_{i \in \mathcal{P}} \frac{w^i}{\sum_{j \in \mathcal{P}} w^j} \mathcal{L}_{GIoU}(\hat{b}_i, b_i), \quad (8.11)$$

where $\mathcal{L}_{GIoU}(\hat{b}_i, b_i)$ is the GIoU Loss [93], and w^i is the *instance-level importance weight*. Unlike *no prioritization* (i.e. $w^i = 1$ for $i \in \mathcal{P}$), recently, a diverse set of heuristics assigns different importances over $i \in \mathcal{P}$: *centerness-based* importance [160, 42] aims to focus on the proposals (i.e. point or anchor) closer to the center of b_i , *score-based* heuristic [175] uses the maximum of confidence scores of a prediction as w^i , *IoU-based* approach [43] increases the losses of the predictions that are already better localised by $w^i = \text{IoU}(\hat{b}_i, b_i)$, and finally *ranking-based* weighting [156] uses $w^i = \frac{1}{|\mathcal{P}|} \left(\sum_{k \in \mathcal{P}} \frac{H(x_{ki})}{\text{rank}(k)} \right)$, where $H(\cdot)$ can be smoothed by an additional hyper-parameter (δ_{loc}).

Observations and Our Design Choices: In our experiments on ATSS trained with RS Loss (Table 8.1), we observed that: (i) value-based task balancing performs similar to tuning λ_{box} (~ 0 AP on average), (ii) instance-level weighting methods also perform similarly (largest gap is 0.2 AP). Thus, we use value-based task balancing and score-based instance weighting, which are both hyper-parameter-free and easily applicable to all networks. *With these design choices, Eq. 8.10 has only 1 hyper-parameter* (i.e. δ_{RS} in $H(\cdot)$, set to 0.50, to smooth the unit-step function).

Comparison with aLRP Loss: We also provide a comparative analysis of aLRP Loss and RS Loss in Appendix N. In that analysis, we first show that competing tasks for

Table 8.1: Comparison of instance- and task-level weighting methods on RS-ATSS. Instance-level importance weighting methods yield similar performance and value-based SB achieves similar performance with constant weighting. Thus, we use score-based weighting and value-based SB with RS Loss (underlined&bold), which are both tuning-free.

Instance-level importance weight (w^i)	Task-level balancing coefficient (λ_{box})				
	Constant weighting			Self-balance (SB)	
	1	2	3	value	magnitude
No prioritization	38.9	39.7	39.7	39.7	39.4
Centerness-based [160]	38.8	39.8	39.6	39.6	39.5
Score-based [175]	39.1	39.8	39.7	<u>39.9</u>	39.7
IoU-based [43]	39.0	39.7	39.8	39.7	39.6
Ranking-based [156]	39.1	39.9	39.6	39.9	39.6

the bounded range of aLRP Loss degrades performance especially when the models are trained 12 epochs following the common training schedule. However, we still observe a performance difference when the models are trained longer. This is mainly because the target of aLRP Loss is hand-crafted, hence does not have an intuitive interpretation, on the other hand, RS Loss aims to sort the positives as the target. See Appendix N for further discussion and see Appendix O for the relation of RS Loss with LRP Error and aLRP Loss.

8.4.3 Training Different Architectures

Fig. 8.2 presents a comparative overview on how we adopt RS Loss to train different architectures: When we use RS Loss to train the classifier (Fig. 8.2(b)), we remove auxiliary heads (e.g. IoU head in IoU-Net [130]) and sampling heuristics (e.g. OHEM in YOLACT [38], random sampling in Faster R-CNN [19]). We adopt score-based weighting in box regression and mask prediction heads, and prefer Dice Loss, instead of the common Cross-entropy Loss, to train mask prediction head for instance segmentation due to (i) its bounded range between 0 and 1, and (ii) holistic evaluation of the predictions, both similar to GIoU Loss. Finally, we set λ_t^k to scalar $\mathcal{L}_{cls}^k / \mathcal{L}_t^k$ (Eq. 8.1) every iteration (Fig. 8.2(c)) with the single exception of RPN and additionally

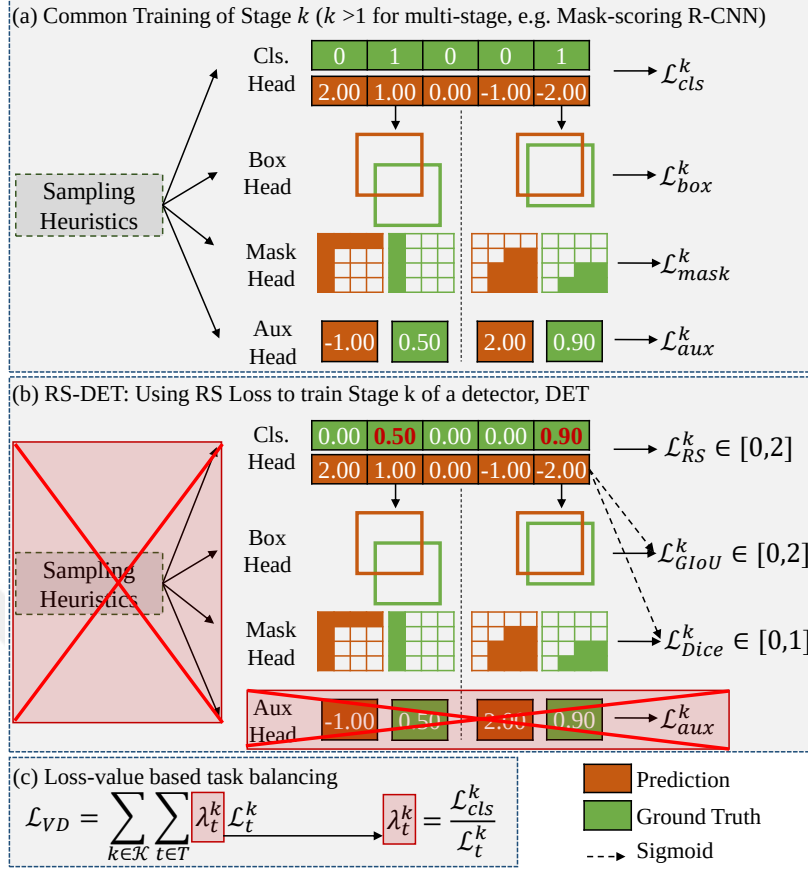


Figure 8.2: (a) A generic visual detection pipeline includes many heads from possibly multiple stages. An auxiliary (Aux) head, in addition to the standard ones, is common in recent methods (e.g. centerness head for ATSS [42], IoU head for IoU-Net [130], and mask IoU head for Mask-scoring R-CNN [41]) to regress localisation quality and prioritize examples during inference (e.g. by multiplying classification scores by the predicted localisation quality). Sampling heuristics are also common to ensure balanced training. Such architectures use many hyper-parameters and are sensitive for tuning. (b) Training detectors with our RS Loss removes (i) auxiliary heads by directly supervising the classification (Cls.) head with continuous IoUs (in red&bold), (ii) sampling heuristics owing to its robustness against class imbalance. We use losses with similar range with our RS Loss in other branches (i.e. GIoU Loss, Dice Loss) also by weighting each by using classification scores, obtained applying sigmoid to logits. (c) Instead of tuning λ_t^k s, we simply balance tasks by considering loss values. With this design, we train several detectors only by tuning the learning rate and improve their performance consistently.

multiply the losses of RPN by 0.20 following aLRP Loss.

8.5 Experiments

To present the contribution of RS Loss in terms of performance and tuning simplicity, we conduct experiments on seven visual detectors with a diverse set of architectures: four object detectors (i.e. Faster R-CNN [19], Cascade R-CNN [20], ATSS [42] and PAA [43] – Section 8.5.1) and three instance segmentation methods (i.e. Mask R-CNN [35], YOLACT [38] and SOLOv2 [179] – Section 8.5.2).

8.5.1 Experiments on Object Detection

Here, we train multi- and one-stage detectors with our RS Loss, and then compare our results with SOTA.

8.5.1.1 Multi-stage Object Detectors

To train Faster R-CNN [19] and Cascade R-CNN [20] by our RS Loss (i.e. RS-R-CNN), we remove sampling from all stages (i.e. RPN and R-CNNs), use all anchors to train RPN and m top-scoring proposals/image (by default, $m = 1000$ for Faster R-CNN and $m = 2000$ Cascade R-CNN in mmdetection [100]), replace softmax classifiers by binary sigmoid classifiers and set the initial learning rate to 0.012. Finally, multi-stage models trained by RS Loss generates detections with larger scores, hence we set NMS score threshold of such models to 0.40 for inference efficiency.

Comparison with different R-CNN variants: RS Loss reaches 39.6AP^{C} on a standard Faster R-CNN and outperforms (Table 8.2): (i) FPN [29] (Cross Entropy & Smooth L1 losses) by 3.4AP^{C} , (ii) aLRP Loss [156], a SOTA ranking-based baseline, by 2.2AP^{C} , (iii) IoU-Net [130] and KL Loss [90] methods with auxiliary heads by 1.5AP^{C} and 0.8AP^{C} respectively and (iv) Dynamic R-CNN, closest counterpart, by 0.7AP^{C} without any effect on inference time (Appendix P). We, then, use the lightweight Carafe [177] as the upsampling operation in FPN and obtain 40.8AP^{C}

Table 8.2: RS-R-CNN uses the standard IoU-based assigner, is sampling-free, employs no auxiliary (aux.) head, is almost tuning-free wrt. task-balancing weights (λ_t^k s – Eq. 8.1), and thus, has the least number of hyper-parameters ($H\# = 3 - \text{two } \delta_{RS}$, one for each RS Loss to train RPN & R-CNN, and one RPN weight). Still, RS-R-CNN improves standard Faster R-CNN with FPN by $\sim 3AP^C$; aLRP Loss (ranking-based loss baseline) by $\sim 2AP^C$; IoU-Net (a method with IoU head) by $1.5AP^C$. RS-R-CNN+ replaces upsampling of FPN by lightweight Carafe operation [177] and maintains $\sim 2AP^C$ gap from Carafe FPN (38.6 to $40.8 AP^C$). All models use ResNet-50, are evaluated in COCO *minival* and trained for 12 epochs on mmdetection except for IoU-Net and KL Loss. Var.: Variance. H#: Number of hyper-parameters. AP^C denotes COCO-style AP.

Method	Assigner	Sampler	Aux. Head	$AP^C \uparrow$	$AP_{50} \uparrow$	$AP_{75} \uparrow$	oLRP $\downarrow$	oLRP _{Loc} $\downarrow$	oLRP _{FP} $\downarrow$	oLRP _{FN} $\downarrow$	H# $\downarrow$	Reference
FPN [29]	IoU-based	Random	None	36.5	58.5	39.4	70.1	18.3	27.8	45.8	9	CVPR 17
aLRP Loss [156]	IoU-based	None	None	37.4	57.9	39.2	69.2	17.6	28.5	46.1	3	NeurIPS 20
GIoU Loss [93]	IoU-based	Random	None	37.6	58.2	41.0	69.2	17.0	28.5	46.3	7	CVPR 19
IoU-Net [130]	IoU-based	Random	IoU Head	38.1	56.3	–	–	–	–	–	11	ECCV 18
Libra R-CNN [32]	IoU-based	IoU-based	None	38.3	59.5	41.9	68.8	17.2	27.5	45.4	11	CVPR 19
AutoLoss-A [178]	IoU-based	Random	None	38.5	58.6	41.8	–	–	–	–	7	ICLR 21
Carafe FPN [177]	IoU-based	Random	None	38.6	59.9	42.2	68.3	17.2	27.0	44.2	7	ICCV 19
KL Loss [90]	IoU-based	Random	Var. Head	38.8	57.8	41.6	–	–	–	–	10	CVPR 19
Dynamic R-CNN [36]	Dynamic	Random	None	38.9	57.6	42.7	68.2	15.7	27.7	46.6	10	ECCV 20
RS-R-CNN (Ours)	IoU-based	None	None	39.6	59.5	43.0	67.9	16.3	27.8	45.4	3	
RS-R-CNN+ (Ours)	IoU-based	None	None	40.8	61.4	43.8	66.9	16.3	26.4	43.7	3	

(RS-R-CNN+), still maintaining $\sim 2\text{AP}^{\text{C}}$ gap from Carafe FPN [177] (38.6AP^{C}) and outperforming all methods in all AP- and oLRP-based [107] performance measures except oLRP_{Loc}, which implies that our main contribution is in classification task trained by our RS Loss and there is still room for improvement in the localisation task. RS Loss also improves the stronger baseline Cascade R-CNN [20] by 1AP^{C} from 40.3AP^{C} to 41.3AP^{C} (Appendix P presents detailed results for Cascade R-CNN).

Robustness to imbalance: Without tuning, RS Loss can train Faster R-CNN with sampling (i.e. data is balanced) or without sampling (i.e. data is imbalanced) consistently (Table 8.3). RS Loss utilizes more data when the samplers are removed, resulting in $\sim 1\text{AP}^{\text{C}}$ gain (38.5 to 39.6AP^{C}). Moreover, while we train Faster R-CNN with different distributions, unlike score-based losses (i.e. Focal Loss [18], QFL [175]), we do not tune the bias terms in the last layer of the classification head to prevent destabilization of the training due to large loss value originating from negatives, which is shown to be important for score-based loss functions [63].

Simplification of training: RS Loss has the least number of hyper-parameters ($H\# = 3$, Table 8.2). It does not need a sampler or an aux. head or tuning of λ_t^k s (Eq. 8.1). Appendix P presents hyper-parameters of methods.

Contribution of the sorting error: To see the contribution of our additional sorting error, during training, we track Spearman’s ranking correlation coefficient (ρ) between IoUs and classification scores, as an indicator of the sorting quality, with and without our additional sorting error (see Eq. 8.2-8.4). As hypothesized, using sorting error improves sorting quality, ρ , averaged over all/last 100 iterations, from $0.38/0.42$ to $0.42/0.47$ for RS-R-CNN.

8.5.1.2 One-stage Object Detectors

We train ATSS [42] and PAA [43] including a centerness head and an IoU head respectively in their architectures. We adopt the anchor configuration of Oksuz et al. [156] for all ranking-based losses (different anchor configurations do not affect performance of standard ATSS [42]) and set learning rate to 0.008. While training PAA,

Table 8.3: RS Loss is robust to class imbalance. It successfully trains Faster R-CNN with both relatively balanced (“Random” sampling) and severely imbalanced (“None” in the table) data. Numbers in parentheses show positive to negative ratio of sampled examples.

RPN	R-CNN	AP ^C	AP ₅₀	AP ₇₅
Random (1:1)	Random (1:3)	38.5	58.5	41.5
None	Random (1:3)	39.3	59.6	42.3
None	None	39.6	59.5	43.0

we keep the scoring function, splitting positives and negatives, for a fair comparison among different loss functions.

Comparison with AP and aLRP Losses, ranking-based baselines: We simply replaced Focal Loss by AP Loss to train networks, and as for aLRP Loss, similar to our RS Loss, we tuned its learning rate as 0.005 due to its tuning simplicity. Both for ATSS and PAA, RS Loss provides significant gains over ranking-based alternatives, which were trained for 100 epochs using SSD-like augmentation [24] in previous work [37, 156]: 1.8/2.2AP^C gain for ATSS and 3.7/3.3AP^C for PAA for AP/aLRP Loss (Table 8.4).

Comparison with Focal Loss, default loss function: RS Loss provides around ~ 1 AP gain when both networks are equally trained without an aux. head (Table 8.4) and 0.6AP^C gain compared to the default networks with aux. heads.

Comparison with QFL, score-based loss function using continuous IoUs as labels: To apply QFL [175] to PAA, we remove the auxiliary IoU head (as we did with ATSS), test two possible options ((i) default PAA setting with $\lambda_{box} = 1.3$ and IoU-based weighting, (ii) default QFL setting: $\lambda_{box} = 2.0$ and score-based weighting – Section 8.4.2) and report the best result for QFL. While the results of QFL and RS Loss are similar for ATSS, there is 0.8AP^C gap in favor of our RS Loss, which can be due to the different positive-negative labelling method in PAA (Table 8.4).

Table 8.4: RS Loss has the least number of hyper-parameters (H#) and outperforms (i) rank-based alternatives significantly, (ii) the default setting with an auxiliary head (underlined) by 0.6 AP, (iii) score-based alternative, QFL, especially on PAA. We test unified losses (i.e. a loss considering localisation quality while training classification head) only without auxiliary head. All models use ResNet-50. AP^C denotes COCO-style AP.

Loss Function	Unified	Rank-based	Aux. Head	ATSS [42]				PAA [43]				H# ↓
				$AP^C \uparrow$	$AP_{50} \uparrow$	$AP_{75} \uparrow$	oLRP ↓	AP ↑	$AP_{50} \uparrow$	$AP_{75} \uparrow$	oLRP ↓	
Focal Loss [18]			✓	38.7	57.6	41.5	68.9	39.9	57.3	43.4	68.7	3
				39.3	57.5	42.6	68.6	40.4	58.4	43.9	67.7	4
AP Loss [37]		✓	✓	38.1	58.2	41.0	69.2	35.3	53.1	38.5	71.5	2
		✓		37.2	55.6	40.2	70.0	37.3	54.3	41.2	70.5	3
QFL [175]	✓			39.7	58.1	42.7	68.0	40.2	57.4	43.8	68.3	2
aLRP Loss [156]	✓	✓		37.7	57.4	39.9	69.4	37.7	56.1	40.1	69.9	1
RS Loss (Ours)	✓	✓		39.9	58.9	42.6	67.9	41.0	59.1	44.5	67.3	1

Table 8.5: Comparison with SOTA for object detection on COCO *test-dev*. All methods use ResNet-101 and DCN. The result of the similarly trained Faster R-CNN is acquired from Zhang et al. [36]. +: upsampling of FPN is Carafe [177]

	Method	AP ^C	AP ₅₀	AP ₇₅	AP _S	AP _M	AP _L
One-stage	ATSS [42]	46.3	64.7	50.4	27.7	49.8	58.4
	GFL [175]	47.3	66.3	51.4	28.0	51.1	59.2
	PAA [43]	47.4	65.7	51.6	27.9	51.3	60.6
	RepPointsv2 [180]	48.1	67.5	51.8	28.7	50.9	60.8
Multi-stage	Faster R-CNN [36]	44.8	65.5	48.8	26.2	47.6	58.1
	Trident Net [81]	46.8	67.6	51.5	28.0	51.2	60.5
	Dynamic R-CNN [36]	46.9	65.9	51.3	28.1	49.6	60.0
	D2Det [181]	47.4	65.9	51.7	27.2	50.4	61.3
Ours	RS-R-CNN	47.8	68.0	51.8	28.5	51.1	61.6
	RS-R-CNN+	48.2	68.6	52.4	29.0	51.3	61.7
	RS-Mask R-CNN+	49.0	69.2	53.4	29.9	52.4	62.8

8.5.1.3 Comparison with SOTA

Here, we use our RS-R-CNN since it yields the largest improvement over its baseline. We train RS-R-CNN for 36 epochs using multiscale training by randomly resizing the shorter size within $[480, 960]$ on ResNet-101 with DCNv2 [171], and report the results on COCO *test-dev* in Table 8.5: Our RS-R-CNN reaches 47.8AP^C at 14.1 fps and outperforms similarly trained Faster R-CNN and Dynamic R-CNN by $\sim 3\text{AP}^C$ and $\sim 1\text{AP}^C$ respectively. Although we do not increase the number of parameters for Faster R-CNN, RS R-CNN also outperforms all multi-stage detectors including TridentNet [81], which has more parameters and inference time. Our RS-R-CNN+ (Section 8.5.1.1) reaches 48.2AP^C at 13.6 fps, and RS-Mask R-CNN+ (see Section 8.5.2) reaches 49.0AP^C at 13.5 fps, outperforming all one- and multi-stage counterparts.

8.5.2 Experiments on Instance Segmentation

Similar to Section 8.5.1, we train multi- and one-stage methods with RS Loss and compare our results with SOTA.

8.5.2.1 Multi-stage Instance Segmentation Methods

Here, we train the common baseline Mask R-CNN [35] by keeping all design choices of Faster R-CNN the same and, observe ~ 2 AP gain for both segmentation and detection performance (Table 8.6) over Mask R-CNN. Also, as hypothesized, RS-Mask R-CNN outperforms Mask Scoring R-CNN [41], with an additional mask IoU head, by 0.4 and 1.8 mask and box AP^C ; and by 0.9 and 1.5 mask and box oLRP respectively. Compared to Mask R-CNN, one training iteration of RS-Mask R-CNN takes around $1.5\times$ longer on average while RS Loss has no effect on inference time.

8.5.2.2 One-stage Instance Segmentation Methods

Here, we train two different approaches with our RS Loss: (i) YOLACT [38], a real-time instance segmentation method, involving sampling heuristics (e.g. OHEM [27]), auxiliary head and carefully-tuned loss weight, and demonstrate RS Loss can discard all by improving its performance (ii) SOLOv2 [179] as an anchor-free SOTA method.

YOLACT: Following YOLACT [38], we train (also test) RS-YOLACT by images with size 550×550 for 55 epochs. Instead of searching for epochs to decay learning rate, carefully tuned for YOLACT as 20, 42, 49 and 52, we simply adopt cosine annealing with an initial learning rate of 0.006. Then, we remove (i) OHEM, (ii) semantic segmentation head, (iii) carefully tuned task weights (i.e. $\lambda_{box} = 1.5$, $\lambda_{mask} = 6.125$) and (iv) size-based normalization (i.e. normalization of mask head loss of each instance by the ground-truth area). Removing each heuristic ensues a slight to significant performance drop (at least requires retuning of λ_t – Table 8.7). After these simplifications, our RS-YOLACT improves baseline by 1.5 mask AP^C and 3.3 box AP^C .

SOLOv2: Following Wang et al. [179], we train anchor-free SOLOv2 with RS Loss for 36 epochs using multiscale training on its two different settings: (i) SOLOv2-light is the real-time setting with ResNet-34 and images with size 448×448 at inference. We use 32 images/batch and learning rate 0.012 for training. (ii) SOLOv2 is the SOTA setting with ResNet-101 and images with size 1333×800 at inference. We use 16 images/batch and learning rate 0.006 for training. Since SOLOv2 does not

Table 8.6: Without an aux. head, RS-Mask R-CNN improves Mask R-CNN by $\sim 2AP^C$ (i.e. COCO-style AP) and outperforms Mask-scoring R-CNN.

Method	Aux. Head	Segmentation Performance					Detection Performance					H# ↓
		$AP^C \uparrow$	$AP_{50} \uparrow$	$AP_{75} \uparrow$	$oLRP \downarrow$		$AP^C \uparrow$	$AP_{50} \uparrow$	$AP_{75} \uparrow$	$oLRP \downarrow$		
Mask R-CNN [35]	None	34.7	55.7	37.2	71.2		38.2	58.8	41.4	68.7		8
Mask Scoring R-CNN [41]	Mask IoU Head	36.0	55.8	38.7	71.0		38.2	58.8	41.7	69.0		9
RS-Mask R-CNN	None	36.4	57.3	39.2	70.1		40.0	59.8	43.4	67.5		3

Table 8.8: Comparison on anchor-free SOLOv2.

Method	Backbone	AP ^C	AP ₅₀	AP ₇₅	oLRP ↓	H# ↓
SOLOv2-light	ResNet-34	32.0	50.7	33.7	73.5	3
RS-SOLOv2-light	ResNet-34	32.6	51.7	34.2	72.7	1
SOLOv2	ResNet-101	39.1	59.8	41.9	67.3	3
RS-SOLOv2	ResNet-101	39.7	60.6	42.2	66.9	1

have a box regression head, we use Dice coefficient as the soft labels of RS Loss (see Appendix P for an analysis of using different localisation qualities as labels for instance segmentation). Again, RS Loss performs better than the baseline (i.e. Focal Loss and Dice Loss) only by tuning the learning rate (Table 8.8).

8.5.2.3 Comparison with SOTA

We use our RS-Mask R-CNN (i.e. standard Mask R-CNN with RS Loss) to compare with SOTA methods. In order to fit in 16GB memory of our V100 GPUs and keep all settings unchanged, we limit the number of maximum proposals in the mask head by 200, which can simply be omitted for GPUs with larger memory. Following our counterparts [179, 182], we first train RS-Mask R-CNN for 36 epochs with multiscale training between [640, 800] using ResNet-101 and reach 40.6 mask AP^C at 14.8 fps (Table 8.9), improving Mask R-CNN by 2.3 mask AP^C and outperforming all SOTA methods by a notable margin ($\sim 1\text{AP}^C$). Then, we train RS-Mask R-CNN+ (i.e. standard Mask R-CNN except upsampling of FPN is lightweight Carafe [177]) also by extending the multiscale range to [480, 960] and reach 42.0 mask AP^C at 14.3 fps, which even outperforms all models with DCN. With DCN [171] on ResNet-101, our RS-Mask R-CNN+ reaches 43.9 mask AP^C at 11.9 fps.

8.6 Conclusion

In this paper, we proposed RS Loss as a ranking-based loss function to train object detectors and instance segmentation methods. Unlike existing ranking-based losses, which aim to rank positives above negatives, our RS Loss also sorts positives wrt.

Table 8.9: Comparison with SOTA for instance segmentation on COCO *test-dev*. All methods use ResNet-101. The result of the similarly trained Mask R-CNN is acquired from Chen et al. [183].

	Method	AP ^C	AP ₅₀	AP ₇₅	AP _S	AP _M	AP _L
w/o DCN	Polar Mask [184]	32.1	53.7	33.1	14.7	33.8	45.3
	Mask R-CNN [183]	38.3	61.2	40.8	18.2	40.6	54.1
	SOLOv2 [179]	39.7	60.7	42.9	17.3	42.9	57.4
	Center Mask [182]	39.8	–	–	21.7	42.5	52.0
	RS-Mask R-CNN (Ours)	40.6	62.8	43.9	22.8	43.6	52.8
	RS-Mask R-CNN+ (Ours)	42.0	64.8	45.6	24.2	45.1	54.6
w DCN	Mask-scoring R-CNN [41]	39.6	60.7	43.1	18.8	41.5	56.2
	BlendMask [185]	41.3	63.1	44.6	22.7	44.1	54.5
	SOLOv2 [179]	41.7	63.2	45.1	18.0	45.0	61.6
	RS-Mask R-CNN+ (Ours)	43.9	67.1	47.6	25.6	47.0	57.8

their localisation qualities, which is consistent with NMS and the performance measure, AP. With RS Loss, we employed a simple, loss-value-based, tuning-free heuristic to balance all heads in the visual detectors. As a result, we showed on seven diverse visual detectors that RS Loss both consistently improves performance and significantly simplifies the training pipeline.



CHAPTER 9

CONCLUSION

This chapter concludes the thesis with a summary, a discussion and an outlook with the limitations and future work.

9.1 Summary

In this thesis, we identified the imbalance problems in object detection and proposed solutions using ranking-based loss functions based on performance measures to address these imbalance problems. We had to alleviate two major challenges to be able to employ such loss functions to train deep learning-based visual detection architectures:

- The first challenge was related to the choice of the performance measure. Having identified the limitations of Average Precision, we proposed a novel performance metric, LRP Error, to evaluate the performance of visual detections.
- The second challenge that we encountered was the non-differentiable nature of such ranking-based performance measures, e.g. AP and our LRP Error. To overcome this, we simplified and generalized the incorporation error-driven optimisation into backpropagation and coined our approach as Identity Update.

Then, we defined average LRP (aLRP) Loss and Rank & Sort (RS) Loss based on our LRP Error and optimized them using our Identity Update. We showed in our experiments that training visual detectors with our loss functions consistently simplifies training and improves performance.

With these, the thesis has made contributions to the visual detection literature with a taxonomy and a review on imbalance problems as well as novel methods for addressing them.

9.2 Discussion

As the first aim of this thesis, we provided a comprehensive review of the imbalance problems in object detection in which we identified four main imbalance problems, which are further divided into 8 sub-problems (Table 1.2, Fig. 3.1). Despite the fact that the term “imbalance” has been mentioned nearly in every visual detection paper, it was being used in various contexts by referring to different imbalance problems. For example, as a pioneer method, Fast R-CNN [22] uses it in the context of objective imbalance; Focal Loss implies foreground-background imbalance and SNIPER [31] refers to scale imbalance. With our review, for the first time, we clearly identified in what levels imbalance exists and can exist (i.e. potential imbalance problems) in deep object detectors, and hence, filled a notable gap in the literature. As an indicator, our repository of the papers addressing imbalance problems in object detection was listed among the “trending research list of papers-with-code” [186] short after we released it in August 2018 and reached almost 900 stars as of April 2020.

In order to reach our second aim, we first proposed LRP Error as a novel performance metric. To the best of our knowledge, our LRP Error is the first performance metric to challenge Average Precision to evaluate the performance of deep-learning era visual detectors. To do so, we identified important features that performance measures to evaluate visual detectors are expected to satisfy. Interestingly, despite its widespread usage, Average Precision fails to satisfy neither of these important features, which, in fact, indicate basic requirements such as considering all performance aspects precisely or interpretability. On the other hand, despite certain advantages of our LRP Error; considering that all the previous papers had reported their results using Average Precision and existing competitions [13, 48, 99, 148] have been relying on Average Precision, it has been difficult to shift the community from Average Precision to our LRP Error, which would make it more difficult to compare their results with papers before LRP Error. After our LRP Error, Average Precision was challenged at least two

times more [149, 153], however, it is still the most common performance measure to evaluate visual detectors.

Another challenge to reach our second aim was the non-differentiable nature of performance measures, hence using them with backpropagation requires a different method to obtain update signals. This challenge ensues due to the unit-step function in the computation of such-ranking based loss functions, which has either zero or infinite gradients. Concurrent to this thesis, several works have been published specifically to optimise Average Precision Loss. Brown et al. [187] replaced the unit step function by a sigmoid, Rolinek et al. [169] used numerical differentiation and Chen et al. [37] showed that error-driven update from perceptron learning also provide the required update signal. Different from these works, we used more complicated loss functions, both of which are based on our LRP Error [107]. As a result, based on the optimisation of AP Loss by Chen et al. [37], we came up with a more general and simple optimisation method as Identity Update. We conceive that by using Identity Update, the update signal of different ranking-based loss functions can easily be obtained.

Finally, we proposed two loss functions with three main benefits compared to conventional loss formulation that combines all individual losses by a scaler:

- Our loss functions are “simple-to-tune” with one hyper-parameter. Having examined the literature, the methods extensively resort to hyperparameters mainly in order to address the imbalance problems. For example, Dynamic R-CNN [36] to train a Faster R-CNN [19] has 10 hyper-parameters (we use 3 hyper-parameters to train Faster R-CNN), or ATSS network [42] has 5 hyperparameters (we have 1 hyper-parameter in this case). We also outperform both of these baselines with our RS Loss.
- Our loss functions consider “correlation”. Due to its improving effect, the recent methods [160, 42, 90, 130] also tend to combine the outputs of the classification and localisation tasks. Since IoU-Net [130] was published in ECCV 2018, the dominant choice to do so had been to include an additional auxiliary head (e.g. for IoU, centerness) during training, which is supervised to predict the localisation quality. Concurrent to this thesis, Li et al. [175] showed that removing this auxiliary head and training directly the localisation head to regress

the localisation quality improves the performance. With our RS Loss, we also achieve prioritisation among positive examples, but differently we enforce an additional target to sort positives with respect to their localisation qualities. As a result, different from Li et al. [175], the idea of RS Loss is in parallel with the performance evaluation of visual detections.

- Our loss functions can train “a diverse set” of object detection and instance segmentation methods. We believe that this is yet another notable outcome of this thesis since the common methods were relying on the common choice of the pipeline (i.e. one-stage or multi-stage) which they belong to. In particular, while two stage methods [23, 19, 36, 181] were using sampling¹ followed by cross-entropy loss; one-stage methods [18, 160, 42] were resorting to focal loss without any sampling. In this paper, we showed that this discrepancy can be removed, and instead we trained all methods following the same methodology (i.e. without any sampling by using our loss functions, see Fig. 8.2), and thus, not only simplified the training but also improved the performance of multi-stage object detection and instance segmentation methods by discarding the sampling from their pipelines.

9.3 Limitations and Future Work

This section presents the limitations of our work and provide further possible research directions based on this thesis.

To begin with, we limited the scope of our review of imbalance problems to object detection as a representative task among visual detection tasks (e.g. instance segmentation, keypoint detection). However, different visual detection tasks can have their own imbalance problems as well. For example, Keypoint R-CNN, an extension of Mask R-CNN [35] for keypoint detection, includes a keypoint prediction head in addition to the classification and box regression heads of conventional Faster R-CNN [19]. This additional head is supervised by a cross entropy loss to infer the single pixel where the corresponding keypoint lies in the predicted mask with a size of 56×56 .

¹ refers to hard-sampling in this context (Chapter 3)

Obviously, cast as a classification problem with more than 3000 classes, keypoint detection faces this additional imbalance problem during training. However, we have not investigated the effects of this problem and limited our scope to object detection.

Unlike our review, we kept the scope of our LRP Error more general and accordingly presented the usage of our LRP Error on four different visual detection tasks, i.e. object detection, keypoint detection, instance segmentation and panoptic segmentation. Besides, we included a section to discuss its usage for other detection tasks such as 3D object detection, but we have not provided any use-case for those tasks. Furthermore, while LRP can provide insight on each performance aspect of visual detection (i.e. false positive rate, false negative rate and localisation error) with its components corresponding to each of these aspects, LRP Error is not an in-depth analysis tool.

Our Identity Update is a general and simple framework for optimising ranking-based loss functions. With this, besides AP Loss, we optimised our more complicated loss functions, aLRP Loss and Rank & Sort Loss, and presented that the gradients obtained via Identity Update is able to train models with balanced and imbalanced data. On the other hand, first, since the scope of this thesis is limited to visual detection, we have not considered the application of different ranking-based loss functions over other machine learning problems such as retrieval tasks, for which ranking is a major concern for performance. Secondly, Identity Update provides provable balance in terms of “gradient magnitudes”, and in practice, we showed that this is important, otherwise for example, we observed the training diverges for aLRP Loss (Appendix L). While we showed it for our loss functions, whether the gradient magnitudes (or a different measure) is the central cause of imbalance needs further exploration. Finally, we use identity update to obtain the update rule of a single non-differentiable step in the sequence of gradients following chain rule. Its usage from a more general perspective, e.g. in the extreme case to replace all gradients in the sequence to obtain a derivative-free optimisation method, remains as an open issue.

We successfully trained a diverse set of seven visual detectors including object detection and instance segmentation models using our RS Loss. We also showed that the task-balancing coefficients can be replaced by the ratio between the RS Loss and the corresponding task loss when all losses have similar range. We believe that these

ideas can benefit from further research: To begin with, while these tuning-free coefficients yield SOTA performance with our simple heuristic, we have not thoroughly analysed for each method we train whether the performance can be improved more by employing more complex methods such as *gradnorm* [188] or *uncertainty-based task weighting* [189]. Furthermore, the application of RS Loss to other visual detection tasks (e.g. keypoint detection, 3D object detection) needs exploration. One can analyse whether prioritising positives are still important for such methods and also our tuning-free coefficients can still ensure balance across different tasks. Finally, one limitation of our ranking-based loss functions is average iteration time of our loss functions is $1.5\times$ longer on average. This is because unlike the score-based loss functions with linear time and space complexity, those for our loss functions are quadratic. While we use the thresholding tricks following Chen et al. for efficiency, a more systematic method for improving time and space complexity needs investigation.

As a result, we can say that direct optimisation of performance measures bears challenges; however, once addressed, compared with their score-based counterparts, the loss functions based on discriminative performance measures provide a natural balance during training, significantly simplify training and yield SOTA performance for visual detectors. On the other hand, while we make the first structured attempt to optimise such loss functions for visual detection, there still remain aforementioned major challenges to be addressed in order to be benefit from such loss functions from a more general perspective.

REFERENCES

- [1] Q. Fan, L. Brown, and J. Smith, “A closer look at faster r-cnn for vehicle detection,” in IEEE Intelligent Vehicles Symposium, 2016.
- [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 Transactions on Image Processing-Accepted, 2019.
- [3] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? the kitti vision benchmark suite,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2012.
- [4] X. Dai, “Hybridnet: A fast vehicle detection system for autonomous driving,” Signal Processing: Image Communication, vol. 70, pp. 79 – 88, 2019.
- [5] P. F. Jaeger, S. A. A. Kohl, S. Bickelhaupt, F. Isensee, T. A. Kuder, H. Schlemmer, and K. H. Maier-Hein, “Retina u-net: Embarrassingly simple exploitation of segmentation supervision for medical object detection,” arXiv, vol. 1811.08661, 2018.
- [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 Medical Image Computing and Computer Assisted Intervention (MICCAI) (A. F. Frangi, J. A. Schnabel, C. Davatzikos, C. Alberola-López, and G. Fichtinger, eds.), 2018.
- [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 IEEE/CVF International Conference on Computer Vision (ICCV), 2017.
- [8] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab, “Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again,” in IEEE/CVF International Conference on Computer Vision (ICCV), 2017.

- [9] B. Tekin, S. N. Sinha, and P. Fua, “Real-Time Seamless Single Shot 6D Object Pose Prediction,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
- [10] T. Hodan, F. Michel, E. Brachmann, W. Kehl, A. GlentBuch, D. Kraft, B. Drost, J. Vidal, S. Ihrke, X. Zabulis, C. Sahin, F. Manhardt, F. Tombari, T.-K. Kim, J. Matas, and C. Rother, “Bop: Benchmark for 6d object pose estimation,” in The European Conference on Computer Vision (ECCV), 2018.
- [11] G. Du, K. Wang, and S. Lian, “Vision-based robotic grasping from object localization, pose estimation, grasp detection to motion planning: A review,” arXiv, vol. 1905.06658, 2019.
- [12] I. Bozcan and S. Kalkan, “Cosmo: Contextualized scene modeling with boltzmann machines,” Robotics and Autonomous Systems, vol. 113, pp. 132–148, 2019.
- [13] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft COCO: Common Objects in Context,” in The European Conference on Computer Vision (ECCV), 2014.
- [14] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, “Detectron2.” <https://github.com/facebookresearch/detectron2>. (Last accessed: 10 July 2020).
- [15] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, “Object detection with discriminatively trained part-based models,” IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), vol. 32, no. 9, pp. 1627–1645, 2010.
- [16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in Advances in Neural Information Processing Systems (NeurIPS), 2012.
- [17] J. Redmon and A. Farhadi, “YOLO9000: Better, faster, stronger,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

- [18] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, “Focal loss for dense object detection,” IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), vol. 42, no. 2, pp. 318–327, 2020.
- [19] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), vol. 39, no. 6, pp. 1137–1149, 2017.
- [20] Z. Cai and N. Vasconcelos, “Cascade R-CNN: Delving into high quality object detection,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
- [21] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2014.
- [22] R. Girshick, “Fast R-CNN,” in IEEE/CVF International Conference on Computer Vision (ICCV), 2015.
- [23] J. Dai, Y. Li, K. He, and J. Sun, “R-FCN: Object detection via region-based fully convolutional networks,” in Advances in Neural Information Processing Systems (NeurIPS), 2016.
- [24] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, and A. C. Berg, “SSD: single shot multibox detector,” in The European Conference on Computer Vision (ECCV), 2016.
- [25] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
- [26] H. Law and J. Deng, “Cornersnet: Detecting objects as paired keypoints,” in The European Conference on Computer Vision (ECCV), 2018.
- [27] A. Shrivastava, A. Gupta, and R. Girshick, “Training region-based object detectors with online hard example mining,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

- [28] W. Ouyang, X. Wang, C. Zhang, and X. Yang, “Factors in finetuning deep model for object detection with long-tail distribution,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
- [29] T. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie, “Feature pyramid networks for object detection,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2017.
- [30] B. Singh and L. S. Davis, “An analysis of scale invariance in object detection - snip,” in The Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
- [31] B. Singh, M. Najibi, and L. S. Davis, “Sniper: Efficient multi-scale training,” in Advances in Neural Information Processing Systems (NeurIPS), 2018.
- [32] J. Pang, K. Chen, J. Shi, H. Feng, W. Ouyang, and D. Lin, “Libra R-CNN: Towards balanced learning for object detection,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
- [33] Y. Cao, K. Chen, C. C. Loy, and D. Lin, “Prime Sample Attention in Object Detection,” arXiv, vol. 1904.04821, 2019.
- [34] K. Chen, J. Pang, J. Wang, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Shi, W. Ouyang, C. C. Loy, and D. Lin, “Hybrid task cascade for instance segmentation,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
- [35] K. He, G. Gkioxari, P. Dollar, and R. Girshick, “Mask R-CNN,” in IEEE/CVF International Conference on Computer Vision (ICCV), 2017.
- [36] H. Zhang, H. Chang, B. Ma, N. Wang, and X. Chen, “Dynamic r-cnn: Towards high quality object detection via dynamic training,” in The European Conference on Computer Vision (ECCV), 2020.
- [37] K. Chen, W. Lin, J. li, J. See, J. Wang, and J. Zou, “Ap-loss for accurate one-stage object detection,” IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), pp. 1–1, 2020.

- [38] D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee, “Yolact: Real-time instance segmentation,” in IEEE/CVF International Conference on Computer Vision (ICCV), 2019.
- [39] X. Zhang, F. Wan, C. Liu, R. Ji, and Q. Ye, “Freeanchor: Learning to match anchors for visual object detection,” in Advances in Neural Information Processing Systems (NeurIPS), 2019.
- [40] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, “Centernet: Key-point triplets for object detection,” in IEEE/CVF International Conference on Computer Vision (ICCV), 2019.
- [41] Z. Huang, L. Huang, Y. Gong, C. Huang, and X. Wang, “Mask scoring rcnn,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
- [42] S. Zhang, C. Chi, Y. Yao, Z. Lei, and S. Z. Li, “Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.
- [43] K. Kim and H. S. Lee, “Probabilistic anchor assignment with iou prediction for object detection,” in The European Conference on Computer Vision (ECCV), 2020.
- [44] F. Rosenblatt, “The perceptron: A probabilistic model for information storage and organization in the brain,” Psychological Review, pp. 65–386, 1958.
- [45] A. Gupta, P. Dollar, and R. Girshick, “Lvis: A dataset for large vocabulary instance segmentation,” in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
- [46] H. Caesar, J. Uijlings, and V. Ferrari, “Coco-stuff: Thing and stuff classes in context,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
- [47] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

- [48] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," International Journal of Computer Vision (IJCV), vol. 88, no. 2, pp. 303–338, 2010.
- [49] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2009.
- [50] L. Liu, W. Ouyang, X. Wang, P. W. Fieguth, J. Chen, X. Liu, and M. Pietikäinen, "Deep learning for generic object detection: A survey," arXiv, vol. 1809.02165, 2018.
- [51] Z. Zou, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," arXiv, vol. 1905.05055, 2018.
- [52] S. Agarwal, J. O. D. Terrail, and F. Jurie, "Recent advances in object detection in the age of deep convolutional neural networks," arXiv, vol. 1809.03193, 2018.
- [53] Zehang Sun, G. Bebis, and R. Miller, "On-road vehicle detection: a review," IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), vol. 28, no. 5, pp. 694–711, 2006.
- [54] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), vol. 34, no. 4, pp. 743–761, 2012.
- [55] S. Zafeiriou, C. Zhang, and Z. Zhang, "A survey on face detection in the wild," Computer Vision and Image Understanding, vol. 138, pp. 1–24, 2015.
- [56] Q. Ye and D. Doermann, "Text detection and recognition in imagery: A survey," IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), vol. 37, no. 7, pp. 1480–1500, 2015.
- [57] X. Yin, Z. Zuo, S. Tian, and C. Liu, "Text detection, tracking and recognition in video: A comprehensive survey," IEEE Transactions on Image Processing, vol. 25, no. 6, pp. 2752–2773, 2016.

- [58] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. van der Laak, B. van Ginneken, and C. I. Sánchez, “A survey on deep learning in medical image analysis,” Medical Image Analysis, vol. 42, pp. 60 – 88, 2017.
- [59] B. Krawczyk, “Learning from imbalanced data: open challenges and future directions,” Progress in Artificial Intelligence, vol. 5, no. 4, pp. 221–232, 2016.
- [60] J. L. Leevy, T. M. Khoshgoftaar, R. A. Bauder, and N. Seliya, “A survey on addressing high-class imbalance in big data,” Journal of Big Data, vol. 5, no. 42, 2018.
- [61] A. Fernández, S. García, M. Galar, R. Prati, B. Krawczyk, and F. Herrera, Learning from Imbalanced Data Sets. Springer International Publishing, 2018.
- [62] J. M. Johnson, Khoshgoftaar, and T. M., “Survey on deep learning with class imbalance,” Journal of Big Data, vol. 6, no. 21, 2019.
- [63] J. Chen, D. Liu, T. Xu, S. Zhang, S. Wu, B. Luo, X. Peng, and E. Chen, “Is sampling heuristics necessary in training deep object detectors?,” arXiv, vol. 1909.04868, 2019.
- [64] H. A. Rowley, S. Baluja, and T. Kanade, “Human face detection in visual scenes,” in The Advances in Neural Information Processing Systems (NeurIPS), 1995.
- [65] T. Kong, F. Sun, A. Yao, H. Liu, M. Lu, and Y. Chen, “Ron: Reverse connection with objectness prior networks for object detection,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2017.
- [66] S. Zhang, L. Wen, X. Bian, Z. Lei, and S. Z. Li, “Single-shot refinement neural network for object detection,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
- [67] J. Nie, R. M. Anwer, H. Cholakkal, F. S. Khan, Y. Pang, and L. Shao, “Enriched feature guided refinement network for object detection,” in IEEE/CVF International Conference on Computer Vision (ICCV), 2019.

- [68] B. Li, Y. Liu, and X. Wang, "Gradient harmonized single-stage detector," in AAAI Conference on Artificial Intelligence, 2019.
- [69] J. Chen, D. Liu, B. Luo, X. Peng, T. Xu, and E. Chen, "Residual objectness for imbalance reduction," arXiv, vol. 1908.09075, 2019.
- [70] Q. Qian, L. Chen, H. Li, and R. Jin, "Dr loss: Improving object detection by distributional ranking," in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.
- [71] X. Wang, A. Shrivastava, and A. Gupta, "A-fast-rcnn: Hard positive generation via adversary for object detection," in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2017.
- [72] S. Tripathi, S. Chandra, A. Agrawal, A. Tyagi, J. M. Rehg, and V. Chari, "Learning to generate synthetic data via compositing," in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
- [73] H. Wang, Q. Wang, F. Yang, W. Zhang, and W. Zuo, "Data augmentation for object detection via progressive and selective instance-switching," arXiv, vol. 1906.00358, 2019.
- [74] K. Oksuz, B. C. Cam, E. Akbas, and S. Kalkan, "Generating positive bounding boxes for balanced training of object detectors," in IEEE Winter Applications on Computer Vision (WACV), 2020.
- [75] J. Wang, K. Chen, S. Yang, C. C. Loy, and D. Lin, "Region proposal by guided anchoring," in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
- [76] 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 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
- [77] Z. Cai, Q. Fan, R. Feris, and N. Vasconcelos, "A unified multi-scale deep convolutional neural network for fast object detection," in The European Conference on Computer Vision (ECCV), 2016.

- [78] J. Li, X. Liang, S. Shen, T. Xu, J. Feng, and S. Yan, "Scale-aware fast r-cnn for pedestrian detection," IEEE Transactions on Multimedia, vol. 20, no. 4, pp. 985–996, 2018.
- [79] Y. Pang, T. Wang, R. M. Anwer, F. S. Khan, and L. Shao, "Efficient featurized image pyramid network for single shot detector," in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
- [80] 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 IEEE/CVF International Conference on Computer Vision (ICCV), 2019.
- [81] Y. Li, Y. Chen, N. Wang, and Z. Zhang, "Scale-aware trident networks for object detection," in IEEE/CVF International Conference on Computer Vision (ICCV), 2019.
- [82] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
- [83] P. Zhou, B. Ni, C. Geng, J. Hu, and Y. Xu, "Scale-transferrable object detection," in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
- [84] S.-W. Kim, H.-K. Kook, J.-Y. Sun, M.-C. Kang, and S.-J. Ko, "Parallel feature pyramid network for object detection," in The European Conference on Computer Vision (ECCV), 2018.
- [85] T. Kong, F. Sun, W. Huang, and H. Liu, "Deep feature pyramid reconfiguration for object detection," in The European Conference on Computer Vision (ECCV), 2018.
- [86] H. Li, Y. Liu, W. Ouyang, and X. Wang, "Zoom out-and-in network with map attention decision for region proposal and object detection," International Journal of Computer Vision, vol. 127, no. 3, pp. 225–238, 2019.

- [87] Q. Zhao, T. Sheng, Y. Wang, Z. Tang, Y. Chen, L. Cai, and H. Ling, “M2det: A single-shot object detector based on multi-level feature pyramid network,” in AAAI Conference on Artificial Intelligence, 2019.
- [88] G. Ghiasi, T. Lin, R. Pang, and Q. V. Le, “NAS-FPN: learning scalable feature pyramid architecture for object detection,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
- [89] H. Xu, L. Yao, W. Zhang, X. Liang, and Z. Li, “Auto-fpn: Automatic network architecture adaptation for object detection beyond classification,” in IEEE/CVF International Conference on Computer Vision (ICCV), 2019.
- [90] Y. He, C. Zhu, J. Wang, M. Savvides, and X. Zhang, “Bounding box regression with uncertainty for accurate object detection,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
- [91] 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.
- [92] L. Tychsen-Smith and L. Petersson, “Improving object localization with fitness nms and bounded iou loss,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
- [93] 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 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
- [94] 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 AAAI Conference on Artificial Intelligence, 2020.
- [95] J. Cao, Y. Pang, J. Han, and X. Li, “Hierarchical shot detector,” in IEEE/CVF International Conference on Computer Vision (ICCV), 2019.
- [96] Z. Li, Z. Xie, L. Liu, B. Tao, and W. Tao, “Iou-uniform r-cnn: Breaking through the limitations of rpn,” arXiv, vol. 1912.05190, 2019.

- [97] X. Zhou, J. Zhuo, and P. Krahenbuhl, “Bottom-up object detection by grouping extreme and center points,” in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
- [98] A. Kuznetsova, H. Rom, N. Alldrin, J. R. R. Uijlings, I. Krasin, J. Pont-Tuset, S. Kamali, S. Popov, M. Mallocci, T. Duerig, and V. Ferrari, “The open images dataset V4: unified image classification, object detection, and visual relationship detection at scale,” arXiv, vol. 1811.00982, 2018.
- [99] S. Shao, Z. Li, T. Zhang, C. Peng, G. Yu, X. Zhang, J. Li, and J. Sun, “Objects365: A large-scale, high-quality dataset for object detection,” in IEEE/CVF International Conference on Computer Vision (ICCV), 2019.
- [100] K. Chen, J. Wang, J. Pang, Y. Cao, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, J. Xu, Z. Zhang, D. Cheng, C. Zhu, T. Cheng, Q. Zhao, B. Li, X. Lu, R. Zhu, Y. Wu, J. Dai, J. Wang, J. Shi, W. Ouyang, C. C. Loy, and D. Lin, “MMDetection: Open mmlab detection toolbox and benchmark,” arXiv, vol. 1906.07155, 2019.
- [101] K.-K. Sung and T. Poggio, “Example-based learning for view-based human face detection,” IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), vol. 20, no. 1, pp. 39–51, 1998.
- [102] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2001.
- [103] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2005.
- [104] D. Dwibedi, I. Misra, and M. Hebert, “Cut, paste and learn: Surprisingly easy synthesis for instance detection,” in IEEE/CVF International Conference on Computer Vision (ICCV), 2017.
- [105] N. Dvornik, J. Mairal, and C. Schmid, “Modeling visual context is key to augmenting object detection datasets,” in The European Conference on Computer Vision (ECCV), 2018.

- [106] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2015.
- [107] K. Oksuz, B. C. Cam, E. Akbas, and S. Kalkan, “Localization recall precision (LRP): A new performance metric for object detection,” in The European Conference on Computer Vision (ECCV), 2018.
- [108] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in The International Conference on Learning Representations (ICLR), 2015.
- [109] A. Newell, K. Yang, and J. Deng, “Stacked hourglass networks for human pose estimation,” in The European Conference on Computer Vision (ECCV) (B. Leibe, J. Matas, N. Sebe, and M. Welling, eds.), 2016.
- [110] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, “Mobilenets: Efficient convolutional neural networks for mobile vision applications,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2017.
- [111] Z. Li, C. Peng, G. Yu, X. Zhang, Y. Deng, and J. Sun, “Detnet: Design backbone for object detection,” in The European Conference on Computer Vision (ECCV), 2018.
- [112] L. Zhu, Z. Deng, X. Hu, C.-W. Fu, X. Xu, J. Qin, and P.-A. Heng, “Bidirectional feature pyramid network with recurrent attention residual modules for shadow detection,” in The European Conference on Computer Vision (ECCV), 2018.
- [113] Y. Sun, P. S. K. P, J. Shimamura, and A. Sagata, “Concatenated feature pyramid network for instance segmentation,” arXiv, vol. 1904.00768, 2019.
- [114] S. S. Seferbekov, V. I. Iglovikov, A. V. Buslaev, and A. A. Shvets, “Feature pyramid network for multi-class land segmentation,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2018.

- [115] A. Kirillov, R. B. Girshick, K. He, and P. Dollár, “Panoptic feature pyramid networks,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
- [116] 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.
- [117] F. Yu and V. Koltun, “Multi-scale context aggregation by dilated convolutions,” in The International Conference on Learning Representations (ICLR), 2016.
- [118] X. Wang, R. Girshick, A. Gupta, and K. He, “Non-local neural networks,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
- [119] G. Huang, Z. Liu, and K. Q. Weinberger, “Densely connected convolutional networks,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2017.
- [120] K. He, X. Zhang, S. Ren, and J. Sun, “Spatial pyramid pooling in deep convolutional networks for visual recognition,” in The European Conference on Computer Vision (ECCV) (D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, eds.), 2014.
- [121] J. Hu, L. Shen, and G. Sun, “Squeeze-and-excitation networks,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
- [122] S. Ioffe and C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift,” in The International Conference on Machine Learning (ICML), 2015.
- [123] B. Zoph and Q. V. Le, “Neural architecture search with reinforcement learning,” in The International Conference on Learning Representations (ICLR), 2017.
- [124] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, “Learning transferable architectures for scalable image recognition,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

- [125] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le, “Regularized evolution for image classifier architecture search,” in AAAI Conference on Artificial Intelligence, 2019.
- [126] M. Tan and Q. V. Le, “Efficientnet: Rethinking model scaling for convolutional neural networks,” in The International Conference on Machine Learning (ICML), 2019.
- [127] Y. Chen, T. Yang, X. Zhang, G. Meng, C. Pan, and J. Sun, “Detnas: Backbone search for object detection,” in Advances in Neural Information Processing Systems (NeurIPS), 2019.
- [128] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. Lecun, “Overfeat: Integrated recognition, localization and detection using convolutional networks,” in International Conference on Learning Representations (ICLR), 2014.
- [129] P. J. Huber, “Robust estimation of a location parameter,” Annals of Statistics, vol. 53, no. 1, pp. 73–101, 1964.
- [130] B. Jiang, R. Luo, J. Mao, T. Xiao, and Y. Jiang, “Acquisition of localization confidence for accurate object detection,” in The European Conference on Computer Vision (ECCV), 2018.
- [131] E. Goldman, R. Herzig, A. Eisenschstat, J. Goldberger, and T. Hassner, “Precise detection in densely packed scenes,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
- [132] 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 IEEE/CVF International Conference on Computer Vision (ICCV), 2019.
- [133] Z. Tan, X. Nie, Q. Qian, N. Li, and H. Li, “Learning to rank proposals for object detection,” in IEEE/CVF International Conference on Computer Vision (ICCV), 2019.
- [134] S. Gidaris and N. Komodakis, “Object detection via a multi-region and semantic segmentation-aware cnn model,” in IEEE/CVF International Conference on Computer Vision (ICCV), 2015.

- [135] S. Gidaris and N. Komodakis, “Attend refine repeat: Active box proposal generation via in-out localization,” in The British Machine Vision Conference (BMVC), 2016.
- [136] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, “Deformable convolutional networks,” in IEEE/CVF International Conference on Computer Vision (ICCV), 2017.
- [137] T. Yang, X. Zhang, Z. Li, W. Zhang, and J. Sun, “Metaanchor: Learning to detect objects with customized anchors,” in Advances in Neural Information Processing Systems (NeurIPS), 2018.
- [138] C. Peng, T. Xiao, Z. Li, Y. Jiang, X. Zhang, K. Jia, G. Yu, and J. Sun, “Megdet: A large mini-batch object detector,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
- [139] M. Guo, A. Haque, D.-A. Huang, S. Yeung, and L. Fei-Fei, “Dynamic task prioritization for dynamic task learning,” in The European Conference on Computer Vision (ECCV), 2018.
- [140] N. Samet, S. Hicsonmez, and E. Akbas, “Houghnet: Integrating near and long-range evidence for bottom-up object detection,” in European Conference on Computer Vision (ECCV), 2020.
- [141] S. Xie, R. B. Girshick, P. Dollár, Z. Tu, and K. He, “Aggregated residual transformations for deep neural networks,” arXiv, vol. 1611.05431, 2016.
- [142] K. Oksuz, B. C. Cam, S. Kalkan, and E. Akbas, “Imbalance problems in object detection: A review,” IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), pp. 1–1, 2020.
- [143] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in Advances in Neural Information Processing Systems (NIPS), 2014.
- [144] S. Li, L. Zhouche, and H. Qingming, “Relay Backpropagation for Effective Learning of Deep Convolutional Neural Networks,” in The European Conference on Computer Vision (ECCV), 2016.

- [145] A. T. Cemgil, “A tutorial introduction to monte carlo methods, markov chain monte carlo and particle filtering,” 2013.
- [146] J. Yang, J. Lu, D. Batra, and D. Parikh, “A faster pytorch implementation of faster r-cnn,” <https://github.com/jwyang/faster-rcnn.pytorch>, Last Accessed: 24 April 2019.
- [147] K. Oksuz, B. C. Cam, S. Kalkan, and E. Akbas, “One metric to measure them all: Localisation recall precision (lrp) for evaluating visual detection tasks,” arXiv (under review), vol. 2011.10772, 2020.
- [148] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, “The cityscapes dataset for semantic urban scene understanding,” in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
- [149] D. Hall, F. Dayoub, J. Skinner, H. Zhang, D. Miller, P. Corke, G. Carneiro, A. Angelova, and N. Suenderhauf, “Probabilistic object detection: Definition and evaluation,” in Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), 2020.
- [150] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollar, “Panoptic segmentation,” in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2019.
- [151] F. Bourgeois and J.-C. Lassalle, “An extension of the munkres algorithm for the assignment problem to rectangular matrices,” Communications of ACM, vol. 14, no. 12, pp. 802–804, 1971.
- [152] D. Hoiem, Y. Chodpathumwan, and Q. Dai, “Diagnosing error in object detectors,” in The IEEE European Conference on Computer Vision (ECCV), 2012.
- [153] D. Bolya, S. Foley, J. Hays, and J. Hoffman, “Tide: A general toolbox for identifying object detection errors,” in The IEEE European Conference on Computer Vision (ECCV), 2020.
- [154] D. Schuhmacher, B. T. Vo, and B. N. Vo, “A consistent metric for performance evaluation of multi-object filters,” IEEE Transactions on Signal Processing, vol. 56, no. 8, pp. 3447 – 3457, 2008.

- [155] K. Oksuz and A. T. Cemgil, “Multitarget tracking performance metric: deficiency aware subpattern assignment,” IET Radar, Sonar Navigation, vol. 12, no. 3, pp. 373–381, 2018.
- [156] K. Oksuz, B. C. Cam, E. Akbas, and S. Kalkan, “A ranking-based, balanced loss function unifying classification and localisation in object detection,” in Advances in Neural Information Processing Systems (NeurIPS), 2020.
- [157] H. Qiu, H. Li, Q. Wu, and H. Shi, “Offset bin classification network for accurate object detection,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.
- [158] C. Feichtenhofer, A. Pinz, and A. Zisserman, “Detect to track and track to detect,” in IEEE/CVF International Conference on Computer Vision (ICCV), 2017.
- [159] Y. Lu, C. Lu, and C. Tang, “Online video object detection using association lstm,” in IEEE International Conference on Computer Vision (ICCV), 2017.
- [160] Z. Tian, C. Shen, H. Chen, and T. He, “Fcos: Fully convolutional one-stage object detection,” in IEEE/CVF International Conference on Computer Vision (ICCV), 2019.
- [161] R. Girshick, I. Radosavovic, G. Gkioxari, P. Dollár, and K. He, “Detectron.” <https://github.com/facebookresearch/detectron>. (Last accessed: 10 July 2020).
- [162] K. Oksuz, B. C. Cam, E. Akbas, and S. Kalkan, “Ablation experiments repository of alrp loss.” <https://github.com/kemaloksuz/aLRPLoss-AblationExperiments>. (Last accessed: 16 November 2020).
- [163] X. Lu, B. Li, Y. Yue, Q. Li, and J. Yan, “Grid r-cnn,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
- [164] Z. Yang, S. Liu, H. Hu, L. Wang, and S. Lin, “Reppoints: Point set representation for object detection,” in IEEE/CVF International Conference on Computer Vision (ICCV), 2019.

- [165] K. Oksuz, B. C. Cam, E. Akbas, and S. Kalkan, “Rank & sort loss for object detection and instance segmentation,” in under review, 2021.
- [166] B. C. Cam, “Training object detectors by direct optimization of lrp metric,” Master’s thesis, Middle East Technical University, 2020.
- [167] F. Chabot, Q.-C. Pham, and M. Chaouch, “Lapnet : Automatic balanced loss and optimal assignment for real-time dense object detection,” arXiv, vol. 1911.01149, 2019.
- [168] A. Kendall, Y. Gal, and R. Cipolla, “Multi-task learning using uncertainty to weigh losses for scene geometry and semantics,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2018.
- [169] M. V. Pogančić, A. Paulus, V. Musil, G. Martius, and M. Rolinek, “Differentiation of blackbox combinatorial solvers,” in International Conference on Learning Representations (ICLR), 2020.
- [170] K. Chen, “Ap-loss,” <https://github.com/cccorn/AP-loss>, Last Accessed: 14 May 2020.
- [171] X. Zhu, H. Hu, S. Lin, and J. Dai, “Deformable convnets v2: More deformable, better results,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
- [172] X. Zhou, J. Zhuo, and P. Krahenbuhl, “Bottom-up object detection by grouping extreme and center points,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
- [173] N. Bodla, B. Singh, R. Chellappa, and L. S. Davis, “Soft-nms – improving object detection with one line of code,” in IEEE/CVF International Conference on Computer Vision (ICCV), 2017.
- [174] T. Kong, F. Sun, H. Liu, Y. Jiang, L. Li, and J. Shi, “Foveabox: Beyond anchor-based object detection,” IEEE Transactions on Image Processing, vol. 29, pp. 7389–7398, 2020.
- [175] X. Li, W. Wang, L. Wu, S. Chen, X. Hu, J. Li, J. Tang, and J. Yang, “Generalized focal loss: Learning qualified and distributed bounding boxes for dense

- object detection,” in Advances in Neural Information Processing Systems (NeurIPS), 2020.
- [176] M. Rolínek, V. Musil, A. Paulus, M. Vlastelica, C. Michaelis, and G. Martius, “Optimizing rank-based metrics with blackbox differentiation,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.
 - [177] J. Wang, K. Chen, R. Xu, Z. Liu, C. C. Loy, and D. Lin, “Carafe: Content-aware reassembly of features,” in IEEE/CVF International Conference on Computer Vision (ICCV), 2019.
 - [178] P. Liu, G. Zhang, B. Wang, H. Xu, X. Liang, Y. Jiang, and Z. Li, “Loss function discovery for object detection via convergence-simulation driven search,” in International Conference on Learning Representations (ICLR), 2021.
 - [179] X. Wang, R. Zhang, T. Kong, L. Li, and C. Shen, “Solov2: Dynamic and fast instance segmentation,” in Advances in Neural Information Processing Systems (NeurIPS), 2020.
 - [180] Y. Chen, Z. Zhang, Y. Cao, L. Wang, S. Lin, and H. Hu, “Reppoints v2: Verification meets regression for object detection,” in Advances in Neural Information Processing Systems (NeurIPS), 2020.
 - [181] J. Cao, H. Cholakkal, R. M. Anwer, F. S. Khan, Y. Pang, and L. Shao, “D2det: Towards high quality object detection and instance segmentation,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.
 - [182] Y. Wang, Z. Xu, H. Shen, B. Cheng, and L. Yang, “Centermask: Single shot instance segmentation with point representation,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.
 - [183] X. Chen, R. Girshick, K. He, and P. Dollár, “Tensormask: A foundation for dense object segmentation,” in IEEE/CVF International Conference on Computer Vision (ICCV), 2019.
 - [184] E. Xie, P. Sun, X. Song, W. Wang, X. Liu, D. Liang, C. Shen, and P. Luo, “Polarmask: Single shot instance segmentation with polar representation,” in

IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.

- [185] H. Chen, K. Sun, Z. Tian, C. Shen, Y. Huang, and Y. Yan, “Blendmask: Top-down meets bottom-up for instance segmentation,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020.
- [186] “Trending research of papers-with-code.” <https://paperswithcode.com/>. (Last accessed: 21 April 2021).
- [187] A. Brown, W. Xie, V. Kalogeiton, and A. Zisserman, “Smooth-ap: Smoothing the path towards large-scale image retrieval,” in European Conference on Computer Vision (ECCV), 2020., 2020.
- [188] C.-Y. L. Zhao Chen, Vijay Badrinarayanan and A. Rabinovich, “Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks,” in International Conference on Machine Learning (ICML), 2018.
- [189] A. Kendall, Y. Gal, and R. Cipolla, “Multi-task learning using uncertainty to weigh losses for scene geometry and semantics,” in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
- [190] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: Synthetic minority over-sampling technique,” Journal of Artificial Intelligence Research, vol. 16, pp. 321–357, 2002.
- [191] C. Drummond and R. C. Holte, “C4.5, class imbalance, and cost sensitivity: Why under-sampling beats over-sampling,” in The International Conference on Machine Learning (ICML), 2003.
- [192] H. He, Y. Bai, E. A. Garcia, and S. Li, “Adasyn: Adaptive synthetic sampling approach for imbalanced learning,” in International Joint Conference on Neural Networks, 2008.
- [193] M. Buda, A. Maki, and M. A. Mazurowski, “A systematic study of the class imbalance problem in convolutional neural networks,” Neural Networks, vol. 106, pp. 249–259, 2018.

- [194] Y.-X. Wang, D. Ramanan, and M. Hebert, “Learning to model the tail,” in Advances in Neural Information Processing Systems (NIPS), 2017.
- [195] Y. Cui, Y. Song, C. Sun, A. G. Howard, and S. J. Belongie, “Large scale fine-grained categorization and domain-specific transfer learning,” in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
- [196] C. Huang, Y. Li, C. C. Loy, and X. Tang, “Learning deep representation for imbalanced classification,” in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
- [197] Y. Cui, M. Jia, T. Lin, Y. Song, and S. J. Belongie, “Class-balanced loss based on effective number of samples,” in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
- [198] M. P. Kumar, B. Packer, and D. Koller, “Self-paced learning for latent variable models,” in Advances in Neural Information Processing Systems (NIPS), 2010.
- [199] K. Kumar Singh and Y. Jae Lee, “Hide-and-seek: Forcing a network to be meticulous for weakly-supervised object and action localization,” in The IEEE International Conference on Computer Vision (ICCV), 2017.
- [200] Q. Dong, S. Gong, and X. Zhu, “Class rectification hard mining for imbalanced deep learning,” in The International Conference on Computer Vision (ICCV), 2017.
- [201] H.-S. Chang, E. Learned-Miller, and A. McCallum, “Active bias: Training more accurate neural networks by emphasizing high variance samples,” in Advances in Neural Information Processing Systems (NIPS), 2017.
- [202] K. Wei, R. Iyer, and J. Bilmes, “Submodularity in data subset selection and active learning,” in The International Conference on Machine Learning (ICML), pp. 1954–1963, 2015.
- [203] V. Kaushal, A. Sahoo, K. Doctor, N. R. Uppalapati, S. Shetty, P. Singh, R. K. Iyer, and G. Ramakrishnan, “Learning from less data: Diversified subset selection and active learning in image classification tasks,” arXiv, vol. 1805.11191, 2018.

- [204] O. Sener and S. Savarese, “Active learning for convolutional neural networks: A core-set approach,” in The International Conference on Learning Representations (ICLR), 2018.
- [205] K. Vodrahalli, K. Li, and J. Malik, “Are all training examples created equal? an empirical study,” arXiv, vol. 1811.12569, 2018.
- [206] V. Birodkar, H. Mobahi, and S. Bengio, “Semantic redundancies in image-classification datasets: The 10% you don’t need,” arXiv, vol. 1901.11409, 2019.
- [207] D. Mahajan, R. Girshick, V. Ramanathan, K. He, M. Paluri, Y. Li, A. Bharambe, and L. van der Maaten, “Exploring the Limits of Weakly Supervised Pretraining,” in The European Conference on Computer Vision (ECCV), 2018.
- [208] Y. Liu, T. Gao, and H. Yang, “Selectnet: Learning to sample from the wild for imbalanced data training,” arXiv, vol. 1905.09872, 2019.
- [209] S. S. Mullick, S. Datta, and S. Das, “Generative Adversarial Minority Oversampling,” arXiv, vol. 1903.09730, 2019.
- [210] X. Zhu, Y. Liu, Z. Qin, and J. Li, “Data augmentation in emotion classification using generative adversarial networks,” arXiv preprint arXiv:1711.00648, 2017.
- [211] C. Bowles, L. Chen, R. Guerrero, P. Bentley, R. Gunn, A. Hammers, D. A. Dickie, M. V. Hernández, J. Wardlaw, and D. Rueckert, “Gan augmentation: augmenting training data using generative adversarial networks,” arXiv preprint arXiv:1810.10863, 2018.
- [212] X. Zhang, Z. Fang, Y. Wen, Z. Li, and Y. Qiao, “Range loss for deep face recognition with long-tailed training data,” in The IEEE International Conference on Computer Vision (ICCV), 2016.
- [213] X. Yin, X. Yu, K. Sohn, X. Liu, and M. K. Chandraker, “Feature transfer learning for deep face recognition with long-tail data,” arXiv, vol. 1803.09014, 2018.

- [214] C. Huang, Y. Li, C. L. Chen, and X. Tang, “Deep imbalanced learning for face recognition and attribute prediction,” IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2019.
- [215] J. Hu, J. Lu, and Y. Tan, “Discriminative deep metric learning for face verification in the wild,” in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014.
- [216] S. Bell and K. Bala, “Learning visual similarity for product design with convolutional neural networks,” ACM Trans. on Graphics (SIGGRAPH), vol. 34, no. 4, 2015.
- [217] F. Schroff, D. Kalenichenko, and J. Philbin, “Facenet: A unified embedding for face recognition and clustering,” in Conference on Computer Vision and Pattern Recognition (CVPR), 2015.
- [218] E. Hoffer and N. Ailon, “Deep metric learning using triplet network,” in The International Conference on Learning Representations (ICLR), 2015.
- [219] W. Ge, “Deep metric learning with hierarchical triplet loss,” in The European Conference on Computer Vision (ECCV), 2018.
- [220] Y. Yuan, K. Yang, and C. Zhang, “Hard-aware deeply cascaded embedding,” in The IEEE International Conference on Computer Vision (ICCV), 2017.
- [221] B. Harwood, G. K. B. VijayKumarB., G. Carneiro, I. D. Reid, and T. Drummond, “Smart mining for deep metric learning,” in The IEEE International Conference on Computer Vision (ICCV), 2017.
- [222] Y. Cui, F. Zhou, Y. Lin, and S. Belongie, “Fine-grained categorization and dataset bootstrapping using deep metric learning with humans in the loop,” in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
- [223] H. O. Song, Y. Xiang, S. Jegelka, and S. Savarese, “Deep metric learning via lifted structured feature embedding,” in The IEEE Computer Vision and Pattern Recognition (CVPR), 2016.

- [224] C. Huang, C. C. Loy, and X. Tang, “Local similarity-aware deep feature embedding,” in Advances in Neural Information Processing Systems (NIPS), 2016.
- [225] Y. Duan, W. Zheng, X. Lin, J. Lu, and J. Zhou, “Deep adversarial metric learning,” in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.
- [226] Y. Zhao, Z. Jin, G.-j. Qi, H. Lu, and X.-s. Hua, “An adversarial approach to hard triplet generation,” in The European Conference on Computer Vision (ECCV), 2018.
- [227] W. Zheng, Z. Chen, J. Lu, and J. Zhou, “Hardness-aware deep metric learning,” in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
- [228] A. Yao, “Interactive object detection,” in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012.
- [229] C. Li, J. Yan, F. Wei, W. Dong, Q. Liu, and H. Zha, “Self-paced multi-task learning,” in AAAI Conference on Artificial Intelligence, 2017.
- [230] S. Liu, E. Johns, and A. J. Davison, “End-to-end multi-task learning with attention,” in The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019.
- [231] K. J. Dembczynski, W. Waegeman, W. Cheng, and E. Hüllermeier, “An exact algorithm for f-measure maximization,” in Advances in Neural Information Processing Systems (NeurIPS), 2011.
- [232] Z. C. Lipton, C. Elkan, and B. Naryanaswamy, “Optimal thresholding of classifiers to maximize f1 measure,” in Machine Learning and Knowledge Discovery in Databases, 2014.
- [233] S. Puthiya Parambath, N. Usunier, and Y. Grandvalet, “Optimizing f-measures by cost-sensitive classification,” in Advances in Neural Information Processing Systems (NeurIPS), 2014.

- [234] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, “Imagenet large scale visual recognition challenge,” International Journal of Computer Vision (IJCV), vol. 115, no. 3, pp. 211 – 252, 2015.
- [235] E. Gundogdu and A. A. Alatan, “Good features to correlate for visual tracking,” IEEE Transactions on Image Processing, vol. 27, no. 5, pp. 2526 – 2540, 2018.





APPENDIX A

IMBALANCE PROBLEMS IN OTHER DOMAINS

This appendix covers imbalance problems in other related domains in order to motivate the adaptation of methods from related domains to the object detection problem. We identify two problems that are closely related to object detection: image classification and metric learning, discussed in individual sections. In addition, the methods pertaining to the multi-task learning are discussed in the last section.

A.1 Image Classification

Image classification is the problem of assigning a category label for an image. This problem is closely related to the object detection problem since it is one of two tasks in object detection. For image classification problem, class imbalance has been extensively studied from very different perspectives (compared to other imbalance problems), and in this section, we will focus only on class imbalance.

A common approach is **Resampling the dataset** [190, 191, 192, 193, 144], including oversampling and undersampling to balance the dataset. While oversampling adds more samples from the under-represented classes, undersampling balances over the classes by ignoring some of the data from the over-represented classes. When employed naively, oversampling may suffer from overfitting because duplication of samples from the under-represented classes can introduce bias. Therefore, despite it means ignoring a portion of the training data, undersampling was preferable for non-deep-learning approaches [191]. On the other hand, Buda et al. [193] showed that deep neural networks do not suffer from overfitting under oversampling and in fact, better performance can be achieved compared to undersampling.

Over the years, methods more complicated than just duplicating examples have been developed. For example, Chawla et al. [190] proposed *Smote* as a new way of over-sampling by producing new samples as an interpolation of neighbouring samples. *Adasyn* [192], an extension of *Smote* to generate harder examples, aims to synthesize samples from the underrepresented classes. Li et al. [144] sample a mini-batch as uniform as possible from all classes also by restricting the same example and class to appear in the same order, which implies promoting the minority classes.

Another approach addressing class imbalance in image classification is **transfer learning** [194, 195]. For example, Wang et al. [194] design a model (i.e. meta-learner) to learn how a model evolves when the size of the training set increases. The meta-learner model is trained gradually by increasing the provided number of examples from the classes with a large number of examples. The resulting meta-model is able to transform another model trained with less examples to a model trained with more examples, which makes it useful to be exploited by an underrepresented class. Another study [195] adopted a different strategy during transfer learning: Firstly, the network is trained with the entire imbalanced dataset, and then the resulting network is fine-tuned by using a balanced subset of the dataset.

Weighting the loss function [196, 197] is yet another way of balancing the classes (we called it soft sampling in Chapter 3). Among these approaches, Huang et al. [196] use the inverse class frequency (i.e. $1/|C_i|$) to give more weight to under-represented classes. This is extended by class-balanced loss [197] to adopt inverse class frequency by employing a hyperparameter β as $w_i = (1 - \beta)/(1 - \beta^{|C_i|})$. When $\beta = 0$, the weight degenerates to 1; and $\beta \rightarrow 1$ makes the weight approximate $1/|C_i|$, the inverse class frequency.

Similar to object detection, giving more importance to “useful” examples is also common [198, 199, 200, 201]. As done in object detection, hard examples have been utilized, e.g. by Dong et al. [200] who identify hard (positive and negative) samples in the batch level. The proposed method uses the hardness in two levels: (i) Class-level hard samples are identified based on the predicted confidence for the ground-truth class. (ii) Instance-level hard examples are the ones with larger $L2$ distance to a representative example in the feature space. An interesting approach that has not been

utilized in object detection yet is to focus training on examples on which the classifier is uncertain (based on its prediction history) [201]. Another approach is to generate hard examples by randomly cropping parts of an image as in Hide and Seek [199], or adopting a curricular learning approach by first training the classifier on easy examples and then on hard examples [198].

Another related set of methods addresses image classification problem from the **data redundancy** perspective [202, 203, 204, 205, 206]. Birodkar et al. [206] showed that around 10% of the data in ImageNet [49] and CIFAR-10 datasets is redundant during training. This can be exploited not only for balancing the data but also to obtain faster convergence by ignoring useless data. The methods differ from each other on how they mine for the redundant examples. Regardless of whether imbalance problem is targeted or not, a subset of these methods uses the active learning paradigm, where an oracle is used to find out the best set of training examples. The core set approach [204] uses the relative distances of the examples in the feature space to determine redundant samples whereas Vodrahalli et al. [205] determine redundancies by looking at the magnitude of the gradients.

Another mechanism to enrich a dataset is to use **weak supervision** for incorporating unlabelled examples. An example study is by Mahajan et al. [207], who augment the dataset by systematically including Instagram images with the hashtags as the labels, which are rather noisy. In another example, Liu et al. [208] selectively add unlabeled data to the training samples after labeling these examples using the classifier.

Generative models (e.g. GANs) can also be used for extending the dataset to address imbalance. Many studies [209, 210, 211] have successfully used GANs to generate examples for under-represented classes for various image classification problems.

Special cases of image classification (e.g. face recognition) are also affected by imbalance [212, 213, 214]. The general approaches are similar and therefore, due to the space constraint, we omit imbalance in specialized classification problems.

Comparative Summary. Our analysis reveals that object detection community can benefit from the imbalance studies in image classification in many different aspects. The discussed methods for image classification are (by definition) the present solu-

tions for the foreground-foreground class imbalance problem (see Section 4.2); however, they can possibly be extended to the foreground-background class imbalance problem. Foreground-background class imbalance is generally handled by under-sampling for the object detection problem, and other advanced resampling or transfer learning methods are not adopted yet for object detection from a class imbalance perspective. While there are loss functions (discussed in Section 4.1.2) that exploit a weighting scheme [18, 68], Cui et al. [197] showed that class-balanced loss is complementary to the focal loss [18] in that focal loss aims to focus on hard examples while class-balanced loss implies balancing over all classes. However, since the number of background examples is not defined, the current definition does not fit into the object detection context. Similarly, the adoption of weakly supervised methods to balance under-represented classes or data redundancy by also decreasing the samples from an over-represented class can be used to alleviate the class imbalance problem. Finally, there are only a few generative approaches in object detection, much less than those proposed for addressing imbalance in image classification.

A.2 Metric Learning

Metric learning methods aim to find an embedding of the inputs, where the distance between the similar examples is smaller than the distance between dissimilar examples. In order to model such similarities, the methods generally employ pairs [215, 216] or triplets [217, 218, 219] during training. In the pair case, the loss function uses the information about whether both of the samples are from the same or different classes. In contrast, training using triplets require an anchor example, a positive example from the same class with the anchor and a negative example from a different class from the anchor. The triplet-wise training scheme introduces imbalance over positive and negative examples in favor of the latter one, and the methods also look for the hard examples as in object detection in both the pair-wise and triplet-wise training schemes. Accordingly, to present the imbalance and useful example mining requirement, note that there are approximately $O(n^2)$ pairs and $O(n^3)$ triplets assuming that the dataset size is n . This increase in the dataset size makes it impossible to mine for hard examples by processing the entire dataset to search for the most useful

(i.e. hard) examples. For this reason, similar to object detection, a decision is to be made on which samples to be used during training.

The metric learning methods use sampling, generative methods or novel loss functions to alleviate the problem. Note that this is very similar to our categorization of foreground-background class imbalance methods in Section 4.1, which also drove us to examine metric learning in detail. Owing to its own training, and resulting loss function configuration, here we only examine sampling strategies and generative methods.

Sampling Methods aim to find a useful set of training examples from a large training dataset. The usefulness criterion is considered to be highly relevant to the hardness of an example. Unlike the methods in object detection, some of the methods avoid using the hardest possible set of examples during training. One example proposed by Schroff et al. [217] use a rule based on Euclidean distance to define “semi-hard” triplets since selecting the hardest triplets can end up in local minima during the early stages of the training. This way, they decrease the effect of confusing triplets and avoid repeating the same hardest examples. Another approach that avoids considering the hardest possible set of examples is Hard-Aware Deeply Cascaded Embedding [220], which proposes training a cascaded model such that the higher layers are trained with harder examples while the first layers are trained with the entire dataset. Similarly, Smart Mining [221] also mines semi-hard examples exploiting the distance between nearest neighbour of anchor and the corresponding anchor and one novelty is that they increase the hardness of the negative examples in the latter epochs adaptively. Note that neither semi-hardness nor adaptive setting of the hardness level is considered by object detectors.

As a different approach, Cui et al. [222] consider to exploit humans to label false positives during training, which are identified as hard examples and be added to the mini-batch for the next iteration. Song et al. [223] mine hard negatives not only for the anchor but also for all of the positives. Note that no object detection method considers the relation between all positives and the negative example while assigning a hardness value to a negative example. One promising idea shown by Huang et al. [224] is that larger intra-class distances can confuse the hard example mining

process while only inter-class distance is considered during mining. For this reason, a position-dependent deep metric unit is proposed to take into account the intra-class variations.

Similar to the generative methods for object detection (Section 4.1.4), **Generative Methods** have been used for generating examples or features for metric learning as well. Deep Adversarial Metric Learning [225] simultaneously learns an embedding and a generator. Here, the generator outputs a synthetic hard negative example given the original triplet. Similar to Tripathi et al. [72], Zhao et al. [226] also use a GAN [143] in order to generate not only hard negatives but also hard positives. The idea to consider inter-class similarity have proven well as in the work by Huang et al. [224]. Finally, a different approach from the previous generative models, Hardness Aware Metric Learning [227], aims to learn an autoencoder in the feature space. The idea is as follows: The authors first manipulate the features after the backbone such that the hardness of the example can be controlled by linearly interpolating the embedding towards the anchor by employing a coefficient relying on the loss values at the last epoch. Since it is not certain that the interpolated embedding preserves the original label, a label preserving mapping back to the feature space is employed using the autoencoder. Also, similar to Harwood et al. [221], the hardness of the examples in the latter epochs is increased.

Comparative Summary. Looking at the studies presented above, we observe that the metric learning methods are able to learn an embedding of the data that preserves the desired similarity between data samples. Object detection literature have used different measures and metrics that have been designed by humans. However, as shown by the metric learning community, a metric that is directly learned from the data itself can yield better results and have interesting properties. Moreover, the self-paced learning, where the hardness levels of the examples is increased adaptively, is definitely an important concept for addressing imbalance in object detection. Another idea that can be adopted by the object detectors is to label the examples by humans in an online manner (similar to the work by Yao [228]) during training and to use the semi-hardness concept.

A.3 Multi-Task Learning

Multi-task learning involves learning multiple tasks (with potentially conflicting objectives) simultaneously. A common approach is to weigh the objectives of the tasks to balance them.

Many methods have been proposed for assigning the weights in a more systematic manner. For example, Li et al. [229] extended the self-paced learning paradigm to multi-task learning based on a novel regularizer. The hyperparameters in the proposed regularizer control the hardness of not only the instances but also the tasks, and accordingly, the hardness level is increased during training. In another work motivated by the self-paced learning approach, Guo et al. [139] use more diverse set of tasks, including object detection. Their method weighs the losses dynamically based on the exponential moving average of a predefined key performance indicator (e.g. accuracy, average precision) for each task. Similar to image classification, one can also use the uncertainty of the estimations [189] or their loss values [230] to assign weights to the tasks.

In addition to the importance or hardness of tasks, Zhao Chen and Rabinovich [188] identified the importance of the pace at which the tasks are learned. They suggested that the tasks are required to be trained in a similar pace. For this end, they proposed balancing the training pace of different tasks by adjusting their weights dynamically based on a normalization algorithm motivated by batch normalization [122].

Comparative Summary. Being a multi-task problem, object detection can benefit significantly from the multi-task learning approaches. However, this aspect of object detectors has not received attention from the community.



APPENDIX B

DETAILS OF THE BOUNDING BOX GENERATOR

This appendix the derivation of the Eq. 4.1 and Eq. 4.2, and explain the $\text{findBRFeasibleSpace}(B, T, \text{TL}(\bar{B}))$ function in Algorithm 1.

B.1 Details of Finding the Feasible Space for Top-Left Point

This section provides details for $\text{findTLFeasibleSpace}(B, T)$ function. In particular, we derive Eq. 4.1 and Eq. 4.2, and present all equations to determine the top-left space.

In order to derive Eq. 4.1 depicting x_{max}^I , we bound the x coordinate first. It is obvious that $x_{min}^I = x_1$ due to the boundary of Region I. For x_{max}^I , we know that $\bar{y}_1 = y_1$ again thanks to the region boundary. Therefore, since we have only one unknown, x_{max}^I , we use the definition of the IoU to determine its value in Eq. B.1-B.6. Eq. B.2 defines IoU based on Eq. B.1. In Eq. B.3, we set $\min(\bar{x}_2, x_2) = x_2$, $\max(\bar{x}_1, x_1) = x_{max}^I$, $\min(\bar{y}_2, y_2) = y_2$ and $\max(\bar{y}_1, y_1) = y_1$ by taking into the intersection definition in Region I. Also note that $\bar{x}_1 = x_{max}^I$, $\bar{y}_1 = y_1$, $\bar{x}_2 = x_2$ and

Table B.1: Top-Left space bounds and equations for Bounding Box Generator. See Fig. 4.4 for the regions.

Region	Min Bound	Max Bound	Equation
I	$\bar{x}_1 = x_1$	$\bar{x}_1 = x_2 - (x_2 - x_1) \times T$	$\bar{y}_1 = y_2 - \frac{\frac{A(B \cap \bar{B})}{T} + A(B \cap \bar{B}) - A(B)}{(x_2 - \bar{x}_1)}$
II	$\bar{y}_1 = y_1$	$\bar{y}_1 = y_2 - \frac{A(B) \times T}{x_2 - x_1}$	$\bar{x}_1 = x_2 - \frac{A(B \cap \bar{B}) \times A(B)}{(y_2 - \bar{y}_1)}$
III	$\bar{y}_1 = y_1$	$\bar{y}_1 = y_2 - \frac{A(B) \times T}{x_2 - x_1}$	$\bar{x}_1 = x_2 - \frac{\frac{A(B \cap \bar{B})}{T} - A(B) + A(B \cap \bar{B})}{(y_2 - \bar{y}_1)}$
IV	$\bar{y}_1 = \frac{(y_2 \times (T-1)) + y_1}{T}$	$\bar{y}_1 = y_1$	$\bar{x}_1 = x_2 - \frac{A(B)}{T \times (y_2 - \bar{y}_1)}$

$\bar{y}_2 = y_2$ in this case. In Eq. B.4-B.6, we just rearrange the terms to have x_{max}^I as a left hand side term.

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

$$= \frac{(\min(\bar{x}_2, x_2) - \max(\bar{x}_1, x_1)) \times (\min(\bar{y}_2, y_2) - \max(\bar{y}_1, y_1))}{(x_2 - x_1) \times (y_2 - y_1) + (\bar{x}_2 - \bar{x}_1) \times (\bar{y}_2 - \bar{y}_1) - A(B \cap \bar{B})} \quad (\text{B.2})$$

$$\Rightarrow T = \frac{(x_2 - x_{max}^I) \times (y_2 - y_1)}{(x_2 - x_1) \times (y_2 - y_1) + (x_2 - x_{max}^I) \times (y_2 - y_1) - (x_2 - x_{max}^I) \times (y_2 - y_1)} \quad (\text{B.3})$$

$$\Rightarrow (x_2 - x_1) \times (y_2 - y_1) \times T = (x_2 - x_{max}^I) \times (y_2 - y_1) \quad (\text{B.4})$$

$$\Rightarrow x_{max}^I = x_2 - \frac{(x_2 - x_1) \times (y_2 - y_1) \times T}{(y_2 - y_1)} \quad (\text{B.5})$$

$$\Rightarrow x_{max}^I = x_2 - (x_2 - x_1) \times T \quad (\text{B.6})$$

Now since we know the values of $\bar{x}_1$ based on the bounds, we can derive the Eq. 4.2 for any $\bar{y}_1$ value in equations by moving within bounds. Since $A(B \cap \bar{B}) = (x_2 - \bar{x}_1) \times (y_2 - y_1)$, it does not rely on $\bar{y}_1$ and we directly use $A(B \cap \bar{B})$ in the following equations:

$$\text{IoU}(B, \bar{B}) = \frac{A(B \cap \bar{B})}{A(B) + (x_2 - \bar{x}_1) \times (y_2 - \bar{y}_1) - A(B \cap \bar{B})} \quad (\text{B.7})$$

$$\Rightarrow T \times (x_2 - \bar{x}_1) \times (y_2 - \bar{y}_1) = A(B \cap \bar{B}) + T \times A(B \cap \bar{B}) - T \times A(B)$$

$$\Rightarrow \bar{y}_1 = y_2 - \frac{\frac{A(B \cap \bar{B})}{T} + A(B \cap \bar{B}) - A(B)}{(x_2 - \bar{x}_1)} \quad (\text{B.8})$$

Table B.1 presents all of the equations derived using the same methodology.

B.2 Details of Finding the Feasible Space for Bottom-Right Point

This section provides the details for `findBRFeasibleSpace(B, T, TL( $\bar{B}$ ))` function by following the same approach with the top left corner. However, different from top-left space this step is required also consider the point generated top-left point. Note that the size of the polygon in the bottom-right space is affected by the distance between $\text{TL}(\bar{B})$ and $\text{TL}(B)$. Maximum bottom-right polygon size, with exactly the same size of the top-left polygon, is achieved when $\text{TL}(\bar{B}) = \text{TL}(B)$. Conversely, bottom-right polygon degenerates to a point at $\text{BR}(B)$ if the sampled $\text{TL}(\bar{B})$ hits the border of the top-left polygon.

Table B.2: Bottom-Right space bounds for Bounding Box Generator.

Region	Min Bound	Max Bound
I	$\bar{y}_2 = \frac{T \times A(B) + T \times (x_2 - \alpha) \times \beta + \beta \times (x_2 - \alpha) - T \times \bar{y}_1 \times (x_2 - \bar{x}_1)}{((T+1) \times (x_2 - \alpha) - T \times (x_2 - \bar{x}_1))}$	$\bar{y}_2 = y_2$
II	$\bar{x}_2 = x_2$	$\bar{x}_2 = \bar{x}_1 + \frac{\frac{A(B \cap \bar{B})}{T} - A(B) + A(B \cap \bar{B})}{(y_2 - \bar{y}_1)}$
III	$\bar{y}_2 = y_2$	$\bar{y}_2 = \bar{y}_1 + \frac{\frac{A(B \cap \bar{B})}{T} - A(B) + A(B \cap \bar{B})}{(x_2 - \bar{x}_1)}$
IV	$\bar{x}_2 = \frac{T \times A(B) + T \times (y_2 - \beta) \times \alpha + \alpha \times (y_2 - \beta) - T \times \bar{x}_1 \times (y_2 - \bar{y}_1)}{((T+1) \times (y_2 - \beta) - T \times (y_2 - \bar{y}_1))}$	$\bar{x}_2 = x_2$

Table B.3: Bottom-Right space equations for Bounding Box Generator.

Region	Equation
I	$\bar{x}_2 = \bar{x}_1 + \frac{\frac{A(B \cap \bar{B})}{T} - A(B) + A(B \cap \bar{B})}{\bar{y}_2 - \bar{y}_1}$
II	$\bar{y}_2 = \bar{y}_1 + \frac{\frac{A(B \cap \bar{B})}{T} - A(B) + A(B \cap \bar{B})}{\bar{x}_2 - \bar{x}_1}$
III	$\bar{x}_2 = \frac{T \times A(B) + \alpha \times T \times (\hat{\beta} - \beta) + \alpha \times (\hat{\beta} - \beta) - T \times \bar{x}_1 \times (\bar{y}_2 - \bar{y}_1)}{(T+1) \times (\hat{\beta} - \beta) - T \times (\bar{y}_2 - \bar{y}_1)}$
IV	$\bar{y}_2 = \frac{T \times A(B) + \beta \times T \times (\hat{\alpha} - \alpha) + \beta \times (\hat{\alpha} - \alpha) - T \times \bar{y}_1 \times (\bar{x}_2 - \bar{x}_1)}{(T+1) \times (\hat{\alpha} - \alpha) - T \times (\bar{x}_2 - \bar{x}_1)}$

We add two additional parameters for the sake of clarity: $\alpha = \max(\bar{x}_1, x_1)$, $\beta = \max(\bar{y}_1, y_1)$, $\hat{\alpha} = \min(\bar{x}_2, x_2)$ and $\hat{\beta} = \min(\bar{y}_2, y_2)$. The bounds and the equations are derived by the same methodology that is illustrated in the first step presented in Tables B.2 and B.3 respectively.



APPENDIX C

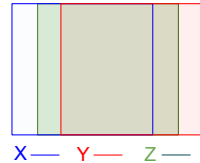
PROOF THAT PQ ERROR IS NOT A METRIC

This appendix proves that PQ Error (i.e. $(1-\text{PQ})$) is not a metric due to the fact that PQ Error violates triangle inequality¹:

Theorem 3. *PQ Error, defined by $1 - \text{PQ}(\mathcal{G}, \mathcal{D})$, violates triangle inequality, and hence it is not a metric.*

Proof. This is a proof by counter-example. If $1 - \text{PQ}(\mathcal{G}, \mathcal{D})$ satisfied triangle inequality, then we would expect $\forall \mathcal{A} \forall \mathcal{B} \forall \mathcal{C} \ 1 - \text{PQ}(\mathcal{A}, \mathcal{B}) \leq 1 - \text{PQ}(\mathcal{A}, \mathcal{C}) + 1 - \text{PQ}(\mathcal{C}, \mathcal{B})$. However, Fig. C.1 presents a counterexample, therefore PQ Error ($1 - \text{PQ}(\mathcal{G}, \mathcal{D})$) violates triangle inequality and it is not a metric. $\square$

¹ PQ satisfies the other two metricity conditions, i.e. reflexivity and symmetry.



Detection Mask	GT Mask	IoU	TP	FP	FN	PQ	1-PQ	LRP
X	Y	0.50	0	1	1	0	1	1
X	Z	0.71	1	0	0	0.71	0.29	0.58
Z	Y	0.71	1	0	0	0.71	0.29	0.58

Figure C.1: A counter-example which shows that PQ Error (i.e., $1-\text{PQ}$) violates triangle inequality. Hence, PQ Error is not a metric. (a) Three different inputs (i.e. masks) X , Y and Z . (b) $1 - \text{PQ}$ does not satisfy the triangle inequality (i.e. $1 - \text{PQ}(X, Y) > 1 - \text{PQ}(X, Z) + 1 - \text{PQ}(Z, Y)$), while LRP does (i.e. $\text{LRP}(X, Y) \leq \text{LRP}(X, Z) + \text{LRP}(Z, Y)$). See Chapter 2 for the notation.



APPENDIX D

PROOF THAT LRP ERROR IS A METRIC

This appendix proves that, unlike PQ Error (Appendix C), LRP Error is a metric if the localisation error is a metric (i.e. $1 - \text{lq}(\cdot, \cdot)$). In our proof, we obtain LRP Error using a reduction from Deficiency Aware Subpattern Assignment (DASA) performance metric [155] from point multitarget tracking literature. Note that DASA is a proven metric.

Theorem 4. *LRP is a metric.*

Proof. DASA metric is defined as:

$$\bar{e}_p^{(c)}(\mathcal{G}, \mathcal{D}) := \frac{l}{Z} \left(\frac{N_{\text{TP}}}{l} \left(\frac{1}{N_{\text{TP}}} \sum_{i=1}^{|\mathcal{G}|} \mathbb{I}[\text{d}(g_i, d_{g_i}) < c] \text{d}(g_i, d_{g_i})^p \right) + \left(\frac{c^p N_{\text{FP}}}{l} \right) + \left(\frac{c^p N_{\text{FN}}}{l} \right) \right)^{1/p}, \quad (\text{D.1})$$

where $l = \max(\mathcal{G}, \mathcal{D})$, $Z = N_{\text{TP}} + N_{\text{FP}} + N_{\text{FN}}$, $\mathbb{I}[\cdot]$ is the indicator function, $\text{d}(g_i, d_{g_i})$ is an arbitrary metric, c is the cut-off length to validate TPs (i.e. TP assignment threshold) based on $\text{d}(g_i, d_{g_i})$, and finally p is the lp-norm parameter (see Chapter 2 for the rest of the notation).

First, we set the lp-norm parameter, p , as 1 and simplify the definition:

$$\frac{1}{Z} \left(\left(\sum_{i=1}^{|\mathcal{G}|} \mathbb{I}[\text{d}(g_i, d_{g_i}) < c] \text{d}(g_i, d_{g_i}) \right) + c N_{\text{FP}} + c N_{\text{FN}} \right). \quad (\text{D.2})$$

Second, we incorporate the TP validation criterion of visual object detectors in Eq. D.2 as follows. A TP is identified if a ground truth, g_i , has a corresponding detection d_{g_i} such that $\text{lq}(g_i, d_{g_i}) > \tau$. Note that $\text{lq}(g_i, d_{g_i}) \in [0, 1]$. Then, to have a lower-better criterion which fits into Eq. D.2, we can rewrite this TP validation criterion

as $1 - \text{lq}(g_i, d_{g_i}) < 1 - \tau$. Having obtained the TP criterion, we set $d(g_i, d_{g_i}) = 1 - \text{lq}(g_i, d_{g_i})$ and $c = 1 - \tau$ in Eq. D.2:

$$\frac{1}{Z} \left(\left(\sum_{i=1}^{|\mathcal{G}|} \mathbb{I}[1 - \text{lq}(g_i, d_{g_i}) < 1 - \tau](1 - \text{lq}(g_i, d_{g_i})) \right) + (1 - \tau)N_{\text{FP}} + (1 - \tau)N_{\text{FN}} \right), \quad (\text{D.3})$$

which can be rewritten by simplifying the predicate of $\mathbb{I}[\cdot]$ as follows:

$$\frac{1}{Z} \left(\left(\sum_{i=1}^{|\mathcal{G}|} \mathbb{I}[\text{lq}(g_i, d_{g_i}) > \tau](1 - \text{lq}(g_i, d_{g_i})) \right) + (1 - \tau)N_{\text{FP}} + (1 - \tau)N_{\text{FN}} \right). \quad (\text{D.4})$$

Next, we just simplify Eq. D.4 in two steps: (i) We remove the Iverson Bracket by replacing $|\mathcal{G}|$ by N_{TP} in the summation,

$$\frac{1}{Z} \left(\left(\sum_{i=1}^{N_{\text{TP}}} (1 - \text{lq}(g_i, d_{g_i})) \right) + (1 - \tau)N_{\text{FP}} + (1 - \tau)N_{\text{FN}} \right), \quad (\text{D.5})$$

and (ii) finally, noting that dividing by a constant does not violate metricity, in order to ensure the upper bound to be 1 and facilitate the interpretation of LRP, we divide Eq. D.5 by $1 - \tau$:

$$\frac{1}{Z} \left(\sum_{i=1}^{N_{\text{TP}}} \frac{1 - \text{lq}(g_i, d_{g_i})}{1 - \tau} + N_{\text{FP}} + N_{\text{FN}} \right) = \text{LRP}(\mathcal{G}, \mathcal{D}). \quad (\text{D.6})$$

To conclude, LRP can be reduced from DASA, a proven metric, and therefore LRP is a metric. $\square$

APPENDIX E

WEIGHTING THE COMPONENTS OF THE LRP ERROR FOR PRACTICAL NEEDS OF DIFFERENT APPLICATIONS

LRP Error does not give priority to any of the performance aspects (FP rate, FN rate and localisation error) and weights each performance aspect by considering their maximum possible contribution to the total matching error (Section 5.5.1). On the other hand, depending on the requirements in a given application, one of the performance aspects can be given an emphasis. To illustrate a use-case, an online video object detector may want to gather as much detections as it can after discarding the “noisy” examples of the still image detector. Note that removing the noisy examples still requires some thresholding, however, the conventional LRP-Optimal threshold, balancing the contribution of FPs and FNs, may not be the best solution to fulfill this requirement, and a lower threshold can be more suitable. In a different use-case, different weights for the components can also be beneficial for evaluation as well. For example, a ballistic missile detector may not tolerate FNs but can accept more FPs errors. To this end, Eq. E.1 presents a weighted form of LRP Error:

$$\frac{1}{Z} \left(\sum_{i=1}^{N_{TP}} \alpha_{TP} \frac{1 - \text{Iq}(g_i, d_{g_i})}{1 - \tau} + \alpha_{FP} N_{FP} + \alpha_{FN} N_{FN} \right), \quad (\text{E.1})$$

where $Z = \alpha_{TP} N_{TP} + \alpha_{FP} N_{FP} + \alpha_{FN} N_{FN}$, and α_{TP} , α_{FP} and α_{FN} correspond to the “importance weights” of each performance aspect. Following the interpretation of LRP (see Section 5), the importance weights imply duplicating each error by the value of this weight. Accordingly, they are included both in the “total matching error” (i.e. nominator) and the “maximum possible value of the total matching error” (i.e. normalisation constant). In order to increase the contribution of a component, the importance weight of the desired component is to be set larger than 1 (e.g. to double the contribution of false negatives, then $\alpha_{FN} = 2$ and $\alpha_{TP} = \alpha_{FP} = 1$). Note that

when $\alpha_{TP} = \alpha_{FP} = \alpha_{FN} = 1$, Eq. E.1 reduces to the conventional definition of LRP (Eq. 5.2). Finally, we note that this modification naturally violates the symmetry of the metric properties when $\alpha_{FP} \neq \alpha_{FN}$.



APPENDIX F

WHY AVERAGE LRP (ALRP) IS NOT AN IDEAL PERFORMANCE MEASURE?

This appendix discusses why the recently proposed loss function aLRP Loss [156] is not an ideal performance measure.

Having a similar intuition to oLRP, we define aLRP Error by averaging the LRP Errors over the confidence scores (see Chapter 2 for the notation):

$$\text{aLRP} := \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \text{LRP}(\mathcal{G}, \mathcal{D}_s). \quad (\text{F.1})$$

where $\mathcal{D}_s$ is the set of detections thresholded at confidence score s (i.e. those detections with larger confidence scores than s are kept, and others are discarded). However, without any improvement in the detection performance, aLRP Error can be reduced to oLRP Error with the following two steps:

1. Delete all the detections with $s < s^*$ from the detection output,
2. Set the confidence score of the remaining detections to 1.00.

This two-step simple algorithm will make the s-LRP curve to be a line determined by $s = s^*$ (see Fig. 5.3(c)), and averaging over the confidence scores will yield oLRP. As a result, considering the fact that the performance with respect to aLRP is affected without any improvement in the detection performance, we do not prefer aLRP Error as a performance measure.



APPENDIX G

THE SIMILARITY BETWEEN PQ AND LRP ERRORS

This appendix derives Eq. 5.8:

$$1 - \text{PQ} = \frac{1}{\hat{Z}} \left(\sum_{i=1}^{N_{\text{TP}}} \frac{1 - \text{lq}(g_i, d_{g_i})}{1 - 0.50} + N_{\text{FP}} + N_{\text{FN}} \right), \quad (\text{G.1})$$

where $\hat{Z} = 2N_{\text{TP}} + N_{\text{FP}} + N_{\text{FN}}$. LRP and PQ Errors are very similar: Removing 2 (in red) from $\hat{Z}$ yields $1 - \text{PQ} = \text{LRP}$.

Recall from 5.1 that PQ is defined as (see Chapter 2 for the notation):

$$\text{PQ}(\mathcal{G}, \mathcal{D}) = \frac{1}{N_{\text{TP}} + \frac{1}{2}N_{\text{FP}} + \frac{1}{2}N_{\text{FN}}} \left(\sum_{i=1}^{N_{\text{TP}}} \text{IoU}(g_i, d_{g_i}) \right). \quad (\text{G.2})$$

First, we replace $\text{IoU}(\cdot, \cdot)$ in Eq. G.2 by $\text{lq}(\cdot, \cdot)$ to align the definitions of LRP and PQ:

$$\text{PQ}(\mathcal{G}, \mathcal{D}) = \frac{1}{N_{\text{TP}} + \frac{1}{2}N_{\text{FP}} + \frac{1}{2}N_{\text{FN}}} \left(\sum_{i=1}^{N_{\text{TP}}} \text{lq}(g_i, d_{g_i}) \right). \quad (\text{G.3})$$

Then, just by simple algebraic operations, we manipulate Eq. G.3:

$$PQ = 1 - 1 + \frac{\sum_{i=1}^{N_{TP}} \text{lq}(g_i, d_{g_i})}{N_{TP} + \frac{1}{2}N_{FP} + \frac{1}{2}N_{FN}} \quad (\text{G.4})$$

$$= 1 - \left(1 - \frac{\sum_{i=1}^{N_{TP}} \text{lq}(g_i, d_{g_i})}{N_{TP} + \frac{1}{2}N_{FP} + \frac{1}{2}N_{FN}} \right) \quad (\text{G.5})$$

$$= 1 - \frac{N_{TP} + \frac{1}{2}N_{FP} + \frac{1}{2}N_{FN} - \sum_{i=1}^{N_{TP}} \text{lq}(g_i, d_{g_i})}{N_{TP} + \frac{1}{2}N_{FP} + \frac{1}{2}N_{FN}} \quad (\text{G.6})$$

$$= 1 - \frac{N_{TP} - \sum_{i=1}^{N_{TP}} \text{lq}(g_i, d_{g_i}) + \frac{1}{2}N_{FP} + \frac{1}{2}N_{FN}}{N_{TP} + \frac{1}{2}N_{FP} + \frac{1}{2}N_{FN}} \quad (\text{G.7})$$

$$= 1 - \frac{\sum_{i=1}^{N_{TP}} 1 - \sum_{i=1}^{N_{TP}} \text{lq}(g_i, d_{g_i}) + \frac{1}{2}N_{FP} + \frac{1}{2}N_{FN}}{N_{TP} + \frac{1}{2}N_{FP} + \frac{1}{2}N_{FN}} \quad (\text{G.8})$$

$$= 1 - \frac{\sum_{i=1}^{N_{TP}} (1 - \text{lq}(g_i, d_{g_i})) + \frac{1}{2}N_{FP} + \frac{1}{2}N_{FN}}{N_{TP} + \frac{1}{2}N_{FP} + \frac{1}{2}N_{FN}} \quad (\text{G.9})$$

$$= 1 - \frac{1}{\hat{Z}} \left(\sum_{i=1}^{N_{TP}} \frac{1 - \text{lq}(g_i, d_{g_i})}{1 - 0.50} + N_{FP} + N_{FN} \right), \quad (\text{G.10})$$

where $\hat{Z} = 2N_{TP} + N_{FP} + N_{FN}$. As a result, we can rewrite Equation G.10 to express the PQ Error (i.e. 1-PQ) as follows:

$$1 - PQ = \frac{1}{\hat{Z}} \left(\sum_{i=1}^{N_{TP}} \frac{1 - \text{lq}(g_i, d_{g_i})}{1 - 0.50} + N_{FP} + N_{FN} \right) \quad (\text{G.11})$$

APPENDIX H

MORE EXPERIMENTS WITH LRP ERROR

In addition to the experiments Chapter 5, this appendix demonstrates a use-case of LRP-Optimal Thresholds, and analyse the latency of LRP computation and the effect of TP validation threshold, τ , on LRP.

H.1 A Use-Case of LRP-Optimal Thresholds in Video Object Detection

Related Work on Setting the thresholds of the classifiers. Employing visual detectors for a practical application requires a confidence threshold that balances precision, recall and localisation performance since, otherwise, the resulting output would be dominated by several false positives with low confidence scores. However, this topic has not received much attention from the research community and is usually handled by practitioners in a problem or deployment-specific manner. Conventionally, the thresholds of classifiers are set by finding the optimal F-measure on the PR curve or G-mean on the receiver operating characteristics curve, which do not consider localisation quality. Prior work on setting the classifier thresholds focused on the probabilistic models [231, 232]. Parambath et al. [233] present a theoretical analysis of the F-measure, and propose a practical algorithm discretizing the confidence scores in order to search for the optimal F-measure. Currently, for object detection, a general single threshold is used for all classes instead of class-specific thresholds [159].

The Experimental Setup. Here, we demonstrate a use-case where oLRP helps us to set class-specific optimal thresholds as an alternative to the naive approach of using a general, class-independent threshold. To this end, we develop a simple, online video

object detection framework where we use an off-the-shelf still-image object detector (RetinaNet-50 [18] trained on COCO [13]) and built three different versions of the video object detector. The first version, denoted with B , uses the still-image object detector to process each frame of the video independently. The second and third versions, denoted with G and S , respectively, again use the still-image object detector to process each frame and in addition, they link bounding boxes across subsequent frames using the Hungarian matching algorithm [151] and update the scores of these linked boxes using a simple Bayesian rule (details of this simple online video object detector is given below). The only difference between G and S is that while G uses a validated threshold of 0.50 (see Fig. 5.10(b) to notice that the LRP-Optimal Threshold distribution of RetinaNet has a mean around 0.50) as the confidence score threshold for all classes, S uses LRP-Optimal Threshold per class. We test these three detectors on 346 videos of ImageNet VID validation set [234] for 15 object classes which also happen to be included in COCO.

Details of the Online Video Object Detectors. There are two online video object detectors: G and S which respectively use the general, class-independent thresholding approach with 0.50 as threshold and the class-specific thresholds. For each of the online detectors, at each time interval, the detections from the previous and current frames are associated using the Hungarian algorithm [151] considering a box linking function and the confidence scores of associated BBs of the current frame are updated using the score distributions from both frames. Since an online tracker, specifically [235], is also used in our method, we use the L1 norm of the difference of confidence score distributions of neighbouring frames and the IoU overlap of the tracker prediction and the detection at current frame. While choosing this box linking function, we inspired from the tube linking score of [158]. The updated score is estimated using the Bayes Theorem such that the prior is the updated tubelet score in the previous frame and likelihood is the currently associated high confidence detection with that tubelet. In such an update method, even though the updated scores converge to 1 quickly, which is harmful for lower recall, precision improves in larger recall portions. Also, we call a BB as “dominant object” if its updated score increases by 0.20. In order to increase the recall, the disappearance of a “dominant object” is closely inspected by using the tracker again to predict the possible location, then the cropped region is

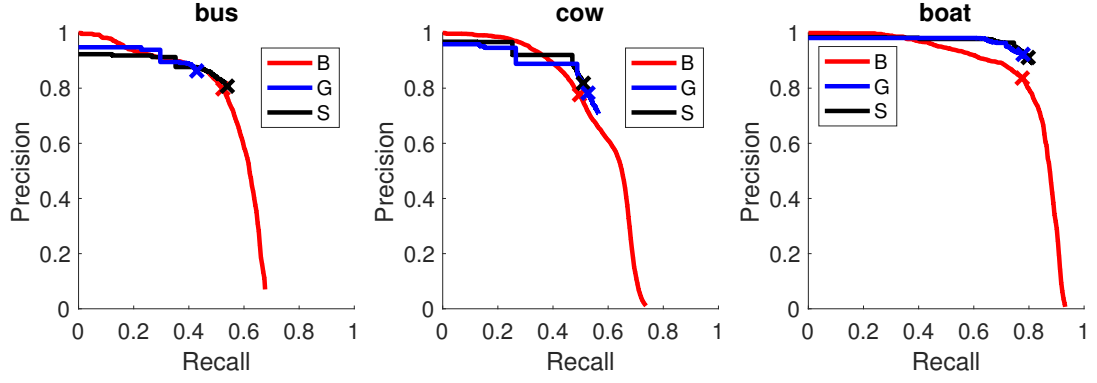


Figure H.1: Example PR curves of the methods on three example classes. Optimal PR pairs are marked with crosses. Using a LRP-Optimal Threshold balances FP and FN errors resulting in a stretched PR curve either in recall (e.g.(a)) or precision (e.g.(b)) axis depending on the chosen general threshold. Furthermore, using AP for these pruned PR curves does not provide consistent performance evaluation.

classified by class-wise binary classifiers (object vs. background).

AP vs. oLRP for Video Object Detection: We compare G with B in order to represent the evaluation perspectives of AP and oLRP – see Fig. H.1 and Table H.1. Since B is a conventional object detector, with conventional PR curves as illustrated in Fig. H.1. On the other hand, in order to be faster, G ignores some of the detections causing its maximum recall to be less than that of B . Thus, these shorter ranges in the recall set a big problem in the evaluation with respect to AP. Quantitatively, B surpasses G by 7.5% AP. On the other hand, despite limited recall coverage, G obtains higher precision than B especially through the end of its PR curve. To illustrate, for the “boat” class in Fig. H.1, G has significantly better precision after approximately between 0.5 and 0.9 recall even though its AP is lower by 6%. Since oLRP compares methods concerning their best configurations, this difference is clearly addressed comparing their oLRP error in which G surpasses S by 4.1%. Furthermore, the superiority of G is shown to be its higher precision since FN components of G and S are very close while FP component of G is 8.6% better, which is also the exact difference of precisions in their peaks of PR curves.

Therefore, while G seems to have very low performance in terms of AP, for 12 classes G reaches better peaks than B as illustrated by the oLRP values in Table H.1. This

Table H.1: Comparison among B , G , S with respect to AP & oLRP and their best class-specific configurations. The mean of class thresholds are assigned as N/A since the thresholds are set class-specific and the mean is not used. s^* denotes the LRP-Optimal Thresholds. Note that unlike AP, lower scores are better for LRP.

	Method	airplane	bicycle	bird	bus	car	cow	dog	cat	elephant	horse	motorcycle	sheep	train	boat	zebra	mean
AP ₅₀	B	68.1	63.0	54.7	56.5	55.5	58.7	46.3	60.1	66.1	47.3	60.2	56.1	71.3	82.9	81.6	61.9
	G	62.1	44.5	49.2	39.8	41.7	51.0	41.6	56.8	58.8	44.1	57.1	54.7	60.0	76.9	76.5	54.4
	S	64.5	53.5	50.0	48.5	41.9	49.2	43.4	56.9	58.9	44.4	57.3	54.5	60.9	79.2	78.2	56.1
oLRP	B	62.7	77.6	71.8	70.2	75.9	69.2	72.8	70.0	62.5	72.3	69.2	67.7	58.3	59.4	43.6	66.9
	G	60.6	78.3	69.1	72.7	75.8	67.9	71.4	69.7	61.4	69.9	65.4	64.8	58.6	55.3	43.2	65.6
	S	60.3	76.2	68.7	68.8	75.9	67.8	71.2	69.7	61.3	70.1	65.5	64.9	58.3	55.1	42.5	65.1
oLRP _{Loc}	B	18.2	27.1	16.9	17.7	20.7	14.5	16.6	20.3	17.0	15.5	19.2	15.4	15.9	19.9	12.8	17.9
	G	18.1	25.8	17.0	16.0	20.7	15.1	16.5	20.0	17.0	16.0	19.5	15.5	15.6	19.5	12.8	17.7
	S	18.6	27.0	17.0	17.3	20.7	14.8	17.0	20.0	17.0	16.0	19.4	15.5	15.9	19.7	13.1	17.9
oLRP _{FP}	B	8.0	22.8	30.0	20.3	30.3	22.4	24.2	24.8	9.5	24.6	15.8	14.1	9.9	16.3	3.4	18.4
	G	8.6	11.6	17.4	13.7	31.1	21.8	22.9	27.9	7.1	22.1	4.9	7.8	9.1	7.7	1.6	14.2
	S	8.7	22.6	18.4	19.3	32.0	18.2	26.9	28.3	7.5	23.1	8.4	7.8	11.0	8.9	3.0	16.3
oLRP _{FN}	B	38.3	42.7	47.8	47.7	49.9	50.4	53.3	39.4	39.5	54.0	44.8	49.4	34.4	22.4	22.0	42.4
	G	35.9	52.3	48.0	57.1	49.3	47.3	51.2	37.2	38.8	49.4	41.5	46.7	36.0	22.1	22.7	42.4
	S	32.6	38.9	48.9	46.1	48.8	49.0	48.0	36.9	38.5	49.3	40.6	46.8	33.9	20.3	20.2	39.8
s^*	B	0.38	0.31	0.44	0.27	0.49	0.61	0.42	0.49	0.49	0.52	0.45	0.51	0.41	0.45	0.31	N/A
	G	0.00	0.69	0.97	0.68	0.00	0.96	0.48	0.70	0.33	0.64	0.60	0.84	0.59	0.90	0.00	N/A
	S	0.00	0.54	0.98	0.45	0.00	0.91	0.49	0.64	0.39	0.58	0.63	0.85	0.55	0.89	0.54	N/A

suggests that oLRP is better than AP in capturing the performance details of this kind methods that uses thresholding.

Effect of the Class-specific LRP-Optimal Thresholds: Compared to G , owing to the class-specific thresholds, S has 1.7% better AP and 0.5% better oLRP as shown in Table H.1. However, since the mean is dominated by s^* around 0.50, it is better to focus on classes with low or high s^* values in order to grasp the effect of the approach. The “bus” class has the lowest s^* with 27%. For this class, S surpasses G by 8.7% in AP and 3.9% in oLRP. This performance increase is also observed for other classes with very low thresholds, such as “airplane”, “bicycle” and “zebra”. For these classes with lower thresholds, the effect of LRP-Optimal threshold on the PR curve is to stretch the curve in the recall domain (maybe by accepting some loss in precision) as shown in the “bus” example in Fig. H.1. Not surprisingly, “cow” is one of the two classes for which AP of S is lower since its threshold is the highest and thereby causing recall to be more limited. On the other hand, regarding oLRP, the result is not worse since this time the PR curve is stretched through the positive precision, as shown in Fig. H.1, allowing better FP errors. Thus, in any case, lower or higher, the LRP-Optimal Threshold aims to discover the best PR curve. There are four classes in total for which G is better than S in terms of oLRP. However, note that the maximum difference is 0.2% in oLRP and these are the classes with thresholds around 0.5. These suggest that choosing class-specific thresholds rather than the general, class-independent thresholding approach increases the performance of the detector especially for classes with low or high class-specific thresholds.

H.2 Analysing Latency of LRP Computation

To evaluate soft predictions and hard predictions using LRP, we incorporated the corresponding LRP variant into the official COCO evaluation [13] and panoptic segmentation repositories [150] respectively. Since the PQ and LRP are very similar in formulation (Section 5.6), it is obvious that they require very similar time to compute. Hence, in this section we focus on how much time computing LRP adds to the AP computation.

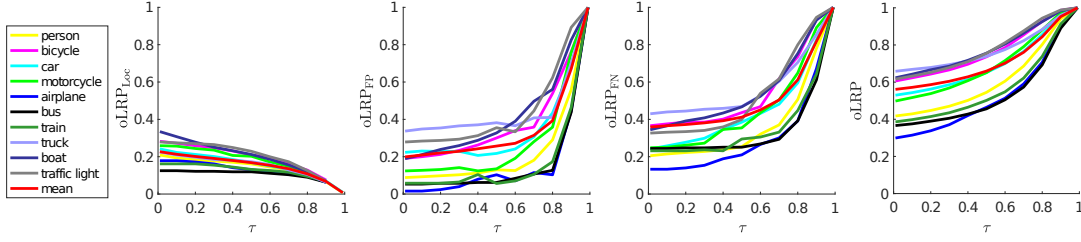


Figure H.2: For each class, oLRP and its components for Faster R-CNN (X101-12) are plotted against τ . The mean represents the mean of 80 classes.

During the computation of AP, COCO evaluation follows a five-step algorithm: (i) loading annotations into memory, (ii) loading and preparing results, (iii) per image evaluation, (iv) accumulating evaluation results, and (v) summarizing (i.e. printing) the results. Since step (i) and (ii) are independent of the performance measure, and (v) is a simple printing operation (i.e. it takes less than 3 seconds compute (i), (ii) and (v) in total) and we do not change (iii) per image evaluation except returning the computed IoUs of TPs, we analysed the additional latency of computing oLRP for step (iv) accumulation, in which the per-image evaluation results are combined into performance values. In order to do that using a standard CPU, we computed and averaged the runtime of this step using all 32 SOTA models in Table 5.4 on COCO *minival* with 5000 images. We observed that LRP computation (including LRP, oLRP, their components, class-specific LRP-Optimal Thresholds for different “size” and “maximum detection number” criteria as done by COCO toolkit - see [13] for details) introduces a negligible overhead with around one second both for (iv) accumulate step (from 5.8 seconds to 6.6 seconds) and for entire computation (from 38.7 to 39.6 seconds).

H.3 Analyzing the Effect of TP Validation Threshold

Finally, we analyse how LRP is affected from the TP validation threshold parameter, τ . We use Faster R-CNN (X101-12) (Table 5.4) results of the first 10 classes and mean-error for clarity, the effect of the τ parameter is analysed in Fig. H.2 on oLRP. As expected, larger τ values imply lower localisation error (oLRP_{Loc}). On the other hand, a larger τ causes FP and FN components to increase rapidly, leading to higher

total error (oLRP). This is intuitive since at the extreme case, i.e., when $\tau = 1$, there are hardly any TP (i.e. all the detections are FPs), which makes oLRP to be 1. Therefore, LRP allows measuring the performance of a detector designed for an application that requires a different τ by also providing additional information. In addition, investigating oLRP for different τ values represents a good extension for ablation studies.





APPENDIX I

A GENERALISATION OF ERROR-DRIVEN OPTIMISATION FOR RANKING-BASED LOSSES

This appendix presents the algorithm that we proposed in our previous work [156] and used to obtain the gradients of aLRP Loss (Chapter 7). Since Chapter 6 provides a more general, simple and interpretable version of this algorithm coined as “Identity Update”, this part of our paper was not discussed in Chapter 7 for clarity, but still, we reserve this appendix to discuss our generalisation algorithm for interested readers considering the fact that aLRP Loss is defined and optimised based on the algorithm presented in this appendix. We again note that both of our optimisation algorithms provide the equal gradient values while Identity Update being more interpretable, simple and general. We highlight the differences of these optimisation algorithms using footnotes in this appendix.

Given a ranking-based loss function¹,

$$\mathcal{L} = \frac{1}{Z} \sum_{i \in \mathcal{P}} \ell(i), \quad (\text{I.1})$$

defined as a sum over individual losses, $\ell(i)$, at positive examples (e.g., Eq. 6.2), with Z as a problem specific normalization constant, our goal is to express $\mathcal{L}$ as a sum of *primary terms* in a more general form than Eq. 6.12:

Definition 3. The *primary term*² L_{ij} concerning examples $i \in \mathcal{P}$ and $j \in \mathcal{N}$ is the loss originating from i and distributed over j via a probability mass function $p(j|i)$.

¹ In comparison to the loss definition of Identity Update (Eq. 6.17), this loss formulation does not consider the target value, hence not easily interpretable. Also, it is limited to the errors defined on positive examples.

² In comparison to Identity Update, this primary term definition does not consider interclass errors such as the errors among positives in our RS Loss.

Formally,

$$L_{ij} = \begin{cases} \ell(i)p(j|i), & \text{for } i \in \mathcal{P}, j \in \mathcal{N} \\ 0, & \text{otherwise.} \end{cases} \quad (\text{I.2})$$

Then, as desired, we can express $\mathcal{L} = \frac{1}{Z} \sum_{i \in \mathcal{P}} \ell(i)$ in terms of L_{ij} :

Theorem 5. $\mathcal{L} = \frac{1}{Z} \sum_{i \in \mathcal{P}} \ell(i) = \frac{1}{Z} \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} L_{ij}.$

Proof. The ranking function is defined as:

$$\mathcal{L} = \frac{1}{Z} \sum_{i \in \mathcal{P}} \ell(i). \quad (\text{I.3})$$

Since $\forall i \sum_{j \in \mathcal{N}} p(j|i) = 1$, we can rewrite the definition as follows:

$$\frac{1}{Z} \sum_{i \in \mathcal{P}} \ell(i) \left(\sum_{j \in \mathcal{N}} p(j|i) \right). \quad (\text{I.4})$$

Reorganizing the terms concludes the proof as follows:

$$\frac{1}{Z} \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} \ell(i)p(j|i) = \frac{1}{Z} \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} L_{ij}. \quad (\text{I.5})$$

□

Eq. I.2 makes it easier to define primary terms and adds more flexibility on the error distribution: e.g., AP Loss takes $p(j|i) = H(x_{ij})/N_{\text{FP}}(i)$, which distributes error uniformly (since it is reduced to $1/N_{\text{FP}}(i)$) over $j \in \mathcal{N}$ with $\hat{s}_i \geq \hat{s}_j$; though, a skewed $p(j|i)$ can be used to promote harder examples (i.e. larger x_{ij}). Here, $N_{\text{FP}}(i) = \sum_{j \in \mathcal{N}} H(x_{ij})$ is the number of false positives for $i \in \mathcal{P}$.

Now we can identify the gradients of this generalized definition similar to Chen et al. (Section 6.1): The update in x_{ij} that would minimize $\mathcal{L}$ is (Eq. 6.16),

$$\Delta x_{ij} = -(L_{ij}^* - L_{ij}), \quad (\text{I.6})$$

where L_{ij}^* denotes “the primary term when i is ranked properly”. Note that L_{ij}^* , which is set to zero in AP Loss, needs to be carefully defined (see Appendix L for a bad example). With Δx_{ij} defined, the gradients can be derived using Eq. 6.15. The steps for obtaining the gradients of $\mathcal{L}$ are summarized in Algorithm 4.

Algorithm 4 Obtaining the gradients of a ranking-based function with error-driven update.

Input: A ranking-based function $\mathcal{L} = (\ell(i), Z)$, and a probability mass function $p(j|i)$

Output: The gradient of $\mathcal{L}$ with respect to model output $\mathbf{s}$

- 1: $\forall i, j$ find primary term: $L_{ij} = \ell(i)p(j|i)$ if $i \in \mathcal{P}, j \in \mathcal{N}$; otherwise $L_{ij} = 0$ (c.f. Eq. I.2).
 - 2: $\forall i, j$ find target primary term: $L_{ij}^* = \ell(i)^*p(j|i)$ ($\ell(i)^*$: the error on i when i is ranked properly.)
 - 3: $\forall i, j$ find error-driven update: $\Delta x_{ij} = -(L_{ij}^* - L_{ij}) = -(\ell(i)^* - \ell(i))p(j|i)$ (c.f. Eq. 6.16).
 - 4: **return** $\frac{1}{Z}(\sum_j \Delta x_{ji} - \sum_j \Delta x_{ij})$ for each $\hat{s}_i \in \hat{\mathbf{s}}$ (c.f. Eq. 6.15).
-



APPENDIX J

DEFINING AND OPTIMISING ALRP LOSS USING IDENTITY UPDATE

Chapter 7 uses our “A Generalisation of Error-Driven Optimisation for Ranking-Based Losses” (Appendix I) to define and optimise aLRP Loss. This appendix presents how aLRP Loss can be defined and its gradients can be obtained using our “Identity Update” (Section 6.2), which is not only more general and simple-to-use, but also provides more interpretable loss values. We note that the gradient values obtained using either of our optimisation algorithms are equal, thus the results will not change.

Definition of aLRP Loss using Identity Update: We use the form of generic loss formulation (Eq. 6.17) of Identity Update to define aLRP Loss as follows (compare with Eq. 7.2 to see that Identity Update additionally requires the target, $\ell_{\text{LRP}}^*(i)$, and measures the error between current and target losses to yield an interpretable loss value):

$$\mathcal{L}^{\text{aLRP}} = \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} (\ell_{\text{LRP}}(i) - \ell_{\text{LRP}}^*(i)), \quad (\text{J.1})$$

where $\ell_{\text{LRP}}(i)$ is the LRP Error computed on example i (c.f. Eq. 7.3):

$$\ell_{\text{LRP}}(i) = \frac{1}{\text{rank}(i)} \left(N_{\text{FP}}(i) + \mathcal{E}_{\text{loc}}(i) + \sum_{k \in \mathcal{P}, k \neq i} \mathcal{E}_{\text{loc}}(k) H(x_{ik}) \right), \quad (\text{J.2})$$

and the target LRP Error is used as (c.f. Eq. 7.6):

$$\ell_{\text{LRP}}(i)^* = \frac{\mathcal{E}_{\text{loc}}(i)}{\text{rank}(i)}, \quad (\text{J.3})$$

which completes the definition of aLRP Loss based on Identity Update (please refer to Chapter 2 for the notation).

Optimisation of aLRP Loss using Identity Update: Computation and optimisation of a ranking-based loss function using Identity Update requires identifying primary

terms defined in Eq. 6.18. For aLRP Loss, we obtain the following primary term:

$$L_{ij} = \begin{cases} (\ell_{\text{LRP}}(i) - \ell_{\text{LRP}}(i)^*) p(j|i), & \text{for } i \in \mathcal{P}, j \in \mathcal{N} \\ 0, & \text{otherwise,} \end{cases} \quad (\text{J.4})$$

where, $p(j|i) = \frac{H(x_{ij})}{N_{\text{FP}}(i)}$ distributing the error over negatives with larger logits uniformly. Using the definitions of $\ell_{\text{LRP}}(i)$ (Eq. J.2) and $\ell_{\text{LRP}}(i)^*$ (Eq. J.3), Eq. J.4 can be expressed as:

$$L_{ij} = \begin{cases} \frac{1}{\text{rank}(i)} \left(N_{\text{FP}}(i) + \sum_{k \in \mathcal{P}, k \neq i} \mathcal{E}_{\text{loc}}(k) H(x_{ik}) \right) \frac{H(x_{ij})}{N_{\text{FP}}(i)}, & \text{for } i \in \mathcal{P}, j \in \mathcal{N} \\ 0, & \text{otherwise,} \end{cases} \quad (\text{J.5})$$

which is equal to the update in difference transform identified in Chapter 7 (c.f. Eq. 7.7), simply concluding the derivation.

APPENDIX K

DETAILS OF ALRP LOSS

This appendix provides details for aLRP Loss.

K.1 A Soft Sampling Perspective for aLRP Localisation Component

In sampling methods, the contribution (w_i) of the i th bounding box to the loss function is adjusted as follows (Chapter 3):

$$\mathcal{L} = \sum_{i \in \mathcal{P} \cup \mathcal{N}} w_i \mathcal{L}(i), \quad (\text{K.1})$$

where $\mathcal{L}(i)$ is the loss of the i th example. 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. While this perspective is quite common to train the classification branch [33, 18]; the localisation branch is conventionally trained by hard sampling with some exceptions (e.g. CARL [33] sets $w_i = \hat{s}_i$ where $\hat{s}_i$ is the classification score – Chapter 3).

Here, we show that, in fact, what aLRP localisation component does is soft sampling. To see this, first let us recall the definition of the localisation component:

$$\mathcal{L}_{loc}^{\text{aLRP}} = \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \frac{1}{\text{rank}(i)} \left(\mathcal{E}_{loc}(i) + \sum_{k \in \mathcal{P}, k \neq i} \mathcal{E}_{loc}(k) H(x_{ik}) \right), \quad (\text{K.2})$$

which is differentiable with respect to the box parameters as discussed in Chapter 7. With a ranking-based formulation, note that (i) the localisation error of a positive example i (i.e. $\mathcal{E}_{loc}(i)$) contributes each LRP value computed on a positive example

j where $\hat{s}_i \geq \hat{s}_j$ (also see Fig. 7.3), and (ii) each LRP value computed on a positive example i is normalized by $\text{rank}(i)$. Then, setting $\mathcal{L}(i) = \mathcal{E}_{loc}(i)$ in Eq. K.1 and accordingly taking Eq. K.2 in $\mathcal{E}_{loc}(i)$ paranthesis, the weights of the positive examples (i.e. $w_i = 0$ for negatives for the localisation component) are:

$$w_i = \frac{1}{|\mathcal{P}|} \left(\left(\sum_{k \in \mathcal{P}, k \neq i} \frac{H(x_{ki})}{\text{rank}(k)} \right) + \frac{1}{\text{rank}(i)} \right). \quad (\text{K.3})$$

Note that $\mathcal{L}(i)$ is based on a differentiable IoU-based regression loss and w_i is its weight, which is a scaler. As a result $H(x_{ki})$ in Eq. K.3 does not need to be smoothed and we use a unit-step function.

K.2 The Relation between aLRP Loss Value and Total Gradient Magnitudes

Here, we identify the relation between the loss value and the total magnitudes of the gradients following the generalized framework due to the fact that it is a basis for our self-balancing strategy introduced in Section 7.3.2 as follows:

$$\sum_{i \in \mathcal{P}} \left| \frac{\partial \mathcal{L}}{\partial \hat{s}_i} \right| = \sum_{i \in \mathcal{N}} \left| \frac{\partial \mathcal{L}}{\partial \hat{s}_i} \right| \approx \mathcal{L}^{\text{aLRP}}. \quad (\text{K.4})$$

Since using our generalization of error-driven optimisation or Identity Update both guarantees $\sum_{i \in \mathcal{P}} \left| \frac{\partial \mathcal{L}}{\partial \hat{s}_i} \right| = \sum_{i \in \mathcal{N}} \left| \frac{\partial \mathcal{L}}{\partial \hat{s}_i} \right|$ (Chapter 6), here we show that the loss value is approximated by the total magnitude of gradients. Based on Eq. 6.15, total gradients of the positives can be expressed as:

$$\sum_{i \in \mathcal{P}} \left| \frac{\partial \mathcal{L}}{\partial \hat{s}_i} \right| = \left| \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} \Delta x_{ij} \right|. \quad (\text{K.5})$$

Since $\Delta x_{ij} \geq 0$,

$$\frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} \Delta x_{ij}. \quad (\text{K.6})$$

Replacing the definition of the Δx_{ij} by $-(L_{ij}^* - L_{ij})$ yields:

$$-\frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} (L_{ij}^* - L_{ij}) = -\frac{1}{|\mathcal{P}|} \left(\sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} L_{ij}^* - \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} L_{ij} \right) \quad (\text{K.7})$$

$$= \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} L_{ij} - \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} L_{ij}^*. \quad (\text{K.8})$$

In our generalization of error-driven update (Appendix I), the first part (i.e. $\frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} L_{ij}$) yields the loss value, $\mathcal{L}$. Hence:

$$\sum_{i \in \mathcal{P}} \left| \frac{\partial \mathcal{L}}{\partial \hat{s}_i} \right| = \mathcal{L} - \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} L_{ij}^*. \quad (\text{K.9})$$

Reorganizing the terms, the difference between the total gradients of positives (or negatives, since they are equal – see Theorem 2) and the loss values itself is the sum of the targets normalized by number of positives:

$$\mathcal{L} - \sum_{i \in \mathcal{P}} \left| \frac{\partial \mathcal{L}}{\partial \hat{s}_i} \right| = \frac{1}{|\mathcal{P}|} \sum_{i \in \mathcal{P}} \sum_{j \in \mathcal{N}} L_{ij}^*. \quad (\text{K.10})$$

Compared to the primary terms, the targets are very small values (if not 0). For example, for AP Loss $L_{ij}^{\text{AP}*} = 0$, and hence, loss is equal to the sum of the gradients: $\mathcal{L} = \sum_{i \in \mathcal{P}} \left| \frac{\partial \mathcal{L}}{\partial \hat{s}_i} \right|$.

As for aLRP Loss, the target of a primary term is $\frac{\mathcal{E}_{loc}(i)}{\text{rank}(i)} \frac{H(x_{ij})}{N_{FP}(i)}$, hence if $H(x_{ij}) = 0$, then the target is also 0. Else if $H(x_{ij}) = 1$, then it implies that there are some negative examples with larger scores, and $\text{rank}(i)$ and $N_{FP}(i)$ are getting larger depending on these number of negative examples, which causes the denominator to grow, and hence yielding a small target as well. Then ignoring this term, we conclude that:

$$\sum_{i \in \mathcal{P}} \left| \frac{\partial \mathcal{L}}{\partial \hat{s}_i} \right| = \sum_{i \in \mathcal{N}} \left| \frac{\partial \mathcal{L}}{\partial \hat{s}_i} \right| \approx \mathcal{L}^{\text{aLRP}}. \quad (\text{K.11})$$

K.3 Self-balancing the Gradients Instead of the Loss Value

Instead of localisation the loss, $\mathcal{L}_{loc}^{\text{aLRP}}$, we multiply $\partial \mathcal{L} / \partial B$ by the average $\mathcal{L}^{\text{aLRP}} / \mathcal{L}_{loc}^{\text{aLRP}}$ of the previous epoch. This is because formulating aLRP Loss as $\mathcal{L}_{loc}^{\text{aLRP}} + w_r \mathcal{L}_{loc}^{\text{aLRP}}$ where w_r is a weight to balance the tasks is different from weighing the gradients with respect to the localisation output, B , since weighting the loss value (i.e. $\mathcal{L}_{loc}^{\text{aLRP}} + w_r \mathcal{L}_{loc}^{\text{aLRP}}$) changes the gradients of aLRP Loss with respect to the classification output as well since $\mathcal{L}_{loc}^{\text{aLRP}}$, now weighed by w_r , is also ranking-based (has $\text{rank}(i)$ term). Therefore, we directly add the self balance term as a multiplier of $\partial \mathcal{L} / \partial B$ and backpropagate accordingly. On the other hand, from a practical perspective, this can simply be implemented by weighing the loss value, $\mathcal{L}_{loc}^{\text{aLRP}}$ without modifying the gradient formulation for $\mathcal{L}_{cls}^{\text{aLRP}}$.



APPENDIX L

ADDITIONAL EXPERIMENTS WITH ALRP LOSS

This appendix presents more ablation experiments, the anchor configuration we use in our models and the effect of using a wrong target for the primary term in the error-driven update rule.

L.1 More Ablation Experiments: Using Self Balance and GIoU with AP Loss

We also test the effect of GIoU and our Self-balance approach on AP Loss, and present the results in Table L.1:

- Using IoU-based losses with AP Loss improves the performance up to 1.0 AP^C as well and reaches 36.5 AP^C with GIoU loss.
- Our SB approach also improves AP Loss between $0.7 - 1.2 \text{ AP}^C$, resulting in 37.2 AP^C as the best performing model without using w_r . However, it may not be inferred that SB performs better than constant weighting for AP Loss without a more thorough tuning of AP Loss since SB is devised to balance the gradients of localisation and classification outputs for aLRP Loss (see Section K.2).
- Comparing with the best performing model of AP Loss with 37.2 AP^C , (i) aLRP Loss has a 1.7 AP^C and 1.3 aLRP points better performance, (ii) the gap is 4.0 AP^C for AP_{90} , and (iii) the correlation coefficient of aLRP Loss, preserves the same gap (0.48 vs 0.44 comparing the best models for AP and aLRP Losses), since applying these improvements (IoU-based losses and SB) to AP Loss does not have an effect on unifying branches.

Table L.1: Using Self Balance and GIoU with AP Loss. For optimal LRP (oLRP), lower is better.

$\mathcal{L}_c$	$\mathcal{L}_r$	SB	AP ^C	AP ₅₀	AP ₇₅	AP ₉₀	oLRP	ρ
AP Loss [37]	Smooth L1		35.5	58.0	37.0	9.0	71.0	0.45
	Smooth L1	✓	36.7	58.2	39.0	10.8	70.2	0.44
	IoU Loss		36.3	57.9	37.9	11.8	70.4	0.44
	IoU Loss	✓	37.2	58.1	39.2	13.1	69.6	0.44
	GIoU Loss		36.5	58.1	38.1	11.9	70.2	0.45
	GIoU Loss	✓	37.2	58.3	39.0	13.4	69.7	0.44
aLRP Loss	with IoU		36.9	57.7	38.4	13.9	69.9	0.49
	with IoU	✓	38.7	58.1	40.6	17.4	68.5	0.48
	with GIoU	✓	38.9	58.5	40.5	17.4	68.4	0.48

L.2 Anchor Configuration

The number of anchors has a notable affect on the efficiency of training due to the time and space complexity of optimising ranking-based loss functions by combining error-driven update and backpropagation. For this reason, different from original RetinaNet using three aspect ratios (i.e. $[0.5, 1, 2]$) and three scales (i.e. $[2^{0/2}, 2^{1/2}, 2^{2/2}]$) on each location, Chen et al. [37] preferred the same three aspect ratios, but reduced the scales to two as $[2^{0/2}, 2^{1/2}]$ to increase the efficiency of AP Loss. In our ablation experiments, except the one that we used ATSS [42], we also followed the same anchor configuration of Chen et al. [37].

One main contribution of ATSS is to simplify the anchor design by reducing the number of required anchors to a single scale and aspect ratio (i.e. ATSS uses 1/9 and 1/6 of the anchors of RetinaNet [18] and AP Loss [37] respectively), which is a perfect fit for our optimisation strategy. For this reason, we used ATSS, however, we observed that the configuration in the original ATSS with a single aspect ratio and scale does not yield the best result for aLRP Loss, which may be related to the ranking nature of aLRP Loss which favors more examples to impose a more accurate ranking, loss and gradient computation. Therefore, different from ATSS configuration, we find it useful to set anchor scales $[2^{0/2}, 2^{1/2}]$ and $[2^{0/2}, 2^{1/2}, 2^{2/2}]$ for aLRPLoss500 and aLRPLoss800 respectively and use a single aspect ratio with 1 following the original design of ATSS.

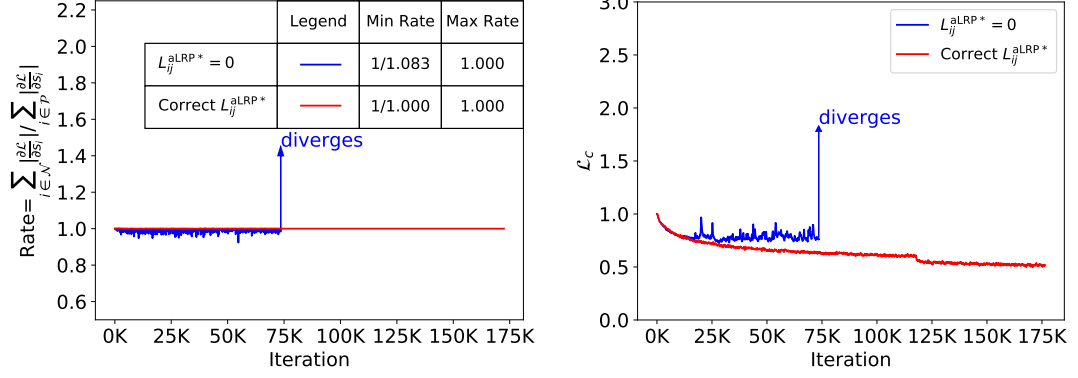


Figure L.1: **(left)** The rate of the total gradient magnitudes of negatives to positives. **(right)** Loss values.

L.3 Using a Wrong Target for the Primary Term in the Error-driven Update Rule

As discussed in Chapter 7, L_{ij}^* , the target value of the primary term L_{ij} is non-zero due to the localisation error. It is easy to overlook this fact and assume that the target is zero. Fig. L.1 presents this case where L_{ij}^* is set to 0 (i.e. minimum value of aLRP). In such a case, the training continues properly, similar to that of the correct case, up to a point and then diverges. Note that this occurs when the positives start to be ranked properly but are still assigned gradients since $L_{ij}^* - L_{ij} \neq 0$ due to the nonzero localisation error. This causes $\sum_{i \in \mathcal{P}} \left| \frac{\partial \mathcal{L}}{\partial \hat{s}_i} \right| > \sum_{i \in \mathcal{N}} \left| \frac{\partial \mathcal{L}}{\partial \hat{s}_i} \right|$, violating Theorem 2 (compare min-rate and max-rate in Fig. L.1). Therefore, assigning proper targets is crucial for balanced training.

L.4 Implementation Details for FoveaBox and Faster R-CNN

In this section, we provide more implementation details on the FoveaBox and Faster R-CNN models that we trained with different loss functions. All the models in this section are tested on COCO *minival*.

Implementation Details of FoveaBox: We train the models for 100 epochs with a learning rate decay at epochs 75 and 95. For aLRP Loss and AP Loss, we preserve

the same learning rates used for RetinaNet (i.e. 0.008 and 0.002 for aLRP Loss and AP Loss respectively). As for the Focal Loss, we set the initial learning rate to 0.02 following the linear scheduling hypothesis [138] (i.e. Kong et al. [174] set learning rate to 0.01 and use a batch size of 16). Following official implementation of AP Loss, the gradients of the regression loss (i.e. Smooth L1) are averaged over the output parameters of positive boxes for AP Loss. As for Focal Loss, we follow the mmdetection implementation which averages the total regression loss by the number of positive examples. The models are tested on COCO *minival* by preserving the standard design by mmdetection framework.

Implementation Details of Faster R-CNN: To train Faster R-CNN, we first replace the softmax classifier of Fast R-CNN by the class-wise sigmoid classifiers. Instead of heuristic sampling rules, we use all anchors to train RPN and top-1000 scoring proposals per image obtained from RPN to train Fast R-CNN (i.e. same with the default training except for discarding sampling). Note that, with aLRP Loss, the loss function consists of two independent losses instead of four in the original pipeline, hence instead of three scalar weights, aLRP Loss requires a single weight for RPN head, which we tuned as 0.20. Following the positive-negative assignment rule of RPN, different from all the experiments, which use $\tau = 0.50$, $\tau = 0.70$ for aLRP Loss of RPN. We set the initial learning rate to 0.04 following the linear scheduling hypothesis [138] for the baselines, and decreased by a factor of 0.10 at epochs 75 and 95. Localisation loss weight is kept as 1 for L1 Loss and to 10 for GIoU Loss [93, 100]. The models are tested on COCO *minival* by preserving the standard design by mmdetection framework. We do not train Faster R-CNN with AP Loss due to the difficulty to tune Faster R-CNN for a different loss function.

APPENDIX M

DETAILS OF RS LOSS

This appendix presents the derivations of gradients and obtains the loss value and gradients of RS Loss on an example in order to provide more insight.

M.1 Derivation of the Gradients

The gradients of a ranking-based loss function can be determined as follows. Eq. 6.15 states that

$$\frac{\partial \mathcal{L}}{\partial \hat{s}_i} = \frac{1}{Z} \left(\sum_{j \in \mathcal{P} \cup \mathcal{N}} \Delta x_{ji} - \sum_{j \in \mathcal{P} \cup \mathcal{N}} \Delta x_{ij} \right). \quad (\text{M.1})$$

Our identity update reformulation suggests replacing Δx_{ij} by L_{ij} yielding:

$$\frac{\partial \mathcal{L}}{\partial \hat{s}_i} = \frac{1}{Z} \left(\sum_{j \in \mathcal{P} \cup \mathcal{N}} L_{ji} - \sum_{j \in \mathcal{P} \cup \mathcal{N}} L_{ij} \right). \quad (\text{M.2})$$

We split both summations into two based on the labels of the examples, and express $\frac{\partial \mathcal{L}}{\partial \hat{s}_i}$ using four terms:

$$\frac{\partial \mathcal{L}}{\partial \hat{s}_i} = \frac{1}{Z} \left(\sum_{j \in \mathcal{P}} L_{ji} + \sum_{j \in \mathcal{N}} L_{ji} - \sum_{j \in \mathcal{P}} L_{ij} - \sum_{j \in \mathcal{N}} L_{ij} \right). \quad (\text{M.3})$$

Then simply by using the primary terms of RS Loss, defined in Eq. 8.5 as:

$$L_{ij} = \begin{cases} (\ell_R(i) - \ell_R^*(i)) p_R(j|i), & \text{for } i \in \mathcal{P}, j \in \mathcal{N} \\ (\ell_S(i) - \ell_S^*(i)) p_S(j|i), & \text{for } i \in \mathcal{P}, j \in \mathcal{P}, \\ 0, & \text{otherwise,} \end{cases} \quad (\text{M.4})$$

With the primary term definitions, we obtain the gradients of RS Loss using Eq. M.3.

Gradients for $i \in \mathcal{N}$. For $i \in \mathcal{N}$, we can respectively express the four terms in Eq. M.3 as follows:

- $\sum_{j \in \mathcal{P}} L_{ji} = \sum_{j \in \mathcal{P}} (\ell_R(j) - \ell_R^*(j)) p_R(i|j),$
- $\sum_{j \in \mathcal{N}} L_{ji} = 0$ (no negative-to-negative error is defined for RS Loss – see Eq. M.4),
- $\sum_{j \in \mathcal{P}} L_{ij} = 0$ (no error when $j \in \mathcal{P}$ and $i \in \mathcal{N}$ for L_{ij} – see Eq. M.4),
- $\sum_{j \in \mathcal{N}} L_{ij} = 0$ (no negative-to-negative error is defined for RS Loss – see Eq. M.4),

which, then, can be expressed as (by also replacing $Z = |\mathcal{P}|$ following the definition of RS Loss):

$$\frac{\partial \mathcal{L}_{RS}}{\partial \hat{s}_i} = \frac{1}{|\mathcal{P}|} \left(\sum_{j \in \mathcal{P}} L_{ji} + \sum_{j \in \mathcal{N}} \overset{0}{L_{ji}} - \sum_{j \in \mathcal{P}} \overset{0}{L_{ij}} - \sum_{j \in \mathcal{N}} \overset{0}{L_{ij}} \right), \quad (\text{M.5})$$

$$= \frac{1}{|\mathcal{P}|} \sum_{j \in \mathcal{P}} \left(\ell_R(j) - \overset{0}{\ell_R^*(j)} \right) p_R(i|j), \quad (\text{M.6})$$

$$= \frac{1}{|\mathcal{P}|} \sum_{j \in \mathcal{P}} \ell_R(j) p_R(i|j), \quad (\text{M.7})$$

concluding the derivation of the gradients if $i \in \mathcal{N}$.

Gradients for $i \in \mathcal{P}$. We follow the same methodology for $i \in \mathcal{P}$ and express the same four terms as follows:

- $\sum_{j \in \mathcal{P}} L_{ji} = \sum_{j \in \mathcal{P}} (\ell_S(j) - \ell_S^*(j)) p_S(i|j),$
- $\sum_{j \in \mathcal{N}} L_{ji} = 0$ (no error when $j \in \mathcal{N}$ and $i \in \mathcal{P}$ for L_{ji} – see Eq. M.4),
- $\sum_{j \in \mathcal{P}} L_{ij}$ reduces to $\ell_S(i) - \ell_S^*(i)$ simply by rearranging the terms and $\sum_{j \in \mathcal{P}} p_S(j|i) =$

1 since $p_S(j|i)$ is a pmf:

$$\sum_{j \in \mathcal{P}} L_{ij} = \sum_{j \in \mathcal{P}} (\ell_S(i) - \ell_S^*(i)) p_S(j|i), \quad (\text{M.8})$$

$$= (\ell_S(i) - \ell_S^*(i)) \sum_{j \in \mathcal{P}} p_S(j|i), \quad (\text{M.9})$$

$$= \ell_S(i) - \ell_S^*(i). \quad (\text{M.10})$$

- Similarly, $\sum_{j \in \mathcal{N}} L_{ij}$ reduces to $\ell_R(i) - \ell_R^*(i)$:

$$\sum_{j \in \mathcal{N}} L_{ij} = \sum_{j \in \mathcal{N}} (\ell_R(i) - \ell_R^*(i)) p_R(j|i) \quad (\text{M.11})$$

$$= (\ell_R(i) - \ell_R^*(i)) \sum_{j \in \mathcal{N}} p_R(j|i) \quad (\text{M.12})$$

$$= \ell_R(i) - \ell_R^*(i). \quad (\text{M.13})$$

Combining these four cases together, we have the following gradient for $i \in \mathcal{P}$:

$$\frac{\partial \mathcal{L}_{RS}}{\partial \hat{s}_i} = \frac{1}{|\mathcal{P}|} \left(\sum_{j \in \mathcal{P}} (\ell_S(j) - \ell_S^*(j)) p_S(i|j) \right) \quad (\text{M.14})$$

$$- (\ell_S(i) - \ell_S^*(i)) - (\ell_R(i) - \ell_R^*(i)). \quad (\text{M.15})$$

Finally, for clarity, we rearrange the terms also by using $\ell_{RS}^*(i) - \ell_{RS}(i) = -(\ell_S(i) - \ell_S^*(i)) - (\ell_R(i) - \ell_R^*(i))$:

$$\frac{1}{|\mathcal{P}|} \left(\ell_{RS}^*(i) - \ell_{RS}(i) + \sum_{j \in \mathcal{P}} (\ell_S(j) - \ell_S^*(j)) p_S(i|j) \right), \quad (\text{M.16})$$

concluding the derivation of the gradients when $i \in \mathcal{P}$.

M.2 More Insight on RS Loss Computation and Gradients on an Example

In Fig. M.1, we illustrate the input and the computation of RS Loss. We also note here that our Identity Update provides interpretable loss values when the target value is non-zero (Fig. M.1(b)). Previous work [37, 156] fail to provide interpretable loss values.

Input id, i ($i \in \mathcal{P}$ is underlined)	<u>0</u>	<u>1</u>	2	3	<u>4</u>	5	<u>6</u>	7
(a) Input of RS Loss (positive if $y_i > 0$; else negative)								
Logits, $\hat{s}_i$	3.00	2.00	1.00	0.00	-1.00	-2.00	-3.00	-4.00
Labels, y_i	0.90	0.40	0.00	0.00	0.80	0.00	0.10	0.00
(b) Current&target error and RS Loss on each $i \in \mathcal{P}$ (N/A: negatives; bold: non-zero loss)								
Current ranking error, $\ell_R(i)$	0.00	0.00	N/A	N/A	0.40	N/A	0.42	N/A
Current sorting error, $\ell_S(i)$	0.10	0.35	N/A	N/A	0.30	N/A	0.38	N/A
Target ranking error, $\ell_R^*(i)$	0.00	0.00	N/A	N/A	0.00	N/A	0.00	N/A
Target sorting error, $\ell_S^*(i)$	0.10	0.35	N/A	N/A	0.15	N/A	0.38	N/A
Ranking Loss, $\ell_R(i) - \ell_R^*(i)$	0.00	0.00	N/A	N/A	0.40	N/A	0.42	N/A
Sorting Loss, $\ell_S(i) - \ell_S^*(i)$	0.00	0.00	N/A	N/A	0.15	N/A	0.00	N/A
Total Loss, $(\ell_R(i) + \ell_S(i)) - (\ell_R^*(i) + \ell_S^*(i))$	0.00	0.00	N/A	N/A	0.55	N/A	0.42	N/A
RS Loss, $\mathcal{L}_{RS}$ (average over total losses) : 0.24								

Figure M.1: An example case to illustrate the computation of RS Loss. (a) The inputs of RS Loss are logits and continuous ground-truth labels (i.e. IoU). (b) Thanks to the “Identity Update” (Section 6.2), the loss computed on each example considers the target error, hence, it is interpretable. E.g. $i = 0, 1, 6$ have positive current sorting errors, but already sorted properly among examples with larger logits, which can be misleading when loss value ignores the target error. Since RS Loss is computed on positives, N/A is assigned for negatives.

Computation of the Loss. To compute the loss following three-step algorithm, the first and the third steps are trivial (Fig. 6.1), thus, here we present in Fig. M.2(a) how the primary terms (L_{ij}) are computed for our example (Fig. M.1) following Eq. M.4.

Optimisation of the Loss. Fig. M.2(b) presents and discusses the gradients obtained using Eq. M.5, M.16.

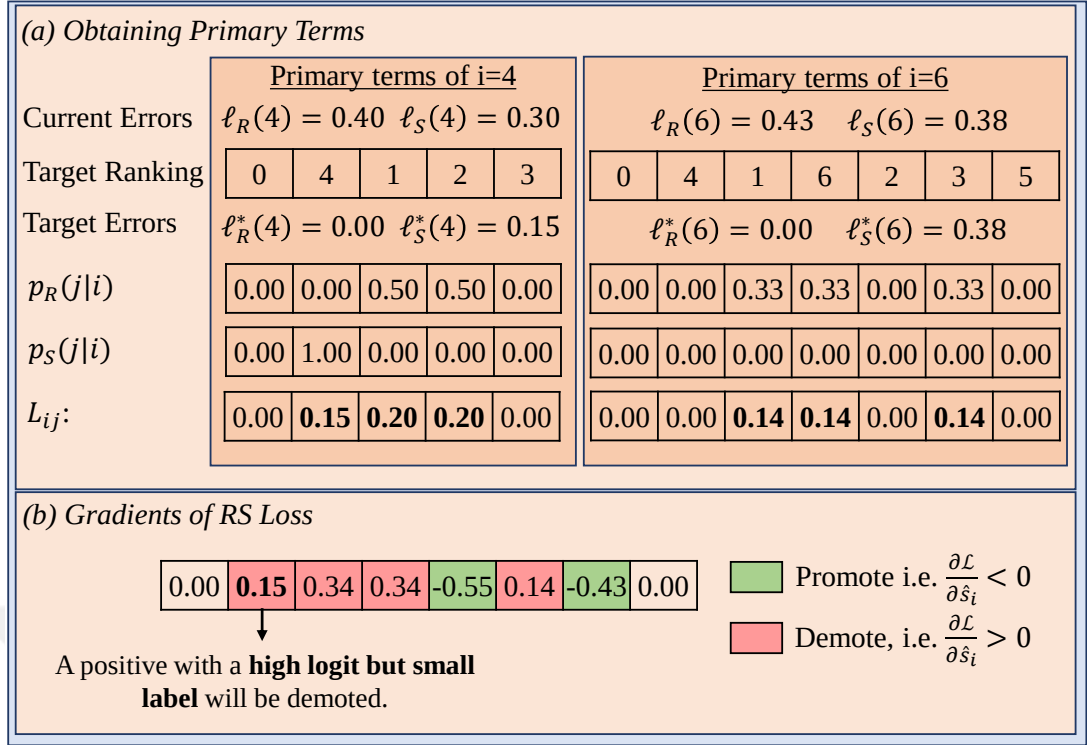


Figure M.2: An example case to illustrate the computation of primary terms in RS Loss. (a) The computation of primary terms. We only show the computation for positives $i = 4$ and $i = 6$ since for $i = 0$ and $i = 1$ the total loss is 0 (see Fig. M.1(b)); and RS Loss does not compute error on negatives by definition (i.e. discretizes the space only on positives). To compute primary terms, L_{ij} , one needs current errors, target errors and pmfs for both ranking and sorting, which are included in the figure respectively. In order to compute the target errors on a positive $i \in \mathcal{P}$, the examples are first thresholded from $\hat{s}_i$ and the ones with larger (i.e. $\hat{s}_j \geq \hat{s}_i$) logits are obtained. Then, target rankings are identified using continuous labels. The ranking and sorting errors computed for the target ranking determines target errors, $\ell_R^*(i)$ and $\ell_S^*(i)$. The ranking and sorting losses, $\ell_R(i) - \ell_R^*(i)$ and $\ell_S(i) - \ell_S^*(i)$ respectively, are then distributed over examples causing these losses uniformly via pmfs $p_S(j|i)$ and $p_R(j|i)$ to determine pairwise errors, i.e. primary terms. (b) The gradients are obtained simply by using primary terms as the update in Eq. M.1 following identity update yielding Eq. M.5 and M.16 for negatives and positives respectively. Thanks to the novel sorting objective, RS Loss can assign a gradient to suppress a positive example when it is not ranked among positives accordingly wrt its continuous label (e.g. $i = 1$).



APPENDIX N

COMPARATIVE ANALYSIS OF RS LOSS AND ALRP LOSS

This appendix provides a comparative analysis of our RS Loss and our aLRP Loss and presents why RS Loss converges faster and performs better. In the following, we list and analyse our four observations on aLRP Loss [156] based on our comparative analysis with RS Loss:

Observation 1: Competing tasks for the bounded range of aLRP Loss degrades performance especially when the models are trained 12 epochs following the common training schedule.

To illustrate this, we train Faster R-CNN [19] with aLRP Loss and our RS Loss using two different settings:

- “Standard Training”, which refers to the common training (e.g. [19, 160, 42]): The network is trained by a batchsize of 16 images with resolution 1333×800 without any augmentation except the standard horizontal flipping. We use 4 GPUs, so each GPU has 4 images during training. We tune the learning rate of aLRP Loss as 0.009 and for our RS Loss we set it to 0.012. Consistent with the training image size, the test image size is 1333×800 .
- “Heavy Training”, which refers to the standard training design of aLRP Loss (and also AP Loss): The network is trained by a batch size of 32 images with resolution 512×512 on 4 GPUs (i.e. 8 images/GPU) using SSD-syle augmentation [24] for 100 epochs. We use the initial learning rate of 0.012 for aLRP Loss as validated in Chapter 7, and for our RS Loss, we simply use linear scheduling hypothesis and set it to 0.024 without further validation. Here, following aLRP Loss (and AP Loss) design the test image size is 833×500 .

Table N.1: Due to epoch-based self-balance and competition of tasks for the limited range, aLRP Loss performs significantly worse in the first epoch. When the model is trained longer using heavy training (i.e. 100 epochs, SSD-style augmentation [24]), the default configuration of aLRP Loss, the performance gap relatively decreases at the end of training, however, the gap is still significant ($\sim 2AP^C$) for the standard training (i.e. 12 epochs, no SSD-style augmentation [24]). All experiments are conducted on Faster R-CNN. COCO-style AP (AP^C) is reported.

Loss Function	Heavy Training		Standard Training	
	Epoch 1	Epoch 100	Epoch 1	Epoch 12
aLRP Loss	9.4	40.7	14.4	37.4
RS Loss	17.7	41.2	22.0	39.6

Table N.1 presents the results and we observe the following:

1. For both “heavy training” and “standard training”, aLRP Loss has significantly lower performance ($17.7AP^C$ vs. $9.2AP^C$ for heavy training and $22.0AP^C$ vs. $14.4AP^C$) compared to RS Loss: aLRP Loss has a bounded range between 0 and 1, which is dominated by the classification head especially in the beginning of the training, and hence, the box regression head is barely trained. To tackle that, we dynamically promoted the loss of box regression head using a self-balance weight, initialized to 50 and updated based on loss values at the end of every epoch. However, we observed that this range pressure has an adverse effect on the performance especially at the beginning of the training, which could not be fully addressed by self-balance since in the first epoch the SB weight is not updated.
2. While the gap between RS Loss and aLRP Loss is $0.5AP^C$ for “heavy training”, it is $2.2AP^C$ for “standard training”. After the SB weight of aLRP Loss is updated, the gap can be reduced when the models are trained for longer epochs. However, the final gap is still large ($\sim 2AP^C$) for “standard training” with 12 epochs since unlike aLRP Loss, our RS Loss (i) does not have a single bounded range for which multiple tasks compete, and (ii) uses an iteration-based update approach, hence SB weight is updated every iteration.

Observation 2: The target of aLRP Loss is hand-crafted, hence does not have an intuitive interpretation.

Self-balance (or range pressure – see Observation 1) is not the single reason why RS Loss performs better than aLRP Loss in both scheduling methods in Table N.1. aLRP Loss uses the following target error for a positive example i :

$$\ell_{aLRP}^*(i) = \frac{\mathcal{E}_{loc}(i)}{\text{rank}(i)}, \quad (\text{N.1})$$

where

$$\mathcal{E}_{loc}(i) = \frac{1 - \text{IoU}(\hat{b}_i, b_i)}{1 - \tau}, \quad (\text{N.2})$$

and τ is the positive-negative assignment threshold. However, unlike the target of RS Loss for specifying the error at the target ranking where positives are sorted wrt their IoUs (see Fig. M.2), the target of aLRP Loss is handcrafted and does not have an intuitive interpretation.

Observation 3: Setting τ in Eq. N.2 to the value of the positive-negative (anchor IoU) assignment threshold creates ambiguity (e.g. anchor-free detectors do not have such threshold).

We identify three obvious reasons: (i) Anchor-free methods do not use IoU to assign positives and negatives, (ii) recent SOTA anchor-based methods, such as ATSS [42] and PAA [43], do not have a sharp threshold to assign positives and negatives, but instead they use adaptive thresholds to determine positives and negatives during training, and furthermore (iii) anchor-based detectors split anchors as positives and negatives; however, the loss is computed on the predictions which may have less IoU with ground truth than 0.50. Note that our RS Loss directly uses IoUs as the continuous labels without further modifying or thresholding them.

Observation 4: Using an additional hyper-parameter (δ_{loc}) for ranking-based weighting yields better performance for the common 12 epoch training.

As also discussed in Section 8.4.2, *ranking-based* importance weighting of the instances corresponds to (see Chapter 2 for the notation):

$$w^i = \frac{1}{|\mathcal{P}|} \left(\sum_{k \in \mathcal{P}} \frac{H(x_{ki})}{\text{rank}(k)} \right). \quad (\text{N.3})$$

Table N.2: Using an additional δ_{loc} to smooth the effect of ranking-based weighting can contribute to the performance.

δ_{loc}	0.00	0.50	1.00	1.50	2.00	Default
AP ^C	39.3	39.4	39.8	39.9	39.8	39.5

aLRP Loss, by default, prefers not to smooth the nominator ($H(x_{ki})$) but $\text{rank}(k)$ is computed by the smoothed unit-step function. We label this setting as “default” and introduce an additional hyper-parameter δ_{loc} to further analyse ranking-based weighting. Note that the larger δ_{loc} is, the less variance will w^i s have. In Table N.2, we compare these different settings on RS-ATSS trained for 12 epochs with our RS Loss, and observe that with different δ_{loc} values, the default ranking-based weighting can be improved. However, for our RS Loss, we adopt score-based weighting owing to its tuning-free nature.

APPENDIX O

THE RELATION OF RS LOSS WITH LRP ERROR

Similar to aLRP Loss, we also derive our RS Loss based on our performance metric, LRP Error (Chapter 5). This appendix provides how RS Loss can be obtained originating from LRP Error.

We showed in Eq. 7.3 that the loss form of the LRP Error (Eq. 5.2) is:

$$\ell_{\text{LRP}}(i) = \frac{1}{\text{rank}(i)} \left(N_{\text{FP}}(i) + \mathcal{E}_{\text{loc}}(i) + \sum_{k \in \mathcal{P}, k \neq i} \mathcal{E}_{\text{loc}}(k) H(x_{ik}) \right). \quad (\text{O.1})$$

Simply by manipulating Eq. O.1, we reach O.6:

$$\ell_{\text{LRP}}(i) = \frac{1}{\text{rank}(i)} \left(N_{\text{FP}}(i) + \sum_{k \in \mathcal{P}} \mathcal{E}_{\text{loc}}(k) H(x_{ik}) \right) \quad (\text{O.2})$$

$$= \frac{N_{\text{FP}}(i)}{\text{rank}(i)} + \frac{\sum_{k \in \mathcal{P}} \mathcal{E}_{\text{loc}}(k) H(x_{ik})}{\text{rank}(i)} \quad (\text{O.3})$$

$$= \frac{N_{\text{FP}}(i)}{\text{rank}(i)} + \frac{\sum_{k \in \mathcal{P}} \mathcal{E}_{\text{loc}}(k) H(x_{ik}) \text{rank}^+(i)}{\text{rank}(i) \text{rank}^+(i)} \quad (\text{O.4})$$

$$= \frac{N_{\text{FP}}(i)}{\text{rank}(i)} + \frac{\text{rank}^+(i)}{\text{rank}(i)} \frac{\sum_{k \in \mathcal{P}} \mathcal{E}_{\text{loc}}(k) H(x_{ik})}{\text{rank}^+(i)} \quad (\text{O.5})$$

$$= \frac{N_{\text{FP}}(i)}{\text{rank}(i)} + \left(1 - \frac{N_{\text{FP}}(i)}{\text{rank}(i)} \right) \frac{\sum_{k \in \mathcal{P}} \mathcal{E}_{\text{loc}}(k) H(x_{ik})}{\text{rank}^+(i)} \quad (\text{O.6})$$

Considering the definitions of the components of LRP (Eq. 5.4-5.6), we can define the loss forms of localisation and FP components using this notation respectively as,

$$\text{LRP}_{\text{FP}}(i) = \frac{N_{\text{FP}}(i)}{\text{rank}(i)}, \quad (\text{O.7})$$

$$\text{LRP}_{\text{Loc}}(i) = \frac{\sum_{k \in \mathcal{P}} \mathcal{E}_{\text{loc}}(k) H(x_{ik})}{\text{rank}^+(i)}. \quad (\text{O.8})$$

Replacing these definitions in Eq. O.6, we have

$$= \text{LRP}_{\text{FP}}(i) + (1 - \text{LRP}_{\text{FP}}(i)) \text{LRP}_{\text{Loc}}(i) \quad (\text{O.9})$$

$$= \text{LRP}_{\text{FP}}(i) + \text{LRP}_{\text{Loc}}(i) - \text{LRP}_{\text{FP}}(i) \text{LRP}_{\text{Loc}}(i), \quad (\text{O.10})$$

which comprises the three terms discussed as follows:

- **First term.** $\text{LRP}_{\text{FP}}(i) = \frac{N_{\text{FP}}(i)}{\text{rank}(i)}$ is the precision error (as in AP Loss), and as a result, includes the errors between positives and negatives.
- **Second term.** $\text{LRP}_{\text{Loc}}(i) = \frac{\sum_{k \in \mathcal{P}} \mathcal{E}_{\text{loc}}(k) H(x_{ik})}{\text{rank}^+(i)}$ is the average localisation error of positives. Noting that it does not require any information from negatives, this term corresponds to the errors among positives.
- **Third term.** $\text{LRP}_{\text{FP}}(i) \text{LRP}_{\text{Loc}}(i)$ combines classification and localisation errors in order to squeeze LRP between 0 and 1. Since the first two terms encode both types of errors that we aim to optimise, this term can simply be ignored, which removes the range pressure arising for aLRP Loss (see Observation 1 in Appendix N).

Thus, we obtain the RS Loss on a positive example as follows:

$$\ell_{\text{RS}}(i) = \text{LRP}_{\text{FP}}(i) + \text{LRP}_{\text{Loc}}(i) \quad (\text{O.11})$$

$$= \frac{N_{\text{FP}}(i)}{\text{rank}(i)} + \frac{\sum_{k \in \mathcal{P}} \mathcal{E}_{\text{loc}}(k) H(x_{ik})}{\text{rank}^+(i)} \quad (\text{O.12})$$

In order to prevent assignment threshold ambiguity (see Observation 3 in Appendix N), we set $\tau = 0$ in $\mathcal{E}_{\text{loc}}(k) = \frac{1 - \text{IoU}(k)}{1 - \tau}$ (see definition of LRP in Eq. 5.2), implying $\mathcal{E}_{\text{loc}}(k) = 1 - \text{IoU}(k)$. Hence we have,

$$\ell_{\text{RS}}(i) = \text{LRP}_{\text{FP}}(i) + \text{LRP}_{\text{Loc}}(i) \quad (\text{O.13})$$

$$= \frac{N_{\text{FP}}(i)}{\text{rank}(i)} + \frac{\sum_{k \in \mathcal{P}} (1 - \text{IoU}(k)) H(x_{ik})}{\text{rank}^+(i)} \quad (\text{O.14})$$

Finally, for generalisation over different classification problems, we simply set IoU to the continuous ground truth label (i.e. $\text{IoU}(k) = y_k$), replace the index subscript k

by j to obtain the definition of RS Loss (c.f. Eq. 8.3):

$$\ell_{\text{RS}}(i) = \underbrace{\frac{N_{\text{FP}}(i)}{\text{rank}(i)}}_{\ell_{\text{R}}(i): \text{Current Ranking Error}} + \underbrace{\frac{\sum_{j \in \mathcal{P}} H(x_{ij})(1 - y_j)}{\text{rank}^+(i)}}_{\ell_{\text{S}}(i): \text{Current Sorting Error}}. \quad (\text{O.15})$$

We use RS Loss to train the classification branch by designing a consistent sorting target different from the handcrafted target of aLRP Loss (see Observation 2 in Appendix N). As for the box regression and mask prediction branches, we do not prefer ranking-based weighting originating from aLRP Loss since we observed that it requires tuning an additional hyperparameter, hence for simplicity, instead we employ score-based weighting (see Observation 4 in Appendix N).

As a conclusion, we say that similar to aLRP Loss, RS Loss originates from LRP Error and fixes drawbacks of aLRP Loss (see observations in Appendix N).



APPENDIX P

MORE EXPERIMENTS WITH RS LOSS

This appendix presents more experiments on RS Loss.

P.1 Effect of smoothing unit-step function, the single hyper-parameter, for RS Loss.

Table P.1 presents the effect of δ_{RS} on RS Loss using ATSS. We observed similar performance between $\delta_{RS} = 0.40$ and $\delta_{RS} = 0.75$. Also note that considering positive-to-positive errors in the sorting error, we set δ_{RS} different from AP Loss and aLRP Loss, both of which smooth the unit step function by using $\delta_{RS} = 1.00$ as validated by Chen et al. [37].

P.2 Training Cascade R-CNN with RS Loss

Table P.2 shows that using RS Loss to train Cascade R-CNN (RS-Cascade R-CNN) also improves baseline Cascade R-CNN [20] by 1.0AP^C . We note that unlike the conventional training, we do not assign different loss weights over each R-CNN.

Table P.1: We set $\delta_{RS} = 0.50$, the only hyper-parameter of RS Loss, in all our experiments.

δ_{RS}	0.25	0.40	0.50	0.60	0.75	1.00
AP^C	39.0	39.7	39.9	39.7	39.8	39.4

Table P.2: RS Loss improves strong baseline Cascade R-CNN [20].

Method	AP ^C ↑	AP ₅₀ ↑	AP ₇₅ ↑	oLRP ↓
Cascade R-CNN	40.3	58.6	44.0	67.0
RS Cascade R-CNN	41.3	58.9	44.7	66.6

P.3 Hyper-parameters of R-CNN Variants in Table 8.2

A two-stage detector that uses random sampling and does not employ a method to adaptively set λ_t^k has at least 7 hyper-parameters since (i) for random sampling, one needs to tune number of foreground examples and number of background examples to be sampled in both stages (4 hyper-parameters), and (ii) at least 3 λ_t^k s need to be tuned as the task-balancing coefficients in a loss with four components (Eq.1.1). As a result, except aLRP Loss and our RS Loss, all methods have at least 7 hyper-parameters. When the box regression losses of RPN and R-CNN are L1 Loss, GIoU Loss or AutoLoss, and the network has not an additional auxiliary head, 7 hyper-parameters are sufficient (i.e. GIoU Loss [93], Carafe FPN [177] and AutoLoss-A [178]). Below, we list the methods with more than 7 hyper-parameters:

- FPN [29] uses Smooth L1 in both stages, resulting in 2 more additional additional hyper-parameters (β) to be tuned for the cut-off from L2 Loss to L1 Loss for Smooth L1 Loss.
- IoU-Net [130] also has Smooth L1 in both stages. Besides, there is an additional IoU prediction head trained also by Smooth L1, implying λ_t^k for IoU prediction head and β for Smooth L1. In total, there are 7 hyper-parameters in the baseline model, and with these 4 hyper-parameters, IoU-Net includes 11 hyper-parameters.
- To train R-CNN, Libra R-CNN [32] uses IoU-based sampler, which splits the negatives into IoU bins with an IoU interval width of κ , then also exploits random sampling. Besides it uses Balanced L1 Loss which adds 2 more hyper-parameters to Smooth L1 Loss (3 hyper-parameters in total). As a result, Libra R-CNN has 11 hyper-parameters in sampling and loss function in total.

Table P.3: Analysis whether using continuous labels is useful for instance segmentation. We use IoU to supervise instance segmentation methods except SOLOv2, in which we use Dice coefficient since bounding boxes are not included in the output. Using Dice coefficient also provides similar performance with IoU. Binary refers to the conventional training without continuous labels. AP^C : COCO-style AP

Label	Segmentation			Detection		
	AP^C	AP_{50}	AP_{75}	AP^C	AP_{50}	AP_{75}
Binary	29.1	49.9	29.4	32.9	53.8	34.2
IoU	29.9	50.5	30.6	33.8	54.2	35.4
Dice	29.8	50.4	30.2	33.5	54.3	35.1
(IoU+Dice)/2	29.6	50.2	30.0	33.4	54.1	34.8

- KL Loss [90] uses Smooth L1 for RPN and initializes the mean and variance of KL Loss. Hence with these 3 additional it has 10 hyper-parameters in total.
- Dynamic R-CNN [36] uses Smooth L1 for RPN and adds one more hyper-parameter to the Smooth L1, resulting in 3 additional hyperparameters. As a result, it has 10 hyper-parameters.

P.4 Using different localisation qualities as continuous labels to supervise instance segmentation methods

In order to provide more insight regarding the employment of continuous labels for the instance segmentation methods, in addition to discarding continuous labels (c.f. “None” in Table P.3), we train YOLACT with three different continuous labels: (i) IoU, as the bounding box quality, (ii) Dice coefficient, as the segmentation quality, and (iii) the average of IoU and Dice coefficient. Table P.3 suggests that all of these localisation qualities improve performance against ignoring them during training. Therefore, we use IoU as the continuous ground truth labels in all of our experiments with the exception of RS-SOLOv2, in which we used Dice coefficient, yielding similar performance to using IoU (Table P.3), since SOLOv2 does not have a box regression head.

Table P.4: Average iteration time of methods trained by the standard loss vs. RS Loss. We only include Faster R-CNN among multi-stage methods. On average, training with RS Loss incurs $\sim 1.5\times$ longer time due to its quadratic time complexity similar to other existing ranking-based loss functions [37, 156].

Method	Standard Loss (sec)	RS Loss (sec)
Faster R-CNN	0.42	0.82
Cascade R-CNN	0.51	2.26
ATSS	0.44	0.70
PAA	0.57	0.99
Mask R-CNN	0.64	1.04
YOLACT	0.57	0.59
SOLOv2-light	0.64	0.90

P.5 Training time comparison and inference time of methods

Table P.4 compares the single iteration time of RS Loss with other loss functions, and also provides the inference time of the methods we train. On average, using RS Loss increases the training time around 1.5 times more.

P.6 Detailed Results

We provide in Table P.5 the detailed performance (i.e. AP-based, oLRP-based performance measures and fps) of two-stage methods trained with our RS Loss on COCO *minival*.

Table P.5: Comprehensive performance results of models trained by RS-Loss on COCO *minival*. We report AP-based and oLRP-based performance measures, an also inference time on a single Tesla V100 GPU as fps. AP^C: COCO-style AP

Method	Backbone	Epoch	MS train	AP ^C ↑	AP ₅₀ ↑	AP ₇₅ ↑	oLRP ↓	oLRP _{Loc} ↓	oLRP _{FP} ↓	oLRP _{FN} ↓	fps
<i>Object Detection</i>											
RS-Faster R-CNN	R-50	12	–	39.6	59.5	43.0	67.9	16.3	27.8	45.4	22.5
RS-Mask R-CNN	R-50	12	–	40.0	59.8	43.4	67.5	16.1	27.6	44.7	22.8
RS-Faster R-CNN+	R-50	12	–	40.7	61.4	43.8	66.9	16.3	26.4	43.7	21.5
RS-Mask R-CNN+	R-50	12	–	41.1	61.4	44.8	66.6	16.0	26.2	44.0	21.1
RS-Mask R-CNN	R-101	36	[640, 800]	44.7	64.4	48.8	63.7	14.7	24.5	41.3	17.1
RS-Mask R-CNN+	R-101	36	[480, 960]	46.1	66.2	50.3	62.6	14.5	23.5	39.9	16.6
RS-Faster R-CNN	R-101-DCN	36	[480, 960]	47.6	67.8	51.2	61.1	14.4	22.7	37.9	14.1
RS-Faster R-CNN+	R-101-DCN	36	[480, 960]	47.6	68.1	51.9	60.9	14.7	20.9	38.2	13.6
RS-Mask R-CNN+	R-101-DCN	36	[480, 960]	48.7	68.9	53.1	60.2	14.3	21.3	36.9	13.5
<i>Instance Segmentation</i>											
RS-Mask R-CNN	R-50	12	–	36.4	57.3	39.2	70.1	18.2	28.7	46.5	17.1
RS-Mask R-CNN+	R-50	12	–	37.3	58.6	40.2	69.4	18.1	28.0	45.3	16.7
RS-Mask R-CNN	R-101	36	[640, 800]	40.3	61.9	43.8	66.9	17.3	24.3	43.0	14.8
RS-Mask R-CNN+	R-101	36	[480, 960]	41.4	63.6	44.7	65.9	17.1	22.8	42.2	14.3
RS-Mask R-CNN+	R-101-DCN	36	[480, 960]	43.5	66.5	47.2	64.0	17.2	22.3	38.5	11.9