

# **Deep Structured Models for Large Scale Object Co-detection and Segmentation**

**Zeeshan Hayder**

A thesis submitted for the degree of  
Doctor of Philosophy  
The Australian National University

July 2017

© Copyright by Zeeshan Hayder 2017  
All Rights Reserved

## Declaration

I hereby declare that this thesis is my own original work (in collaboration with co-authors), except where due references are made in the text. The work presented in this thesis contains no material previously published or written by another person or me in whole or part for the award of any other degree or diploma at ANU or any other educational institution or other tertiary educational institution, except where due acknowledgment has been made. I also declare that all sources used in this thesis have been fully and properly cited. The content of this thesis is mainly based on the peer-reviewed publications during my PhD as listed below,

1. Zeeshan Hayder, Xuming He and Mathieu Salzmann. Boundary-aware instance segmentation, In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. (acceptance rate: 29.0%).
2. Zeeshan Hayder, Xuming He and Mathieu Salzmann. Learning to co-generate object proposals with a deep structured network, In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. (acceptance rate: 29.0%).
3. Zeeshan Hayder, Xuming He and Mathieu Salzmann. Structural kernel learning for large scale multiclass object co-detection, In *IEEE International Conference on Computer Vision (ICCV)*, 2015. (acceptance rate: 30.0%).
4. Zeeshan Hayder, Mathieu Salzmann and Xuming He. Object co-detection via efficient inference in a fully-connected crf, In the *European Conference on Computer Vision (ECCV)*, 2014. (acceptance rate: 30.0%).

Zeeshan Hayder  
20 July 2017





To my family.



---

# Acknowledgements

---

First of all, I offer my foremost gratitude to Almighty for the immense blessings that he has bestowed upon me by giving me the ability, strength, knowledge and courage to undertake this research and to complete my Ph.D. studies successfully.

It is really ineffable and difficult to overstate my sincere gratitude to my venerable supervisors, Assoc. Prof. Xuming He and Assoc. Prof. Mathieu Salzmann, for their benevolent guidance, unceasing encouragement and invaluable support throughout my research work. It has been an absolute privilege to work under their careful supervision and get benefited from their profound knowledge and expertise in the field of computer vision and machine learning. This manuscript would never have reached its present form without their genuine interest, constructive suggestions, critical appraisal and contagious enthusiasm for research. Also, a special thanks go to other members of my thesis advisory committee, Prof. Richard Hartley (panel chair) and Dr. Hanxi Li (advisor) for their constructive suggestions, ever available help and encouragement throughout the study.

I would like to thank Australian National University (ANU), NICTA (now Data61, CSIRO) and the Australian Government for generous financial support including ANU HDR, NICTA Ph.D. and supplementary scholarships. Additionally, I would like to acknowledge the support I received in the form of multiple International Travel Grants to attend top vision conferences including CVPR/ICCV and ECCV.

I express my sincere gratitude to my colleagues and postdoctoral researchers at Data61 and ANU for their invaluable comments and expert advise in both academic and personal matters. Especially, I would like to convey my gratitude to Sadeep Jayasumana (for guidance and useful suggestions earlier in my Ph.D.), Arash, Buyu, Dylan, Masood, Mehrtash, Muhammad, Saeed, Salman and Sarah.

I would like to thank Prof. Vladlen Koltun for offering me a doctoral internship at Intel Labs for 6 months. I am deeply indebted to him for being a great mentor and introducing me to exciting new problems. I would also like to thank my collaborators, Stephan Richter, Alexey Dosovitskiy and other colleagues at the Intel Labs for warmly welcoming me to their group.

Lastly, and most importantly, I feel no words in expressing my profound gratitude to my parents for their immense blessings and indispensable guidance throughout my life to achieve extraordinary success.



---

# Abstract

---

Structured decisions are often required for a large variety of image and scene understanding tasks in computer vision, with few of them being object detection, localization, semantic segmentation and many more. Structured prediction deals with learning inherent structure by incorporating contextual information from several images and multiple tasks. However, it is very challenging when dealing with large scale image datasets where performance is limited by high computational costs and expressive power of the underlying representation learning techniques. In this thesis, we present efficient and effective deep structured models for context-aware object detection, co-localization and instance-level semantic segmentation.

First, we introduce a principled formulation for object co-detection using a fully-connected conditional random field (CRF). We build an explicit graph whose vertices represent object candidates (instead of pixel values) and edges encode the object similarity via simple, yet effective pairwise potentials. More specifically, we design a weighted mixture of Gaussian kernels for class-specific object similarity, and formulate kernel weights estimation as a least-squares regression problem. Its solution can therefore be obtained in closed-form. Furthermore, in contrast with traditional co-detection approaches, it has been shown that inference in such fully-connected CRFs can be performed efficiently using an approximate mean-field method with high-dimensional Gaussian filtering. This lets us effectively leverage information in multiple images.

Next, we extend our class-specific co-detection framework to multiple object categories. We model object candidates with rich, high-dimensional features learned using a deep convolutional neural network. In particular, our max-margin and direct-loss structural boosting algorithms enable us to learn the most suitable features that best encode pairwise similarity relationships within our CRF framework. Furthermore, it guarantees that the time and space complexity is  $O(n * t)$  where  $n$  is the total number of candidate boxes in the pool and  $t$  the number of mean-field iterations. Moreover, our experiments evidence the importance of learning rich similarity measures to account for the contextual relations across object classes and instances. However, all these methods are based on precomputed object candidates (or proposals), thus localization performance is limited by the quality of bounding-boxes.

To address this, we present an efficient object proposal co-generation technique that leverages the collective power of multiple images. In particular, we design a

deep neural network layer that takes unary and pairwise features as input, builds a fully-connected CRF and produces mean-field marginals as output. It also lets us backpropagate the gradient through entire network by unrolling the iterations of CRF inference. Furthermore, this layer simplifies the end-to-end learning, thus effectively benefiting from multiple candidates to co-generate high-quality object proposals.

Finally, we develop a multi-task strategy to jointly learn object detection, localization and instance-level semantic segmentation in a single network. In particular, we introduce a novel representation based on the distance transform of the object masks. To this end, we design a new residual-deconvolution architecture that infers such a representation and decodes it into the final binary object mask. We show that the predicted masks can go beyond the scope of the bounding boxes and that the multiple tasks can benefit from each other.

In summary, in this thesis, we exploit the joint power of multiple images as well as multiple tasks to improve generalization performance of structured learning. Our novel deep structured models, similarity learning techniques and residual-deconvolution architecture can be used to make accurate and reliable inference for key vision tasks. Furthermore, our quantitative and qualitative experiments on large scale challenging image datasets demonstrate the superiority of the proposed approaches over the state-of-the-art methods.

**Keywords:** Deep structured models, Context modeling, Object (co-)detection, Instance-level semantic segmentation

---

# Contents

---

<b>Acknowledgements</b>	<b>vii</b>
<b>Abstract</b>	<b>ix</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Some Computer Vision Challenges . . . . .	3
1.1.1 Object Detection . . . . .	3
1.1.2 Object Co-detection . . . . .	3
1.1.3 Objectness . . . . .	4
1.1.3.1 Box Proposals . . . . .	4
1.1.3.2 Segment Proposals . . . . .	4
1.1.4 Semantic Segmentation . . . . .	5
1.1.5 Semantic Instance Segmentation . . . . .	6
1.2 Contributions . . . . .	6
1.3 Thesis Structure . . . . .	8
1.4 Publications . . . . .	9
<b>2 Background and Preliminaries</b>	<b>11</b>
2.1 Supervised Learning . . . . .	11
2.2 Structured Prediction . . . . .	14
2.2.1 Probabilistic Graphical Models . . . . .	16
2.2.1.1 Bayesian Networks . . . . .	17
2.2.1.2 Markov Network . . . . .	18
2.2.1.3 Conditional Random Fields . . . . .	19
2.2.2 Variational Inference . . . . .	22
2.3 Feature Engineering . . . . .	24
2.3.1 Hand-Crafted Features . . . . .	26
2.3.2 Deep Learned Features . . . . .	28
2.3.2.1 Data Processing Layers . . . . .	29
2.3.2.2 Activation Layers . . . . .	30
2.3.2.3 Utility Layers . . . . .	31
2.3.2.4 Loss Layers . . . . .	32
2.3.3 Optimization using Gradient Descent . . . . .	33

---

2.4	Popular Deep Networks . . . . .	35
2.4.1	Alex Net . . . . .	35
2.4.2	VGG Network . . . . .	36
2.4.3	Residual Network . . . . .	36
2.5	Object Detection and Instance Segmentation . . . . .	37
2.5.1	Deformable Part-based Models (DPM) . . . . .	37
2.5.2	Region-based Convolutional Neural Networks (R-CNN) . . . . .	39
2.5.3	Fast Region-based Convolutional Neural Networks (Fast R-CNN) . . . . .	40
2.5.4	Faster R-CNN . . . . .	42
2.5.5	Multi-task Network Cascades (MNC) . . . . .	42
2.6	Large Scale Image Datasets . . . . .	44
2.6.1	Pascal VOC . . . . .	44
2.6.2	Image-Net . . . . .	45
2.6.3	Microsoft COCO . . . . .	45
2.6.4	Cityscapes . . . . .	46
2.7	Evaluation Metrics . . . . .	47
2.8	Summary . . . . .	49
<b>3</b>	<b>Learning Category-Specific Object Similarity</b>	<b>51</b>
3.1	Introduction . . . . .	51
3.2	Putting Objects into Context . . . . .	53
3.2.1	Limitations of Existing Approaches . . . . .	53
3.2.2	Our Idea . . . . .	53
3.3	A Fully-Connected CRF for Co-Detection . . . . .	54
3.3.1	Object Candidate Generation . . . . .	55
3.3.2	CRF Formulation . . . . .	55
3.3.2.1	Unary Potentials. . . . .	57
3.3.2.2	Pairwise Potentials. . . . .	57
3.3.2.3	Efficient Co-detection . . . . .	58
3.3.3	Learning Object Similarity . . . . .	59
3.4	Experiments . . . . .	60
3.4.1	Datasets and Setup . . . . .	60
3.4.2	Results and Discussion . . . . .	61
3.4.2.1	Pedestrian Dataset. . . . .	61
3.4.2.2	Ford Car Dataset. . . . .	63
3.4.2.3	Human Co-Detection Dataset. . . . .	65
3.4.2.4	Scaling up. . . . .	68
3.5	Conclusion . . . . .	69



---

<b>4</b>	<b>Multiclass Structural Kernel Boosting</b>	<b>71</b>
4.1	Object Co-detection: Single Class to Multiclass . . . . .	72
4.2	Multi-class Objects in Context . . . . .	73
4.2.1	Our Idea . . . . .	74
4.3	Large-scale Object Co-detection . . . . .	76
4.3.1	Object Hypotheses Generation . . . . .	76
4.3.2	CRF for Multiclass Object Co-detection . . . . .	76
4.3.3	Unary Potential . . . . .	76
4.3.4	Pairwise Potential . . . . .	77
4.3.5	Efficient Inference for Co-detection . . . . .	78
4.4	Kernel Learning for Fully-connected CRFs . . . . .	79
4.4.1	Structural Boosting for Fully-connected CRFs . . . . .	79
4.4.2	Max-margin Structural Boosting . . . . .	81
4.4.2.1	Space-Time Complexity . . . . .	82
4.4.3	Direct-loss Structural Boosting . . . . .	83
4.5	Experiments . . . . .	83
4.5.1	Datasets and Setup . . . . .	84
4.5.2	Results on VOC 2007 . . . . .	86
4.5.2.1	Per-class Scores . . . . .	86
4.5.2.2	Multi-class Scores . . . . .	88
4.5.3	Results on VOC 2012 . . . . .	90
4.6	Qualitative Results on PASCAL VOC . . . . .	91
4.7	Conclusion . . . . .	94
<b>5</b>	<b>Class-agnostic Similarity Learning with a Deep Structured Network</b>	<b>95</b>
5.1	Generating Object Proposals . . . . .	95
5.1.1	Limitations of Existing Approaches . . . . .	96
5.1.2	Our Idea . . . . .	97
5.2	Co-Generating Object Proposals . . . . .	98
5.2.1	Deep CNNs for Individual Object Candidates . . . . .	100
5.2.2	Fully-connected CRF for Candidate Similarity . . . . .	101
5.2.3	Efficient Object Proposal Co-Generation . . . . .	102
5.3	Learning our Deep Structured Network . . . . .	102
5.3.1	Pre-training the Deep CNN Module . . . . .	103
5.3.2	End-to-end Learning with Mini-batches . . . . .	104
5.4	Experiments . . . . .	106
5.4.1	Datasets and Setup . . . . .	106
5.4.2	Results on VOC 2007 . . . . .	107
5.4.3	Results on Microsoft COCO . . . . .	109

---

5.5	Deep Co-Objectness and Detection . . . . .	111
5.6	Conclusion . . . . .	112
<b>6</b>	<b>Beyond Boxes: Boundary-Aware Instance Segmentation</b>	<b>115</b>
6.1	From Detection to Instance Segmentation . . . . .	115
6.2	Limitation of Existing Approaches . . . . .	116
6.2.1	Our Idea . . . . .	118
6.3	Boundary-aware Segment Prediction . . . . .	119
6.3.1	Boundary-aware Mask Representation . . . . .	120
6.3.2	Object Mask Network . . . . .	122
6.4	Learning Instance Segmentation . . . . .	122
6.4.1	Boundary-aware Instance Segmentation Network . . . . .	123
6.4.2	Network Learning and Inference . . . . .	124
6.5	Experiments . . . . .	125
6.5.1	Instance-level Semantic Segmentation . . . . .	125
6.5.1.1	Results on VOC 2012 . . . . .	126
6.5.1.2	Results on Cityscapes . . . . .	127
6.5.2	Segment Proposal Generation . . . . .	129
6.6	Ablation Studies . . . . .	131
6.6.1	Detailed Comparisons with MNC on Pascal VOC 2012 . . . . .	131
6.6.2	Object Mask Network: Comparison with MNC and DeepMask for Different Object Sizes . . . . .	131
6.6.3	Qualitative Comparison with MNC . . . . .	131
6.7	Conclusion . . . . .	133
<b>7</b>	<b>Conclusions and Future Work</b>	<b>135</b>
7.1	Contributions . . . . .	135
7.2	Future Directions . . . . .	137
	<b>Bibliography</b>	<b>139</b>

---

# List of Figures

---

1.1	<b>An illustration of a pedestrian detection system.</b> It takes a single image as input and provides a list of all pedestrians. . . . .	3
1.2	<b>An illustration of a pedestrian co-detection system.</b> It takes multiple images as input at test time and provides a collective list of all pedestrians. . . . .	4
1.3	<b>An illustration of class-agnostic box and segmentation proposal methods.</b> <b>Left:</b> An example with the selective search technique [Uijlings et al., 2013]. It takes a single image as input and provides a list of all potential objects. <b>Right:</b> An example with the sharp-mask technique [Pineiro et al., 2016]. It takes a single image as input and provides a list of all potential segments. . . . .	5
1.4	<b>An illustration of a semantic segmentation.</b> <b>Left:</b> An image from Pascal VOC 2012 [Everingham et al.], and <b>Right:</b> Ground-truth pixel-wise labeling. . . . .	5
1.5	<b>An illustration of semantic instance segmentation.</b> <b>Left:</b> An image from Pascal VOC 2012 [Everingham et al.]. <b>Right:</b> Ground-truth pixel-wise instance semantic labeling. . . . .	6
2.1	<b>Example of a Bayesian network</b> [Friedman et al., 1997] represented by a directed acyclic graph. The nodes correspond to random variables and the edges represent statistical dependencies between the variables. . . . .	17
2.2	<b>Example of a possible factor graph in probabilistic graphical models</b> [Nowozin and Lampert, 2011]. Here the nodes and edges represent the unary potentials and pairwise interactions respectively. <b>Left:</b> Markov random field; <b>Right:</b> Conditional random field. . . . .	19
2.3	This image shows the process of computing the <b>local binary patterns (LBP) features</b> as in [Wang and He, 1990]. . . . .	26
2.4	<b>Example of histogram of oriented gradients (HoG) features</b> [Dalal and Triggs, 2005]. Here the left column shows an original image and the right column shows the HoG features. . . . .	27
2.5	<b>An illustration of the Alexnet architecture</b> used in [Krizhevsky et al., 2012]. . . . .	35

---

2.6	<b>An illustration of the visual geometry group network VGG16 architecture</b> as used in [Simonyan and Zisserman, 2015]. . . . .	36
2.7	<b>An illustration of the residual building block</b> as used in residual neural network [He et al., 2015]. . . . .	37
2.8	<b>R-CNN: Object detection using regions-based CNN</b> [Girshick et al., 2014]. . . . .	40
2.9	<b>Fast R-CNN: Efficient object detection using regions-based CNN</b> [Girshick, 2015]. . . . .	41
2.10	<b>Faster R-CNN: Towards real-time object detection with region proposal networks</b> [Ren et al., 2015]. <b>Left:</b> A single, unified network for object detection, <b>Right:</b> Region Proposal Network (RPN). . . . .	42
2.11	<b>MNC: Multi-task network cascade</b> [Girshick et al., 2014]. <b>Left:</b> Traditional multi-task learning, <b>Right:</b> A 3-stage cascade as in MNC. . . . .	43
2.12	<b>An example image for each category in the Pascal 2012 dataset</b> [Everingham et al., 2010]. . . . .	44
2.13	<b>An illustration of the low-level filters learned using AlexNet</b> [Krizhevsky et al., 2012] when trained on Imagenet dataset [Russakovsky et al., 2015].	45
2.14	<b>Few examples of Cityscapes dataset</b> [Cordts et al., 2016] images and pixel-level level semantic segmentations. . . . .	46
3.1	<b>Overview of our category-specific object similarity learning method.</b> Left: Original input images and candidates generated with a DPM; Middle: Fully-connected CRF on the candidates and corresponding learned pairwise similarities; Right: Jointly detected objects by efficient inference in the fully-connected CRF (actual result). . . . .	55
3.2	<b>Sample object candidates generation results.</b> Left: Pedestrian dataset [Ess et al., 2007]; Middle: Ford Car dataset [Bao et al., 2012]; Right: Human Co-detection dataset [Shi et al., 2013]. . . . .	56
3.3	<b>Pedestrian co-detection results:</b> Examples of our co-detection results on test pairs of the Pedestrian dataset. Top two rows: Input image pairs (Green dash: our results, Red solid: DPM results); Bottom row: Precision-recall curves of our method and DPM for the three image pairs. . . . .	62
3.4	<b>Predicting similarity:</b> Sample similarity matrix obtained by applying our learned similarity function to one test pair of the Pedestrian dataset. Top: Input image pair; Middle: (Left) target (ground truth) similarity matrix, (Right) learned similarity matrix (brighter means more similar); Bottom: Normalized histograms of similarity scores for matched and non-matched candidate pairs. . . . .	63

- 
- 3.5 **Car co-detection Results:** Examples of our co-detection results on test pairs of the Ford Car dataset. Top two rows: Input image pairs (Green dash: our results, Red solid: DPM results); Bottom row: Precision-recall curves of our method and DPM for the three image pairs. . . . . 64
- 3.6 **Predicting similarity:** Sample similarity matrix obtained by applying our learned similarity function to one test pair of the Ford Car dataset. Left: Input image pair; Middle: (Top) target (ground truth) similarity matrix, (Bottom) learned similarity matrix (brighter means more similar); Right: Normalized histograms of similarity scores for matched and non-matched candidate pairs. . . . . 66
- 3.7 **Category-specific object co-detection precision-recall curves:** The precision-recall curves over all pairs (Blue solid: our results, Red solid: DPM results). Left: Pedestrian dataset ; Right: Ford car dataset. . . . . 67
- 3.8 **Sample Results:** Examples of our co-detection results on two sets from the Human Co-detection dataset. Top: Input image set overlaid with detection output (Green dash: our results, Red solid: DPM results); Bottom: Precision-recall curves of our method and DPM for the two sets. 67
- 3.9 **Predicting similarity:** Sample similarity matrix obtained by applying our learned similarity function to one test set in the Human Co-Detection dataset. Top: Input images (three examples); Middle: (Left) target (ground truth) similarity matrix, (Right) learned similarity matrix (brighter means more similar); Bottom: Normalized histograms of similarity scores for matched and non-matched candidate pairs. . . . . 68
- 4.1 **Top:** Conventional single-class object co-detection vs. **Bottom:** our multi-class object co-detection approach. . . . . 72
- 4.2 **Precision/Recall curves:** Performance comparison on four representative categories using the VOC 2007 dataset. Blue curve represents the R-CNN (without bounding-box regression) [Girshick et al., 2014], whereas, Red curve shows our method results. . . . . 87
- 4.3 **Sensitivity and Impact Analysis:** Overall detailed performance comparison using different metrics (i.e., occlusion (acc), truncated (trn), size, aspect ratio (asp), view point and part visibility). The black dashed line indicates the overall average normalized precision  $AP_N$ . . . 88
- 4.4 **False positive analysis using all the vehicles category.** . . . . . 89

- 
- 4.5 **CoDeT-G-DL detection trends.** Following [Hoiem et al., 2012], we show the evaluation of the type of detection as the number of detections increases; The white areas correspond to the correct detections; The blue areas represent the detections with localization error; The red areas correspond to confusion with a similar category; The green areas represent the confusion with a dissimilar category. Finally, the line plots show the recall as a function of the number of objects (dashed=weak localization, solid=strong localization). . . . . 91
- 4.6 **CoDeT-G-DL vs. R-CNN-BB (Normalized) Scores.** For each image, we show the changes in scores before and after performing dense CRF inference with our learned structured kernels. Columns 1-3 depict examples where our CoDeT-G-DL algorithm improved the scores of R-CNN-BB. Example bounding boxes with increased confidence are shown in column 1-2, column 3 shows the cases where bounding box confidence is decreased. Column 4 shows examples where the R-CNN-BB scores are better than ours. Note that, as in the false positive analysis of Fig. 4.8, these cases correspond to either parts of objects, or multiple objects of the same class. . . . . 92
- 4.7 **CoDeT-G-DL top detections.** We show the top-scoring detections in decreasing order from left to right for 10 representative classes. If an image has multiple detections, only its top-scoring detection is shown. Note that our top-scoring detections correspond to accurate object localizations. . . . . 93
- 4.8 **CoDeT-G-DL top false positives.** We show the top-scoring false positives in decreasing order from left to right for 8 representative classes. Note that our top scoring false positives belong to two main categories: **(i)** Parts of objects are confused with complete objects due to high similarity with objects that are truly partially observed; and **ii)** Bounding boxes containing multiple objects of the same category are confused with single objects due to high similarity with other boxes truly containing single objects. Note, however, that our top-scoring false positives do not contain truly absurd detections. . . . . 93
- 5.1 **Overview of our deep structured network for object proposal co-generation.** Our model consists of one deep CNN module per object candidate, linked by a fully-connected CRF. . . . . 99

- 
- 5.2 **Detailed architecture of our deep structured network for object proposal co-generation.** Each input image first goes through a series of convolutional layers, followed by Region-of-Interest (RoI) pooling corresponding to the different candidates in the image. Each candidate then passes through several fully-connected layers to predict unary features, pairwise features and bounding box location offsets. The features are finally employed in a fully-connected CRF. During training, our model makes use of a multi-task loss. . . . . 100
- 5.3 **Pascal VOC 2007 test:** We compare our method with state-of-the-art object proposal baselines according to the criteria of [Hosang et al., 2016]. **Left: Recall v.s. IoU threshold.** These recall curves were generated using the highest-scoring 10, 100, 1000 and 10000 object proposals, respectively. **Right: Recall v.s. Number of Proposals.** The first plot shows the AR, and the remaining recall curves were generated using IoU thresholds of 0.5, 0.7 and 0.8, respectively. In all the plots, the dashed lines correspond to our co-generation results, in blue when using Selective Search candidates (**Co-Obj (SS)**) and in black when using Bing candidates (**Co-Obj (B)**). The baselines correspond to Bing (**B**), EdgeBoxes (**EB70**), Geodesic (**G**), MCG (**M**) and SelectiveSearch (**SS**). These results clearly evidence the benefits of our co-generation approach. . . . . 108
- 5.4 **MS COCO validation:** We compare our method with state-of-the-art object proposal baselines according to the criteria of [Hosang et al., 2016]. **Top: Recall v.s. IoU threshold.** These recall curves were generated using the highest-scoring 10, 100, 1000 and 10000 object proposals, respectively. **Bottom: Recall v.s. Number of Proposals.** The first plot shows the AR, and the remaining recall curves were generated using IoU thresholds of 0.5, 0.7 and 0.8, respectively. In all the plots, the dashed lines correspond to our co-generation results, in blue when using EdgeBox candidates (**Co-Obj (EB)**) and in black when using Bing candidates (**Co-Obj (B)**). The baselines correspond to Bing (**B**), EdgeBoxes (**EB70**), Geodesic (**G**), MCG (**M**) and SelectiveSearch (**SS**). These results again evidence the benefits of our co-generation approach. . . . 110
- 5.5 **Sensitivity and impact analysis:** Overall detailed performance comparison using different metrics (i.e., occlusion, truncated, size, aspect ratio, view point and part visibility). The black dashed line indicates the overall average normalized precision  $AP_N$ . . . . . 112

- 
- 5.6 **Deep co-objectness and detection trends.** Following [Hoiem et al., 2012], we show the evolution of the type of detection as the number of detections increases; The white areas correspond to the correct detections; The blue areas represent the detections with localization error; The red areas correspond to confusion with a similar category; The green areas represent the confusion with a dissimilar category. Finally, the curves show the recall as a function of the number of objects (dashed=weak localization, solid=strong localization). . . . . 113
- 6.1 **Traditional instance segmentation vs our shape based representation.** **Left:** Original image and ground-truth segmentation. **Middle:** Given a bounding box, traditional methods directly predict a binary mask, whose extent is therefore limited to that of the box and thus suffers from box inaccuracies. **Right:** We represent the object segment with a multi-valued map encoding the truncated minimum distance to the object boundary. This can be converted into a mask that goes beyond the bounding box, which makes our approach robust to box errors. . . . . 118
- 6.2 **Left:** Truncated distance transform. **Right:** Our deconvolution-based shape-decoding network. Each deconvolution has a specific kernel size ( $ks$ ), padding ( $p$ ) and stride ( $s$ ). Here,  $K$  represents the number of binary maps. . . . . 120
- 6.3 **Left:** Detailed architecture of our boundary-aware instance segmentation network. An input image first goes through a series of convolutional layers, followed by an RPN to generate bounding box proposals. After RoI warping, each proposal passes through our OMN to obtain a binary mask that can go beyond the box's spatial extent. Mask features are then extracted and used in conjunction with bounding-box features for classification purpose. During training, our model makes use of a multi-task loss encoding, bounding box, segmentation and classification errors. **Right:** 5-stage BAIS network. The first three stages correspond to the model on the left. The five-stage model then concatenates an additional OMN and classification module to these three stages. The second OMN takes as input the classification score and refined box from the previous stage, and outputs a new segmentation with a new score obtained via the second classification module. The weights of the OMN and classification modules in both stages are shared. . . . . 123



---

6.4	<b>Qualitative results on Cityscapes</b> From left to right, we show the input image, our instance level segmentations and the segmentations projected onto the image with class labels. Note that our segmentations are accurate despite the presence of many instances. . . . .	128
6.5	<b>Failure cases.</b> The typical failures of our approach correspond to cases where one instance is broken into multiple ones. . . . .	129
6.6	<b>Recall v.s. IoU threshold on Pascal VOC 2012.</b> The curves were generated using the highest-scoring 10, 100 and 1000 segmentation proposals, respectively. In each plot, the solid line corresponds to our OMN results. Note that we outperforms the baselines when using the top 10 and 100 proposals. For 1000, our approach still yields state-of-the-art results at high IoU thresholds. . . . .	130



---

# List of Tables

---

3.1	<b>Pedestrian co-detection with stereo pairs:</b> Comparison of our approach with state-of-the-art co-detection methods on Pedestrian dataset using stereo pairs. . . . .	61
3.2	<b>Pedestrian co-detection with random pairs:</b> Comparison of our approach with state-of-the-art co-detection methods on Pedestrian dataset using random pairs. . . . .	61
3.3	<b>Car co-detection with stereo pairs:</b> Comparison of our approach with state-of-the-art co-detection methods on the Ford Car dataset using stereo pairs. . . . .	64
3.4	<b>Car co-detection with random pairs:</b> Comparison of our approach with state-of-the-art co-detection methods on the Ford Car dataset using random pairs. . . . .	65
3.5	<b>Human co-detection:</b> Comparison of our approach with state-of-the-art co-detection methods on the HCD dataset . . . . .	65
4.1	<b>Detection average precision(%) on the PASCAL VOC 2007 test set using AlexNet.</b> Columns 1-2 shows the co-detection baselines i.e. MLRR [Guo et al., 2013] and MCOL [Desai et al., 2009]. Columns 3-4 provide the baseline state-of-the-art results for detection [Girshick et al., 2014]. R-CNN (without bounding-box regression); R-CNN BB (with bounding-box regression). Column 5 provides results of Chapter 3 co-detection method with deep network unary potentials. Columns 6-8 show co-detection performance. CoDet-G (R-CNN-BB with geometric context model learning); CoDet-G-LA (Kernel selection using max-margin learning); CoDet-G-DL (Kernel selection using direct loss minimization). . . . .	85
4.2	<b>Detection average precision(%) on the PASCAL VOC 2007 test set using VGG16.</b> Column 1 shows the results of the Fast-RCNN baseline and Column 2 shows our method results using VGG16. . . . .	86

- 
- 4.3 **Multi-class average precision(%) on the PASCAL VOC 2007 test set.** CoDet-G-DL (Kernel selection using direct loss minimization). We constructed the baseline curve for R-CNN-BB (with bounding-box regression) [Girshick et al., 2014] by pooling the detections across all object classes and images when computing the PR curves. Our model clearly provides a noticeable boost in overall performance. . . . . 89
- 4.4 **Detection average precision(%) on the PASCAL VOC 2012 test set.** Columns 1-2 provide the baseline state-of-the-art detection results! [Girshick et al., 2014]. R-CNN (without bounding-box regression); R-CNN BB (with bounding-box regression). Columns 3-4 show our co-detection performance. CoDet-G (R-CNN-BB with geometric context model learning); CoDet-G-DL (Kernel selection using direct loss minimization). . . . 90
- 5.1 **AR analysis on the PASCAL VOC 2007 test set:** We compare our method with state-of-the-art object proposal baselines according to the criteria of [Lin et al., 2014]. The results of our approach are provided in Rows 7-8 when using Bing and Selective Search to generate the initial candidates, respectively. The AR for small, medium and large objects were computed for 100 proposals. Note that our co-generation approach outperforms the state-of-the-art baseline in all metrics. The difference in speed between two versions of our approach is due to the fact that Bing yields a larger candidate pool than Selective Search. . . . 108
- 5.2 **Using different initial proposal methods:** Rows 1-2 show the baseline object proposal methods. Rows 3-6 show our results using various candidate generation options at training and test time. . . . . 109
- 5.3 **AR analysis on the MS COCO validation set:** We compare our method with state-of-the-art object proposal baselines according to the criteria of [Lin et al., 2014]. The results of our approach are provided in Rows 7-8 when using Bing and EdgeBox to generate the initial candidates, respectively. The AR for small, medium and large objects were computed for 100 proposals. Note that our co-generation approach outperforms the state-of-the-art baseline in all metrics. . . . . 110
- 5.4 **Detection Average Precision(%) on the PASCAL VOC 2007 test set:** Columns 1-3 show the multi-class object detection state-of-the-art results for Fast RCNN (SS) [Girshick, 2015], CoDet-G-DL (SS) of Chapter 4, RPN [Ren et al., 2015] and Fast RCNN (SS) respectively. The results of our approach are provided in Row 4. . . . . 111

---

6.1	<b>Instance-level semantic segmentation on Pascal VOC 2012.</b> Comparison of our method with state-of-the-art baselines. The results of [Hariharan et al., 2014, 2015] are reproduced from [Dai et al., 2016b]. . . . .	126
6.2	<b>Influence of the number of stages during training.</b> Whether trained using 3 stages or 5, our approach outperforms both MNC baselines. . . . .	127
6.3	<b>Instance-level semantic segmentation on Cityscapes.</b> We compare our method with the state-of-the-art baselines on the Cityscapes test set. These results were obtained from the online evaluation server. . . . .	127
6.4	<b>Detailed comparison with DTW [Bai and Urtasun, 2017] at AP(50%).</b> Note that our approach outperforms this baseline on all the classes except truck and motorcycle for the Cityscapes test dataset. . . . .	128
6.5	<b>Comparison with MNC-new on the Cityscapes validation data.</b> Note that our approach outperforms this baseline, thus showing the importance of allowing the masks to go beyond the box proposals. . . . .	128
6.6	<b>Evaluation of our OMN on the PASCAL VOC 2012 validation set.</b> We compare our method with state-of-the-art segmentation proposal baselines according to the criteria of [Hariharan et al., 2014; Lin et al., 2014]. Note that our approach outperforms the state-of-the-art methods for the top 10 and 100 segmentation proposals, which correspond to the most common scenarios when later processing is involved, e.g., instance level segmentation. . . . .	130
6.7	<b>Detailed Evaluation on the Pascal VOC 2012 validation set.</b> We report results with mask level IoU thresholds of 0.5 and 0.7. Note that our method outperforms MNC [Dai et al., 2016b] for most classes and achieves an improvement of 2.1% over MNC at a 0.7% IoU threshold . . . . .	130
6.8	<b>Object size analysis on the Pascal VOC 2012 validation set.</b> We compare our method with state-of-the-art segmentation proposal baselines according to the criteria of [Lin et al., 2014]. The AR for small, medium and large objects were computed for 100 proposals. Note that our object mask network outperforms the state-of-the-art baselines for large objects by a significant margin. . . . .	131
6.9	<b>Qualitative results on Pascal VOC 2012.</b> From left to right, we show: the original image, the segmentations of MNC, these results projected on the image, our segmentations, our results projected on the image. Note the better quality of our results. . . . .	132



---

# List of Algorithms

---

2.1	Mean field in fully connected CRFs [Krähenbühl and Koltun, 2011] . . .	24
4.1	Structural Kernel Learning Algorithm . . . . .	82
5.1	Mean-field Gradient for Classification Loss . . . . .	106





# Introduction

---

Image and scene understanding is one of the most important unsolved problems in computer vision, and the complexity of real environments makes it challenging to interpret natural images and videos. Diagnosing brain cancer by detecting and analysing medical images, safe navigation for an autonomous vehicle and landing the Curiosity rover on Mars by detecting and avoiding the obstacles using vision sensors are only a few examples of the numerous applications that all require artificial systems that mimic the human vision system to accurately extract high-level information from images. Decoding natural scenes into objects and their interactions relies on addressing the intermediate vision challenges of object detection, recognition and segmentation. In recent years, the availability of large scale natural image datasets, e.g., Image-Net [Krizhevsky et al., 2012], Pascal VOC [Everingham et al., 2010; Everingham et al.], Microsoft COCO [Lin et al., 2014] and Cityscapes [Cordts et al., 2016], and the statistical models inspired by efficient machine learning techniques have shown impressive progress by achieving compelling and accurate results in solving the most complex visual computing challenges, but they are still far from matching the human brain’s remarkable performance.

Over the years, the techniques proposed to achieve the intermediate goals of image and scene understanding were mostly focused on single image techniques and much work has been done to address the problem of object detection by either class-specific methods [Felzenszwalb et al., 2010] or using object proposal generation methods [Girshick, 2015; Ren et al., 2015; He et al., 2015]. Generating class-independent object proposals, based on objectness measures, facilitates detection by proposing a smaller number of interesting candidate regions. Whether class-specific or class-independent, most existing methods make decisions on individual objects to perform detection. Such a myopic view, however, is too restrictive as it ignores all structural and contextual information, and the appearance of natural objects is often ambiguous due to the lack of context, poor resolution, occlusions, or challenging lighting conditions.

**Context-aware deep structured models** are important because they tend to learn semantic contents accurately and efficiently by analysing an entire image or multiple images jointly. Recently, small scale context-aware models in image parsing have gained wide popularity. Context-aware foreground-background image segmentation [Nowozin and Lampert, 2011; Blaschko and Lampert, 2012] is the task of predicting pixel-level correspondences given some image or video data. In structured prediction, probabilistic graphical models provide a powerful framework to learn the dependence structure between variables, which can be represented as a graph. Typically, vertices of graph represents the nodes (pixels or superpixels) and edges encode the neighborhood relations. The task is to produce a consistent labeling of all the vertices. Modeling and learning these structured and contextual models using large-scale data is very challenging. A number of methods have been proposed [Bengio, 2012] to obtain computationally feasible solutions, but are complicated by several factors. In particular, given a large graph with millions of nodes, the input and output spaces tend to have exponential number of possibilities and performing inference using millions of interrelated variables is difficult and time consuming.

**High performance computing** using a graphics processing unit (GPU) is vital and is at the heart of the emerging visual computational techniques. GPU-accelerated deep learning utilizes parallel computation on GPU to significantly reduce the time it takes to learn these models. The advent of the NVIDIA CUDA deep neural network library (cuDNN) [NVIDIA, 2015] has led to a surge of interest for research in increasingly effective techniques for object detection, localization and recognition. Learning an end-to-end deep structured model requires an enormous amount of computation and time, and proper utilization of GPU computation power makes it possible to learn these useful models efficiently.

In this thesis, we focus on structured prediction and propose and validate efficient deep structured models for context-aware object detection, localization and segmentation. Context-aware image analysis has a major impact on applications that rely on the analysis of large scale multi-image data. Visual question answering, image based content retrieval and video based self-learning cannot be achieved without knowing the context. In addition, context aware instance detection/segmentation is a fundamental research topic since it constitutes a critical step for many complex processes, such as segmenting MRI images for cancer treatment, multiple camera fusion for tracking, autonomous navigation and free-viewpoint film production.

Before presenting our contributions, we briefly review a few examples of computer vision challenges that we intend to study, in this thesis, by using context-aware deep structured models.

## 1.1 Some Computer Vision Challenges

Here we briefly discuss some core computer vision challenges that are building blocks for image understanding and scene analysis. We will formally define the methods developed over the years in Chapter 2. However, for better understanding, we first briefly outline some of the applications and how they are related to each other in this section.

### 1.1.1 Object Detection

Object modeling and recognition has been one of the fundamental problems in computer vision since its early days [Dickinson et al., 2009]. Numerous methods have been proposed to tackle this task which can be categorized into two broad categories: one that rely on hand-crafted representation, *e.g.*, [Viola and Jones, 2004], and others that automatically learn representations from data, *e.g.*, [Girshick, 2015]. In particular, object detection has evolved into a core challenge in vision research [Everingham et al., 2010], and much progress has been made recently due to advances in deformable part-based object models, *e.g.*, DPM [Felzenszwalb et al., 2010], as well as in deep network based models [Girshick et al., 2014].



Figure 1.1: An illustration of a pedestrian detection system. It takes a single image as input and provides a list of all pedestrians.

Given an image and an object category list, the goal of object detection is to find all the relevant individual objects in the image. An example of this is illustrated in Figure 1.1. These predicted objects usually have a unique location in the input image and are marked with a bounding-box in the output image. Object detection in the wild is a challenging task because of the complexity of unconstrained environments, large viewpoint variations, object occlusions, motion, deformations and lighting variations.

### 1.1.2 Object Co-detection

Object co-detection methods [Bao et al., 2012; Guo et al., 2013; Shi et al., 2013] were recently introduced to exploit the collective power of a set of images. In contrast to

object detection, object co-detection techniques jointly process multiple images at test time and find all objects collectively. This is illustrated in Figure 1.2. The benefits of jointly processing multiple images is that it can help model complex environments with large viewpoint variations, and can overcome object detection limitations, such as occlusions, deformations and lighting variations.

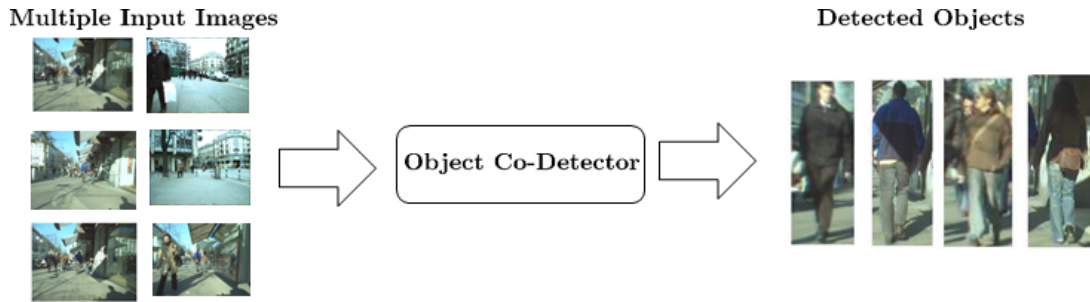


Figure 1.2: **An illustration of a pedestrian co-detection system.** It takes multiple images as input at test time and provides a collective list of all pedestrians.

### 1.1.3 Objectness

Traditionally, object (co-)detection methods take a scanning window approach and exhaustively search for object candidates at every location and scale in an image [Viola and Jones, 2004; Dalal and Triggs, 2005; Felzenszwalb et al., 2010]. More recently, objectness criteria have been used to propose potential candidates, which drastically reduces the search space [Endres and Hoiem, 2010; Alexe et al., 2010; Uijlings et al., 2013]. Objectness, however, is very challenging to adequately represent, since it has to account for the huge intra-class variations of the general *object* category, while still being able to differentiate it from the *background* class. As a matter of fact, these challenges remain unsolved even when detecting specific objects. Objectness methods are generally categorized based on the type of output they produce.

#### 1.1.3.1 Box Proposals

Box proposal methods provide just a bounding-box around a potential object instance. This is often desired when learning a class-generic object detector. An illustration using selective search [Uijlings et al., 2013] is shown in Figure 1.3 (Left).

#### 1.1.3.2 Segment Proposals

Segmentation proposal methods provide not only the bounding-box around a potential object instance but also a pixel-level labeling that separates the instance from its

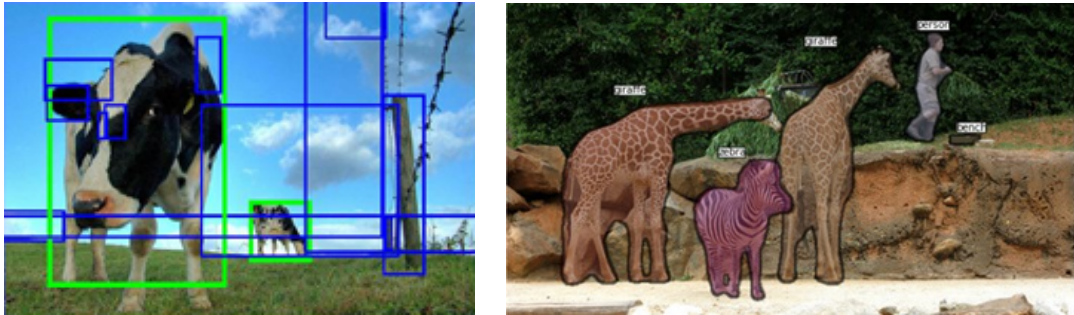


Figure 1.3: **An illustration of class-agnostic box and segmentation proposal methods.** **Left:** An example with the selective search technique [Uijlings et al., 2013]. It takes a single image as input and provides a list of all potential objects. **Right:** An example with the sharp-mask technique [Pinheiro et al., 2016]. It takes a single image as input and provides a list of all potential segments.

background inside that bounding-box. This is often desired when designing an instance segmentation method. An illustration using the sharp-mask method [Pinheiro et al., 2016] is shown in Figure 1.3 (Right).

#### 1.1.4 Semantic Segmentation

Semantic segmentation is the task of partitioning an input image into segments related to multiple categories. In contrast to object (co-)detection techniques, it provides a detailed pixel-level view of the scene and is useful for many scene understanding tasks. A limitation of semantic segmentation is that it ignores the boundaries between the individual objects. A sample semantic segmentation is shown in Figure 1.4. Note that each object category is represented by a different color.



Figure 1.4: **An illustration of a semantic segmentation.** **Left:** An image from Pascal VOC 2012 [Everingham et al.], and **Right:** Ground-truth pixel-wise labeling.

### 1.1.5 Semantic Instance Segmentation

Semantic instance segmentation is the task of segmenting/detecting individual objects along with their exact pixel-wise outline in the scene. In contrast to semantic segmentation, it distinguishes the individual instances of the same category. This is a very challenging task since the number of objects are not known and varies from image to image. Furthermore, instance segmentation is often termed as simultaneous detection and segmentation and provides useful insights about the objects in context. A sample instance segmentation is shown in Figure 1.5. Note that each object instance is represented by a different color.



Figure 1.5: **An illustration of semantic instance segmentation.** **Left:** An image from Pascal VOC 2012 [Everingham et al.]. **Right:** Ground-truth pixel-wise instance semantic labeling.

## 1.2 Contributions

In this thesis, we introduce new algorithms to learn context-aware models for improving the accuracy of object detection, localization and instance segmentation. The goal of the proposed methods is to introduce algorithms for efficiently processing large scale image datasets with an emphasis on the quality and robustness of the results when compared with the state-of-the-art methods. The contributions are four-fold: first, we present a novel class-specific similarity learning algorithm that can be used to jointly process large scale datasets for efficient object co-detection; second, we extend our method to multi-class similarity learning and further leverage deep features and geometric scene layout, thus providing a novel algorithm for large scale object detection in context; third, we introduce a class-agnostic object similarity formulation for overcoming the poor localization performance of object detectors and provide a solution that learns context jointly within a deep structured network;

---

fourth, we develop a boundary-aware instance segmentation approach which not only can predict segmentation mask inside a bounding-box but also can go beyond it, which is a major limitation of most existing state-of-the-art approaches. Specifically, we discuss how our algorithms address the limitations of the existing ones.

- **Learning Category-Specific Object Similarity:** We introduce a principled formulation of object co-detection as an efficient inference in a fully-connected CRF whose nodes represent object candidates and edges encode the object similarity. We demonstrate that modeling the similarity between pairs of candidates as a weighted mixture of Gaussian kernels allows us to efficiently perform inference in large graphs optimally. As a result, our approach can effectively leverage the information in multiple images to improve detection accuracy, thus outperforming existing co-detection techniques on benchmark datasets.
- **Multiclass Structural Kernel Boosting:** We extend our co-detection algorithm to handle multiple object classes simultaneously in large scale image datasets. The unified framework to jointly learn a CRF and interclass similarity based on deep features allows us to yield better performance. Furthermore, our structural boosting strategy lets us benefit from rich, high-dimensional features to learn the contextual relationships within our CRF framework. Empirical comparisons show that our method outperforms state-of-the-art object detection methods on Pascal VOC 2007 and 2012.
- **Class-agnostic Similarity Learning with a Deep Structured Network:** We present an end-to-end learning algorithm to jointly generate object proposals from multiple images, thus leveraging the collective power of multiple object candidates. In contrast to most existing methods, our method is based on a deep structured network that jointly predicts the objectness scores and the bounding box locations of multiple object candidates by extracting features that model the individual object candidates and the similarity between them. Empirical comparisons show the benefits of our approach over the state-of-the-art methods that generate object proposals individually.
- **Beyond Boxes: Boundary-Aware Instance Segmentation:** We develop a distance transform-based mask representation that allows us to predict instance segmentations beyond the limits of initial bounding boxes. Subsequently, we present an efficient approach to infer and decode this representation with a fully-differential Object Mask Network (OMN) which inherently relies on a residual-deconvolutional architecture. This modular approach allows us to employ this OMN module into an instance segmentation framework. To the best

of our knowledge, this constitutes the first method that can segment instances beyond the bounding-box limit. Furthermore, our quantitative and qualitative experiments demonstrate the superiority of proposed approach over the state-of-the-art instance-level semantic segmentation methods.

### 1.3 Thesis Structure

In this chapter, we provided a brief overview of challenging applications that benefits from our research. Following this introduction chapter, the remaining chapters of the thesis are summarized below.

**Chapter 2.** This chapter starts with an overview of supervised learning and introduces the basic notations. Furthermore, we provide an overview of fundamental structured learning models and review relevant state-of-the-art techniques. We also present techniques to compute hand-engineered representations as well as automatically learned deep representations. In particular, we discuss conditional random fields to incorporate structure and contextual information, formalize feature learning and later discuss the popular deep convolutional neural network architectures. In the end, an overview of large scale challenging image datasets and respective evaluation metrics is also provided. Note that the material included in this chapter is intended for readers who are not familiar with related concepts and methods used in rest of the thesis. However, readers with relevant background may skip this chapter.

**Chapter 3.** This chapter is based on our work [Hayder et al., 2014] to learn category-specific object similarity. Here, we formulate object co-detection as inference in a fully-connected conditional random field (CRF) whose edges model the similarity between object candidates. Furthermore, we demonstrate the scalability of our algorithm compared to existing co-detection techniques that rely on exhaustive or greedy search.

**Chapter 4.** This chapter is based on our work [Hayder et al., 2015] that extends our model in Chapter 3 to learn multi-category object similarity. Here, our structural boosting strategy lets us benefit from rich, high-dimensional features to learn the object relationships within our CRF framework, thus lead to superior object (co-)detection results. Note that, in last chapter, although we start with simple-yet-effective hand-crafted object features to learn the similarity, here, we model object instances with features learned using deep neural network to improve the quality of our similarity learning algorithms.



---

**Chapter 5.** This chapter is based on our work [Hayder et al., 2016] that learns a class-agnostic geometric model and instance similarity jointly to improve the accuracy of object detection and localization. Here, we develop an end-to-end learning algorithm that, by unrolling the CRF inference procedure, lets us backpropagate the loss gradient throughout the entire structured network, thus learning better contextual features for detection.

**Chapter 6.** This chapter is based on our work [Hayder et al., 2017] that introduces a novel representation based on the distance transform of the object masks. Here, we design an object mask network with a new residual-deconvolution architecture that infers such a representation and decodes it into the final binary object mask. This allows us to predict masks that go beyond the scope of the bounding boxes and are thus robust to inaccurate object candidates.

**Chapter 7.** In this chapter, we draw conclusions based on the observations made in previous chapters and suggest several future research directions.

## 1.4 Publications

The contributions described in this thesis have previously appeared in the following publications.

- **Z. Hayder**, X. He and M. Salzmann, "Boundary-aware Instance Segmentation", *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- **Z. Hayder**, X. He and M. Salzmann, "Learning to co-generate object proposals with a deep structured network", *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
- **Z. Hayder**, X. He and M. Salzmann, "Structural kernel learning for large scale multiclass object co-detection", *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.
- **Z. Hayder**, M. Salzmann and X. He, "Object co-detection via efficient inference in a fully-connected crf", *The European Conference on Computer Vision (ECCV)*, September 2014.



---

# Background and Preliminaries

---

The role of this chapter is to provide fundamental concepts and an overview of structured models. This chapter starts with an overview of the basic notation and terminology used for learning supervised models. We build upon this and provide an overview of fundamental structured learning models and review relevant state-of-the-art techniques. We also present techniques to compute hand-engineered representations as well as automatically learned deep representations. In particular, we discuss conditional random fields to incorporate structure and contextual information, deep feature learning and later discuss the popular deep convolutional neural network architectures. In the end, an overview of large scale challenging image datasets and respective evaluation metrics are also provided.

## 2.1 Supervised Learning

The idea of inferring from examples, or labeled data, is typically formulated as supervised learning. In supervised learning, the main focus is on accurate prediction and to perform well on previously unseen data. Generally, the dataset  $\mathcal{D}$  consists of  $n$  data and label pairs  $\{(\mathbf{X}^n, \mathbf{y}^n)\} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_n, y_n)\}$ , where each  $\mathbf{x}_i \in \mathcal{R}^d$  is a real-valued vector of  $d$  dimensions representing features of a single data point, and  $y_i$  is the given label for that datum.

Given a training set  $\mathcal{D}$  with input attributes set  $\mathbf{X}^n = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$  and a nominal target attribute  $\mathbf{y}^n = \{y_1, y_2, \dots, y_n\}$  from an unknown fixed distribution  $\mathcal{P}$  over the labeled instance space, the goal is to induce an optimal classifier with minimum generalization error.

The dataset  $\mathcal{D}$  is divided into multiple sets to quantify if the model generalizes well to previously unseen examples. In this sense, one distinguishes between the data that is used to train a model, validate its performance, and to test the trained & validated model. These sets are commonly named as the training  $\{\mathbf{X}_{train}, \mathbf{y}_{train}\}$ , validation  $\{\mathbf{X}_{val}, \mathbf{y}_{val}\}$  and test  $\{\mathbf{X}_{test}, \mathbf{y}_{test}\}$  set.

Given a training dataset, the goal of supervised learning is to learn the complex relationship or a function between input  $\mathbf{X}$  and the output  $\mathbf{y}$ , i.e.,  $f_\theta : \mathbf{X} \rightarrow \mathbf{y}$ , such that  $\mathbf{y}^{pred} = f(\mathbf{X}|\theta)$  is a good predictor for the corresponding  $\mathbf{y}$ . For historical reasons, this function  $f$  is called the hypothesis and the model parameters are denoted by  $\theta$ . In supervised learning, our interest is learning a good hypothesis  $f$  such that it not only accurately classifies the  $\mathbf{x}_i$  in the training data but also classifies new samples  $\mathbf{x}$  from the distribution  $(\mathbf{x}, y) \sim \mathcal{P}$  reasonably well. In a typical supervised learning scenario, learning a good hypothesis  $f$  from data is an iterative process and it usually performs the following basic steps

- Given  $\mathbf{X}_{train}$ , learn a model  $\hat{f}_\theta$  that best predicts the label vector  $\mathbf{y}_{train}$ .
- Validate the model using  $\mathbf{X}_{val}$  and find the best parameters  $\theta$ .
- Test the model using  $\mathbf{X}_{test}$  and compute the model accuracy.

For example, in a license plate recognition system (LPR),  $\mathbf{x}_i$  might be a vector representing a scanned image of a car's number plate with English letters and many numerical digits. The corresponding label  $\mathbf{y}_i$  might be a the supervised learning output of the image as a vector of alphanumeric characters. The goal is to learn the inherent relationship in the training set and to make intelligent guesses about the labels for the image by making intelligent guesses from the training set.

**Loss function:** In learning, a loss function  $\mathcal{L}(\mathbf{X}, \mathbf{y}, f(\mathbf{X}))$  is very important and is explicitly specified to gauge the accuracy of a classifier  $f$  and it is a measure of how well  $f$  classifies  $\mathbf{X}$  when the true label is  $\mathbf{y}$ . In an ideal scenario when the learned function accurately predicts all the data-points, e.g.,  $f(\mathbf{X}) = \mathbf{y}$ , the loss is zero, i.e.,  $\mathcal{L}(\mathbf{X}, \mathbf{y}, f(\mathbf{X})) = 0$ .

The most common loss is called the 0/1 loss defined as

$$\mathcal{L}^{0/1}(\mathbf{X}, \mathbf{y}, f(\mathbf{x})) = \mathcal{I}(\mathbf{y} \neq f(\mathbf{X})), \quad (2.1)$$

where  $\mathcal{I}(\cdot)$  denotes the indicator function, i.e.,  $\mathcal{I}(t) = 1$  if  $t$  is true and 0 otherwise. The empirical risk  $\mathcal{R}_{(\mathcal{L}, \mathcal{P})}(f)$  of the classifier  $f$  can be calculated as

$$\mathcal{R}_{(\mathcal{L}, \mathcal{P})}(f) = \frac{1}{n} \sum_{i=1}^n \mathcal{L}(\mathbf{x}_i, y_i, f(\mathbf{x}_i)). \quad (2.2)$$

The generalization error is defined as the misclassification rate over the distribution  $\mathcal{P}$ . The classification accuracy of a classifier is one minus the expected loss or generalization error. The training error is defined as the percentage of examples in the training set correctly classified by the classifier. This is often calculated on the

entire training set, although, in recently famous deep learning frameworks, this is usually calculated on mini- batches.

**Regularization:** Choosing a classifier  $f$  with the minimum empirical risk could be misleading due to two main reasons. First of all, it assigns equal weight to all data samples irrespective of their label frequency. Secondly, a naive classifier might overfit to the training data, i.e., it might have zero classification loss on the training dataset  $\mathcal{D}$ , but very high loss on test set, which occurs due to poor generalization performance of the classifier on unseen data. Therefore, a *regularization* term is often introduced to avoid the problem of over-fitting and as a trade off between the risk of the classifier and the complexity of the hypothesis. In particular, given a set of hypotheses  $\mathcal{F}$ , the best hypothesis is selected by minimizing the regularized empirical risk as

$$f^* = \operatorname{argmin}_{f \in \mathcal{F}} \mathcal{R}_{(\mathcal{L}, \mathcal{P})}(f) + \lambda \Omega(f), \quad (2.3)$$

where  $\lambda$  is a trade-off factor called the regularization parameter. The regularizer is represented by  $\Omega(f)$  and is generally a non-negative function which penalizes the inherent complexity of the classifier  $f$ . If the hypothesis class  $\mathcal{F}$  is very large and complex, the search for the optimal  $f^*$  in Equation 2.3 can become a very challenging task. In particular, finding the optimal hypothesis becomes intractable due to the exponential size of the label space. For these intractable classes, it becomes very important to choose a fast searching procedure or an efficient inference technique, for the algorithm to remain tractable.

**Classification v.s. Regression:** The learning problem is often called a *regression* problem when the target variable  $\mathbf{y}$  is continuous, i.e.,  $\mathbf{y} \in \mathcal{R}$ . However, it is termed a *classification* problem when  $\mathbf{y}$  is one of a discrete number of possible values often called class labels, generally represented by  $c$ , i.e.,  $\mathbf{y} \in \{0, 1, \dots, c\}$ . Examples of techniques that have been proposed in the past for supervised learning include *linear/non-linear regression, decision trees, support vector machines, naive bayes* and *neural networks*. Binary and multi-class classification have been the focus of general supervised learning methods.

**Curse of dimensionality:** In most computer vision problems, images are the most predominant form of data used for supervised learning algorithms. These images are often preprocessed to compute properties, often called features, representing the images in a compact form. The size of the parameter search space increases expo-

nentially with the high dimensionality of these input features, and thus increases the chance that the classification algorithm will find spurious classifiers that are generally invalid. Indeed, the required number of labeled samples for supervised classification increases as a function of dimensionality [Jimenez and Landgrebe, 1998]. This phenomenon is usually called the *curse of dimensionality* [Bellman, 1957]. An interesting example of this phenomenon are decision tree classifiers. Although the efficiency of binary decision trees is very high in low dimensions, they fail to maintain this performance when the number of dimensions increases beyond a certain limit. Interestingly, classifiers with fewer features are much more efficient to learn and their is less chance of over-fitting to the data. In this line of research, a number of dimensionality reduction techniques have been proposed, e.g., Linear discriminant analysis [Riffenburgh and Clunies-Ross, 1960], Principal component analysis [Dunteman, 1989], Independent component analysis [Hyvärinen et al., 2004], and Robust PCA [Candès et al., 2011]. Indeed, learning compact representations in itself is a very challenging problem.

**Computational complexity:** An important criterion for comparing the performance of different supervised learning techniques is to take into account their computational complexities. Formally, this is defined as the amount of CPU cycles or GPU time used at time of training and testing. Typically, the inference complexity at test time is important since it measures exactly how well a technique will scale with the amount of test data especially when massive datasets are used. Testing complexity can be critical especially for certain real time applications. Since most of the algorithms have worse than linear computational complexity, dealing with large scale datasets requires efficient inference algorithms.

In this section, we have defined a general supervised learning problem and the important terms related to learning a better classifier. However, with increasing complexity of data with multiple, complex and correlated labels structured prediction and Bayesian analysis methods often provide better generalization guarantees. In the next sections, we provide an overview of structured prediction learning and a general formulation to model these problems using probabilistic graphical models.

## 2.2 Structured Prediction

Real world data exhibits complex relations and strong correlations between different parts of an input. Structured prediction deals with the problem of classifying structured outputs and/or multiple, interdependent outputs. Essentially, it deals with data with inherent structure and often described as a generalization of the proba-

bilistic inference task. This is in contrast with supervised learning, as discussed in the previous chapter, where inference is performed to learn the functional dependencies between inputs and outputs from training data.

Formally, for an input domain  $\mathcal{X}$  and a structured output domain  $\mathcal{Y}$ , structured prediction is defined as predicting  $f(\mathcal{X})$  by maximizing an underlying function  $g$ , *i.e.*,

$$y^* = f(\mathcal{X}) = \operatorname{argmax}_{y \in \mathcal{Y}} g(\mathcal{X}, y). \quad (2.4)$$

Finding the optimal  $y^*$  in the above equation is generally termed as inference or prediction. However, the function  $g(\mathcal{X}, y)$  is often formulated using a linear parametrization of model parameters  $\theta$  and with a learnable feature function  $\Phi$ , which yields

$$y^* = f(\mathcal{X}) = \operatorname{argmax}_{y \in \mathcal{Y}} \theta^T \Phi(\mathcal{X}, y). \quad (2.5)$$

Interestingly, the output space has inherent structure and label dependencies that encode interesting concepts rather than a single discrete value. This is in contrast with the previous section, in which data consists of several parts, here, not only the parts themselves contain information, but also the configuration in which the parts belong together, *e.g.*, on a spatial grid. An example of this problem is pixel-wise *image segmentation*, where an individual pixel often doesn't contain enough information on its own but can benefit from dependencies with its neighbouring pixels.

It is important to note that the structured prediction problems are often formulated using graphical models. Also, the model parameters  $\theta$  and feature function  $\Phi$  are generally learned using efficient inference techniques in probabilistic graphical models. Here, we first briefly describe essential concepts related to statistical inference. It rests on the assumption that prior knowledge can be incorporated in the model parameters. The key ingredients to statistical inference are the *prior distribution* and the *likelihood function*.

**Prior distribution:** A prior distribution quantifies the uncertainty about the parameters before any data is observed. The model parameters  $\theta$  is a random vector and is assumed to have a probability distribution. The prior distribution is generally represented as  $p(\theta \mid \alpha)$  where  $\alpha$  is a vector of hyper-parameters.

**Likelihood function:** A likelihood function  $p(\mathbf{X} \mid \theta)$  (often simply the likelihood) is a function of the parameters of a statistical model and of the data. Likelihood functions play a key role in statistical inference.

**Conditional probability:** The conditional probability is formally defined as

**Definition 2.2.1.** Let  $\mathcal{X}$  and  $\mathcal{Y}$  be any two events in a sample space  $S$  with  $P(\mathcal{Y}) > 0$ . The conditional probability of  $\mathcal{Y}$  given  $\mathcal{X}$  is written  $p(\mathcal{Y} | \mathcal{X})$  and is defined by

$$p(\mathcal{Y} | \mathcal{X}) = \frac{p(\mathcal{Y} \cap \mathcal{X})}{p(\mathcal{X})}. \quad (2.6)$$

**Bayes' theorem:** Bayes' theorem, also known as Bayes' rule, relates conditional probabilities of multiple events and provides a rule to update or revise the strengths of evidence-based beliefs when a new evidence becomes available. It gives a formula to find the conditional probability of an event  $p(\mathcal{Y} | \mathcal{X})$ , say, when the "reverse" conditional probability  $p(\mathcal{X} | \mathcal{Y})$  is the probability that is known. Formally,

**Definition 2.2.2.** Let the events  $y_1, y_2, \dots, y_k$  be disjoint and exhaustive events in the sample space  $S$ , such that  $p(y_i) > 0$ , and let  $\mathcal{X}$  be an event such that  $p(\mathcal{X}) > 0$ . Then according to Equation 2.6,

$$p(y_i | \mathcal{X}) = \frac{p(\mathcal{X} | y_i)p(y_i)}{p(\mathcal{X})}. \quad (2.7)$$

In the remainder of this section, we present a principled way to encode the relationships between the structured outputs using graphical models such as *Markov random fields* and *Conditional random fields*.

### 2.2.1 Probabilistic Graphical Models

Graphs are an intuitive way of representing the relationships among a large set of random variables. A graphical model is a representation of the probability distribution in a way that the distribution can encode the conditional independence structure of a large number of variables with graphs [Koller and Friedman, 2009]. It is an intuitive and computationally useful way to reason about probability by abstracting out the parametric details from the conditional dependence relationships between the variables. In a graphical model, given some observations the goal is to estimate the best solution among all feasible solutions by encoding the joint or conditional probability distribution. Furthermore, prior probability distributions let us incorporate knowledge and some additional assumptions [Nowozin and Lampert, 2011].

There are two kinds of graphical models; those based on directed graphs and those based on undirected graphs. The models represented by directed graphs are typically termed as *Bayesian networks* and the undirected graph models are known as *Markov random fields*. The following sections provides a brief overview of these two types of graphical models.



### 2.2.1.1 Bayesian Networks

A Bayesian network is a graphical model that represents random variables along with their conditional dependencies with a directed acyclic graph (DAG) [Koller and Friedman, 2009]. The distribution is represented by variables that are the nodes in the graph and the directed edges between the nodes describe the conditional dependencies. The model represents a factorization of the joint probability of all random variables. More precisely, given a set of variables  $\mathbf{Y} = \{y_1, \dots, y_n\}$ , the joint probability distribution for  $\mathbf{Y}$  factorizes into the following form

$$p(\mathbf{y}) = \prod_{i=1}^n p(y_i | \hat{y}_i), \quad (2.8)$$

such that

$$\sum_{\mathbf{y}} p(\mathbf{y}) = 1. \quad (2.9)$$

We use  $y_i$  to denote both the variable and its corresponding node, and  $\hat{y}_i$  to denote the parents of node  $y_i$ . Together, these components define the joint probability distribution for  $\mathbf{Y}$ . Equation 2.8 shows the conditional probability factorization of the joint probability distribution, which can be further factorized into prior probabilities and conditional probabilities.

Figure 2.1 shows an example of a simple Bayesian network with 5 random variables. The joint probability of the variables  $\mathbf{X} = \{A, B, C, D, E\}$  is given by

$$p(A, B, C, D, E) = p(A)p(B)p(C|A, B)p(D|B, C)p(E|C, D). \quad (2.10)$$

It is interesting to note that the Bayesian network structure allows us to ignore

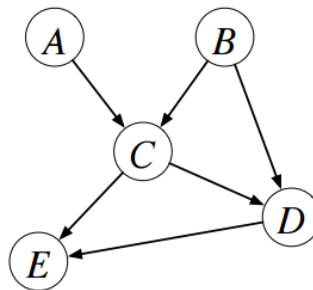


Figure 2.1: **Example of a Bayesian network** [Friedman et al., 1997] represented by a directed acyclic graph. The nodes correspond to random variables and the edges represent statistical dependencies between the variables.

certain dependencies if we already know the value of certain nodes in the network. The structure of the factorization that the network implies will restrict the different possibilities of what distributions we can have that fit the model [Koller and Friedman, 2009]. However, the directed edges in a Bayesian network are used to capture only the causal relationships between random variables. It cannot be used to model the correlation relationships between random variables. Therefore undirected graphical models (Markov Random Fields) are important.

### 2.2.1.2 Markov Network

A Markov network, also known as Markov random field (MRF), is an undirected graphical model of a joint probability distribution [Koller and Friedman, 2009]. Interestingly, the goal is similarly to the Bayesian networks, to simplify the joint probability distribution, as well as preserve interesting dependencies. However, each node  $\mathbf{y}$  is connected to node  $\hat{\mathbf{y}}$  if they are somehow related regardless of other variables. This relation is encoded using multiple *factors*.

**Factor:** A factor  $\phi$  is a function from the value of a random variable to a real value. Formally, let  $\phi_f(\mathbf{y}_f)$ ,  $f = 1 \cdots k$  be a set of factors  $\mathcal{F}$  defined on random variables  $\mathbf{y}_1, \cdots, \mathbf{y}_n$ . The only constraint on factors is that the entries in the factors are non-negative. These factors are usually learned by maximum likelihood estimation.

**Joint probability:** The probability is closely related to the product of all factors and the only difference lies in that the products do not sum to 1 over  $\mathbf{y}$ 's. This unnormalized measure is defined as

$$\tilde{p}(\mathbf{y}) = \prod_{f \in \mathcal{F}} \phi_f(\mathbf{y}_f). \quad (2.11)$$

A partition function  $Z$  is typically measured as the finite sum of this unnormalized measure. Formally,

$$Z = \sum_{\mathbf{y}} \tilde{p}(\mathbf{y}) = \sum_{\mathbf{y}} \prod_{f \in \mathcal{F}} \phi_f(\mathbf{y}_f). \quad (2.12)$$

The joint probability is then defined as

$$p(\mathbf{y}) = \frac{1}{Z} \tilde{p}(\mathbf{y}) = \frac{\prod_{f \in \mathcal{F}} \phi_f(\mathbf{y}_f)}{\sum_{\mathbf{y}} \prod_{f \in \mathcal{F}} \phi_f(\mathbf{y}_f)}. \quad (2.13)$$

Note that, given a set of factors, the Markov network is unique [Koller and Friedman, 2009], and factorization cannot be read by simply looking at the graph.

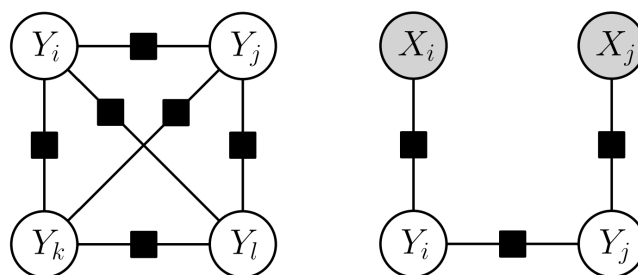


Figure 2.2: Example of a possible factor graph in probabilistic graphical models [Nowozin and Lampert, 2011]. Here the nodes and edges represent the unary potentials and pairwise interactions respectively. Left: Markov random field; Right: Conditional random field.

An example of a four node MRF with pairwise potentials is shown in Figure 2.2. As discussed in Section 2.2.1.1, even though a Bayesian network is complete for modeling random variables, it is not successful in all applications. For example, two neighboring pixels in an image are certainly correlated, but which pixel is the parent of the other pixel is clearly not defined. When modeling with a Bayesian network, the parent nodes of a pixel will grow exponentially, and thus little is gained from such decomposition in terms of computational cost.

### 2.2.1.3 Conditional Random Fields

Conditional Random Fields (CRFs) are essentially the same as Markov random fields except that instead of modeling the joint probability, as discussed in section 2.2.1.2, they model the conditional probability [Koller and Friedman, 2009]. Indeed, the conditional independence assumptions are encoded in the graph, as shown in Figure 2.2. In other words, with  $X$  and  $Y$  be two random variables and we are interested in  $p(Y | X)$  rather than  $p(X, Y)$ .

For example, in an image classification problem, the task is to tag a label  $y$  to a given image  $X$ . Indeed, this is called a discriminative model if we model  $p(X | Y)$ , in contrast to generative modeling  $p(Y, X)$ .

**CRF conditional probability:** We use the conditional probability formulation in Equation 2.6 to formulate the conditional probability of two random variables  $Y$

given  $\mathbf{X}$  in an MRF setting. This yields

$$\begin{aligned} p(\mathbf{Y} | \mathbf{X}) &= \frac{p(\mathbf{Y}, \mathbf{X})}{p(\mathbf{X})} \\ &= \frac{\tilde{p}(\mathbf{Y}, \mathbf{X})}{\tilde{p}(\mathbf{X})} \\ &= \frac{\tilde{p}(\mathbf{Y}, \mathbf{X})}{\sum_{\mathbf{y}} \tilde{p}(\mathbf{X}, \mathbf{y})}, \end{aligned} \quad (2.14)$$

where the joint probability in Equation 2.14 is defined according to the unnormalized MRF measure in Equation 2.11

$$\tilde{p}(\mathbf{X}, \mathbf{Y}) = \prod_{f \in F} \phi_f(\mathbf{y}_f; \mathbf{x}_f). \quad (2.15)$$

A partition function  $Z(\mathbf{X})$  is typically measured as the finite sum of this unnormalized measure. Formally,

$$Z(\mathbf{X}) = \sum_{\mathbf{y}} \tilde{p}(\mathbf{X}, \mathbf{y}) = \sum_{\mathbf{y}} \prod_{f \in F} \phi_f(\mathbf{y}_f; \mathbf{x}_f). \quad (2.16)$$

The CRF conditional probability is then defined as,

$$p(\mathbf{Y} | \mathbf{X}) = \frac{1}{Z(\mathbf{X})} \tilde{p}(\mathbf{X}, \mathbf{y}) = \frac{\prod_{f \in F} \phi_f(\mathbf{y}_f; \mathbf{x}_f)}{\sum_{\mathbf{y}} \prod_{f \in F} \phi_f(\mathbf{y}_f; \mathbf{x}_f)}. \quad (2.17)$$

**Energy minimization:** In graph-based learning tasks, it is common to find the solution of the problem by minimizing an energy function. An energy function is defined for all possible solutions and measures the quality of the solutions. This is usually interpreted as solving for the state of maximum probability [Nowozin and Lampert, 2011].

Formally, the energy is defined for each factor  $\phi_f(\mathbf{y}_f)$  in the graph, and we use  $E_f(\phi_f)$  to describe the energy of a factor. The overall energy is defined as the product of these factors. Both factors and the energy function can be re-written as

$$\begin{aligned} \phi_f(\mathbf{y}_f) &= \exp(-E_f(\phi_f)) \\ E_f(\phi_f) &= -\log(\phi_f(\mathbf{y}_f)). \end{aligned} \quad (2.18)$$

We can now rewrite Equation 2.17 as

$$\begin{aligned}
 p(\mathbf{Y} = y) &= \frac{1}{Z} \prod_{f \in \mathcal{F}} \phi_f(\mathbf{y}_f) \\
 &= \frac{1}{Z} \prod_{f \in \mathcal{F}} \exp(-E_f(\mathbf{y}_f)) \\
 &= \frac{1}{Z} \exp\left(-\sum_{f \in \mathcal{F}} E_f(\mathbf{y}_f)\right).
 \end{aligned} \tag{2.19}$$

In general, finding the state  $y \in \mathbf{Y}$  which has the highest probability is formulated as an energy minimization problem of the form

$$y^* = \operatorname{argmax}_{y \in \mathbf{Y}} p(\mathbf{Y} = y) = \operatorname{argmax}_{y \in \mathbf{Y}} \frac{1}{Z} \exp\left(-\sum_{f \in \mathcal{F}} E_f(\mathbf{y}_f)\right). \tag{2.20}$$

**Maximum A Posteriori (MAP) Inference:** The optimal labeling is often obtained using maximum-a-posteriori inference for the energy minimization problem. Formally, given the factors  $\mathcal{F}$  of a graph and an observation  $\mathbf{x}$ , the state  $y^* \in \mathbf{Y}$  of maximum probability is

$$y^* = \operatorname{argmax}_{y \in \mathbf{Y}} p(\mathbf{Y} = y \mid \mathbf{x}) = \operatorname{argmax}_{y \in \mathbf{Y}} \frac{1}{Z} \exp\left(-\sum_{f \in \mathcal{F}} E_f(\mathbf{y}_f; \mathbf{x}_f)\right). \tag{2.21}$$

**Marginal Inference:** The optimal probability is sometimes obtained using the probabilistic inference for the energy minimization problem. Formally, given the factors  $\mathcal{F}$  of a graph and an observation  $\mathbf{x}$ , the log partition function value and the marginal distributions for each factor is

$$\begin{aligned}
 \log Z(\mathbf{x}) &= \log \sum_{y \in \mathbf{Y}} \exp(-E(y; \mathbf{x})) \\
 \pi_f(\mathbf{y}_f) &= p(\mathbf{Y}_f = \mathbf{y}_f \mid \mathbf{x}),
 \end{aligned} \tag{2.22}$$

whereas the result  $\pi$  of the probabilistic inference describes the marginal distributions. MRFs and CRFs provide a principled way of integrating local evidence and modeling the local neighborhood dependencies, which are strong in most graph based tasks. However, the maximum-a-posteriori (MAP) and marginal inference problems are known to be NP-hard for general graphs and factors [Nowozin and Lampert, 2011], and therefore are usually restricted to a certain form so that the underlying problem remains tractable.

### 2.2.2 Variational Inference

As discussed earlier in this section, performing exact probabilistic inference on general factor graphs is hard [Koller and Friedman, 2009] especially for graphs with a large number of nodes and edges. However, an approximation of this exact inference can be estimated efficiently by using fast approximate inference methods. In this line of research, *variational inference* techniques are useful because they turn inference into optimization problem. Optimization goal is to fit variational parameters such that learned distribution is close to exact posterior in KL-divergence. Indeed, variational inference approximates complex quantities for difficult models and are scalable for massive parameters.

**Mean-field inference:** The *Mean field* methods perform approximate probabilistic inference by restricting the large solution search space within a tractable subset of distributions. Mean field inference estimates an approximate distribution which best approximates the original distribution [Jordan et al., 1999; Wainwright and Jordan, 2008].

The idea of mean field inference is to formulate an optimization problem using a family  $\mathcal{Q}$  of tractable probability distributions,  $q \in \mathcal{Q}$  on  $\mathbf{Y}$ , so as to best approximate the required distribution. This is expressed as

$$q^* = \operatorname{argmin}_{q \in \mathcal{Q}} D_{KL}(q(\mathbf{y}) \| p(\mathbf{y} | \mathbf{x})), \quad (2.23)$$

where  $D_{KL}$  represents the *Kullback-Leibler* divergence between the two probability distributions, and the marginal probability  $q(\mathbf{y})$  provides an approximation of  $p(\mathbf{y} | \mathbf{x})$ . This *Kullback-Leibler* divergence can be expanded as

$$D_{KL}(q(\mathbf{y}) \| p(\mathbf{y} | \mathbf{x})) = \sum_{y \in \mathbf{Y}} q(\mathbf{y}) \log q(\mathbf{y}) - \sum_{y \in \mathbf{Y}} q(\mathbf{y}) \log p(\mathbf{y} | \mathbf{x}). \quad (2.24)$$

In general, the set  $\mathcal{Q}$  is the set of all fully factorized distributions that can be expressed as a product of independent marginals as

$$q(\mathbf{y}) = \prod_{i=1}^n q_i(y_i). \quad (2.25)$$

In such a set, the entropy decomposes as a sum over entropies of each node in the graph. Hence, using Equations 2.25 and 2.24, the *Kullback-Leibler* divergence in Equation 2.23 yields the following variational inference problem:

$$q^* = \operatorname{argmax}_{q \in \mathcal{Q}} \left[ - \sum_i \sum_{y \in \mathcal{Y}} q(y_i) \log q(y_i) - \sum_{f \in \mathcal{F}} \sum_{y \in \mathcal{Y}} \left( \prod_i q_i(y_i) \right) E_f(\mathbf{y}_f; \mathbf{x}_f) \right]. \quad (2.26)$$

Indeed, the maximization problem in Equation 2.26 can be analytically solved to obtain the optimal solution  $q^*$  [Nowozin and Lampert, 2011]. The resulting traditional inference algorithm is quadratic in number of variables and is impractical for large graphs with billions of edges. Therefore, a fast approximate mean-field inference method is used for densely connected graphs.

**Fast mean-field inference in fully-connected CRF:** Mean-field inference in densely connected conditional random fields (CRFs) can be efficiently approximated using high-dimensional Gaussian filtering. Note that Gaussian filtering can be performed efficiently using *permutohedral lattice*, and is both linear in input size and polynomial in feature dimensionality [Adams et al., 2010]. The idea is to encode pairwise edge potentials using a linear combination of Gaussian kernels [Krähenbühl and Koltun, 2011]. The benefit of using a Gaussian filter is that inference reduces to bilateral filtering problem. Furthermore, it allows to learn high dimensional linear filters that operate in sparsely populated feature spaces.

Let us first recall the formulation of CRF as described in section 2.2.1.3. The energy function  $E(\cdot)$  of densely-connected CRF model is defined as

$$E(\mathcal{Y}) = \sum_{i=1}^N \psi_u(y_i) + \sum_{i=1}^N \sum_{j>i} \psi_p(y_i, y_j). \quad (2.27)$$

where  $\psi_u$  and  $\psi_p$  represents unary and pairwise potentials respectively. The pairwise potential  $\psi_p(y_i, y_j)$  is defined as a linear combination of Gaussian kernels  $k^{(t)}(\mathbf{f}_i, \mathbf{f}_j)$ :

$$\psi_p(y_i, y_j) = \mu(y_i, y_j) \sum_{t=1}^T w^{(t)} k^{(t)}(\mathbf{f}_i, \mathbf{f}_j). \quad (2.28)$$

where  $\mu(\cdot)$  is a compatibility function, and  $T$  represents total number of Gaussian kernels. The CRF distribution is defined by

$$P(\mathcal{Y}) = \frac{1}{Z} \exp \left( \sum_{i=1}^N \psi_u(y_i) + \sum_{i=1}^N \sum_{j>i} \psi_p(y_i, y_j) \right), \quad (2.29)$$

where  $Z$  defines the partition function. Minimizing the *Kullback-Leibler* divergence in Equation 2.23 and substituting pairwise potential in Equation 2.28, yields following iterative update equation:

$$Q_i(y_i = l) = \frac{1}{Z_i} \exp \left\{ \psi_u(y_i) + \sum_{l' \in \mathcal{L}} \mu(l, l') \sum_{t=1}^T w^{(t)} \sum_{j \neq i} k^{(t)}(\mathbf{f}_i, \mathbf{f}_j) Q_j(l') \right\}. \quad (2.30)$$

In practice, Equation 2.30 is optimized iteratively by using a series of message passing steps as outlined in the following algorithm.

---

**Algorithm 2.1** Mean field in fully connected CRFs [Krähenbühl and Koltun, 2011]

---

```

1: procedure FastMeanFieldInference
2:   Initialize  $Q$  ▷  $Q_i(y_i) \leftarrow \frac{1}{Z_i} \exp\{-\psi_u(y_i)\}$ 
3:   while not converged do
4:      $\bar{Q}_i^{(t)}(l) \leftarrow \sum_{j \neq i} k^{(t)}(\mathbf{f}_i, \mathbf{f}_j) Q_j(l)$  for all  $l$  ▷ Message passing
5:      $\hat{Q}_i(y_i) \leftarrow \sum_{l' \in \mathcal{L}} \mu^{(t)}(l, y_i) \sum_{i=1}^T w^{(t)} \bar{Q}_i^{(t)}(l)$  ▷ Compatibility transform
6:      $Q_i(y_i) \leftarrow \exp\{\psi_u(y_i) - \hat{Q}_i(y_i)\}$  ▷ Local update
7:     normalize  $Q_i(y_i)$ 
8:   end while
9: end procedure

```

---

Each message passing step updates a single variable by aggregating information from all other variables. Note that message passing step can be expressed as convolution with a Gaussian kernel in feature space as

$$\begin{aligned} \bar{Q}_i^{(t)}(l) &= \sum_j k^{(t)}(\mathbf{f}_i, \mathbf{f}_j) Q_j(l) - Q_i(l) \\ &= [G_{\Lambda^{(t)}} \otimes Q(l)](\mathbf{f}_i) - Q_i(l). \end{aligned} \quad (2.31)$$

Employing approximate high-dimensional filtering [Adams et al., 2010], the computational complexity of message passing step is reduced from quadratic to linear in number of variables. Thus, overall approximate mean-field inference algorithm becomes sublinear in number of edges in the model.

## 2.3 Feature Engineering

Traditional learning methods, including the ones described in section 2.1 and 2.2, rely on an intermediate representation suitable for the learning subsystem. In this section, we consider learning data representations, also called *features*, which is an essential aspect of learning models from data. Fundamentally, any characteristic piece of information is attributed as a feature as long as it is useful for the prediction model. All the learning methods described earlier in this chapter heavily depend on



the choice of data features used to train the models. The goal of feature learning is to seek optimal transformation of potentially unstructured raw input data, which is generally in the form of a text string, image, or audio signal, to increase the accuracy of the learned models. Coming up with this transformation even when using domain knowledge is both expensive and difficult. However, this need can be obviated by automatically learning the representation from the data.

Formally, let  $\mathcal{D}$  denote the dataset which consists of  $n$  data and label pairs  $\{(\mathcal{X}, \mathcal{Y})\} = \{(\mathbf{X}_1, y_1), (\mathbf{X}_2, y_2), \dots, (\mathbf{X}_n, y_n)\}$ , where each  $\mathbf{X}_i \in \mathcal{R}^{uxv}$  is a matrix of  $uxv$  dimensions representing an input image, and  $y_i$  is the given label for that data. Feature learning is defined as follows:

**Definition 2.3.1.** Let  $\mathcal{H}$  be the feature space and  $\mathcal{G}$  be a set of functions that map an input image  $\mathbf{X}$  to the feature space  $\mathcal{H}$ , i.e.,  $\mathcal{G} : \mathbf{X} \rightarrow \mathcal{H}$ . Given a function  $g \in \mathcal{G}$  and for any  $i \in [1 \dots n]$ , the feature learning is defined for input data point  $\mathbf{X}_i \in \mathbf{X}$  as learning a function  $g(\mathbf{X}_i) \in \mathcal{H}$ . Thus producing an extracted feature vector as  $g(\mathcal{D}) = \{(g(\mathbf{X}_1), y_1), (g(\mathbf{X}_2), y_2), \dots, (g(\mathbf{X}_n), y_n)\}$ .

The function  $g(\cdot)$  can be either hand-designed or automatically learned from the data. In large scale visual recognition tasks, this function is generally learned from the training data.

**Representation learning techniques:** In this line of research, many methods have been proposed in the past including techniques for hand-crafted feature selection [Wang and He, 1990; Lowe, 1999; Ojala et al., 2002; Dalal and Triggs, 2005], distance metric learning [Weinberger et al., 2006; Weinberger and Saul, 2008; Weinberger and Chapelle, 2009; Weinberger and Saul, 2009; Parameswaran and Weinberger, 2010; Kedem et al., 2012], dimensionality reduction [Riffenburgh and Clunies-Ross, 1960; Dunteman, 1989; Hyvärinen et al., 2004; Candès et al., 2011] and automatic representation learning [LeCun et al., 1989, 1998; Caruana, 1994; Hinton, 2002; Hinton et al., 2006; Bengio, 2009; Goodfellow et al., 2009; Bengio, 2011; Bergstra and Bengio, 2012; Bengio, 2012; Bengio et al., 2013]. Indeed, feature representations learned for one task might not be useful for other tasks. Therefore, in many cases a diverse range of features are used in an ensemble to achieve a certain task-specific accuracy. Furthermore, there is no empirical way of evaluating the selected or learned features directly other than using them in a task-specific classifier.

In general, hand-crafted features are not scalable to the large scale visual challenges described in Section 1.1. Note that in this thesis, we mostly focus on directly

using/learning the representation using deep learning architectures [LeCun et al., 1998]. In the remaining of this section, we first briefly present some hand-engineered features and then we will discuss in detail automatically learning features from data.

### 2.3.1 Hand-Crafted Features

In this section, we describe the two hand-crafted features, *Local binary patterns* [Wang and He, 1990] and *Histogram of oriented gradients* [Dalal and Triggs, 2005], used in our research for similarity learning. These features are selected because of their simpler yet effective representation which can be efficiently computed for large scale datasets.

**Local binary patterns [Wang and He, 1990]:** Local binary patterns (LBP) are a famous visual descriptor for object and face recognition. The main idea of LBP is to summarize the local structure as a histogram in an image. This is achieved by dividing an image into a regular grid, and within each grid cell, all the pixels are compared against their 8 neighbors. Indeed, each grid cell is encoded as an 8-bit binary vector containing a 0 or 1 which is calculated by comparing center pixel with the pixels, clockwise or counter-clockwise, along a circle of some fixed radius. Furthermore, a histogram is calculated on all the grid cells and a normalization is performed to have a consistent image representation. Finally, the normalized histograms of all grid cells is concatenated to form an LBP feature vector for the entire image, as shown in Figure 2.3.

The benefits of using LBP is that it can ignore edges that do not provide enough information. Bit encoding by following a fixed radius circle makes it more invariant to the orientation of the textures, and makes texture identification much more efficient by using small sized feature vectors. The limitation of LBP is that it does not take into consideration of the contrast of the local neighbourhood [Lowe, 1999].

**Histogram of oriented gradients [Dalal and Triggs, 2005]:** In the era before automatically learning visual representations, the histogram of oriented gradients (HoG)

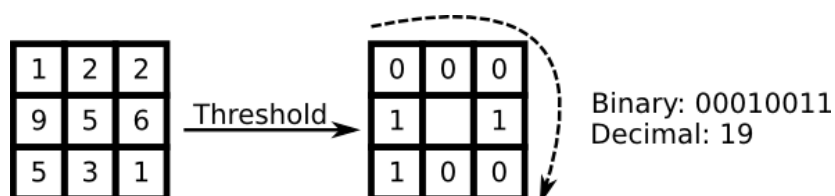


Figure 2.3: This image shows the process of computing the **local binary patterns (LBP) features** as in [Wang and He, 1990].

descriptor was one of the most popular low-level image representation technique for its invariance to geometric and photometric transformations. The main idea of HoG is to first calculate the gradients along the horizontal and vertical directions, and then a histogram of gradients is calculated to form a feature vector. Indeed, this can be easily achieved by filtering an image with  $[-1, 0, 1]$  and  $[-1, 0, 1]^T$  kernels [Dalal and Triggs, 2005].

Let  $g_x$  and  $g_y$  be the gradients in the horizontal and vertical spatial directions, respectively. The magnitude can be calculated as  $g = \sqrt{g_x^2 + g_y^2}$  and the direction as  $\theta = \arctan \frac{g_y}{g_x}$ . The reason for calculating gradients is to highlight outlines, and in doing so, a lot of non-essential information is removed, e.g., smooth areas. Finally, the image is divided into a grid and a histogram is calculated. A normalization is also performed to have a consistent representation across the full image. Figure 2.4 shows an example of HoG features. Image patches of fixed aspect ratio are required to compute histogram of oriented gradients feature vector.

The benefits of using HoG is that it provides better perceptual similarity because edge responses are localized by using intensity gradients instead of pixel intensity. Also, the normalization scheme is locally applied thus it is insensitive to the global contrast. Note that HoG uses the bilinear interpolation between HoG cells, thus can also handle small localization errors. Another important advantage is that the perceptual similarity is approximated by Euclidean, sometimes cosine, distance between two HoG vectors. This means that HoG features can be directly employed in many supervised and unsupervised learning algorithms.

In this section, we provided an overview of hand-crafted features generation techniques. Next we present an overview of techniques that learn features automatically.

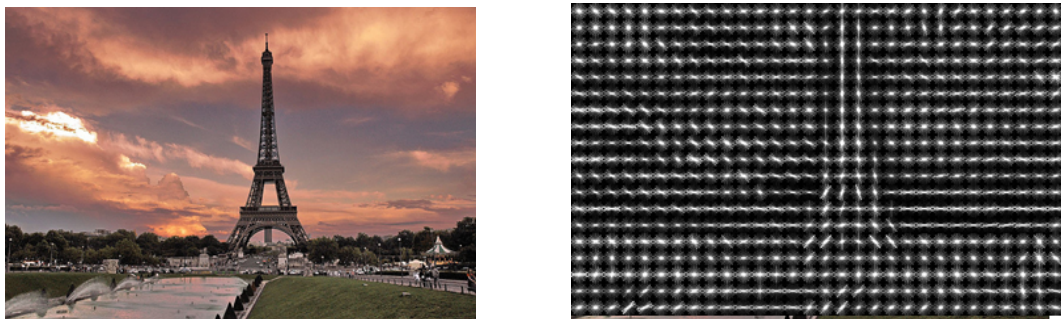


Figure 2.4: **Example of histogram of oriented gradients (HoG) features** [Dalal and Triggs, 2005]. Here the left column shows an original image and the right column shows the HoG features.

### 2.3.2 Deep Learned Features

Learning automatic representations from the raw data allows a machine to automatically discover a representative set of features suitable for a certain task. Breakthroughs in deep networks as well as the availability of large amounts of data and efficient computational resources have had a significant impact on feature extraction [Bengio, 2009]. Recently, deep learning has gained wide popularity due to its empirical successes in a number of traditional computer vision and artificial intelligence tasks. The goal of deep learning algorithms is to yield abstract representations by a composition of multiple simple but non-linear transformations [Bengio, 2011; Bengio et al., 2013]. Indeed, each transformation discovers a representation at a higher, slightly more abstract level, and when combined these less abstract transformations lead to learning of very complex functions [Lecun et al., 2015]. This idea of learning multiple levels of representations also draws inspiration from the theory behind the artificial systems that mimic the human vision system to accurately extract information from natural images, video sequences.

These transformation functions are grouped together in multiple building blocks, often called *layers*. These layers are generally arranged in a network, often called neural network, where each node performs a basic computation. Intuitively, a neural network is decomposed into a complex composite function where each function element corresponds to a differentiable operation. Furthermore, these layers are trained jointly to essentially learn a composition of multiple non-linear transformations on top of the raw input data, thus creating a new feature representation [Bengio et al., 2013]. It is important to note that deep learned features can take advantage of large amounts of data and computation, since very little engineering is required in designing the layer structure. We begin our discussion with a simple neural network.

**Perceptron:** Perceptron is a single layer feed-forward neural network. Let  $(\mathbf{X}, y)$  denote a data and label pair, such that  $\mathbf{X} \in \mathcal{R}^m$  is a vector representing an input image, and  $y$  is the given label for that data. The output of perceptron is defined as

$$h_{(\mathbf{W}, \mathbf{b})}(\mathbf{X}) = f(\mathbf{W}^T \mathbf{X} + \mathbf{b}), \quad (2.32)$$

where  $\mathbf{W}$  and  $\mathbf{b}$  are the transformation parameters. Intuitively, a simple neuron performs a dot product with the input  $\mathbf{X}$  and its weights  $\mathbf{W}$ , adds the bias  $\mathbf{b}$  and applies a non-linearity (or *activation function*)  $f$ . Indeed to learn the transformation parameters, it is important that the activation function  $f$  is fully differentiable w.r.t. the transformation parameters  $\mathbf{W}$  and  $\mathbf{b}$ . An approximation is generally used when the function is not fully differentiable e.g.  $\max(\cdot)$  or  $\min(\cdot)$ . The ultimate goal of the

learning is to minimize an objective function  $J(\theta)$  where  $\theta = \{\mathbf{W}_1, \mathbf{b}_1, \dots, \mathbf{W}_l, \mathbf{b}_l\}$ , and to accomplish this task a set of *loss functions* plays a vital role.

In the remainder of this section, we briefly describe some of the most important layers which are trained using the Mini-batch gradient descent approach. All the layers can be categorized into four broad categories; 1) data processing layers, 2) activation layers, 3) utility layers and 4) loss layers. Formally, each layer in a neural network is parameterized by a set of weights and biases [Jia et al., 2014].

### 2.3.2.1 Data Processing Layers

The most important data processing layers include the *convolution*, *fully connected* and *pooling* layers.

**Convolutional Layer:** The Convolutional layer is the most important layer in image processing tasks since it convolves the input image with a set of learnable filters and transforms the raw input data into low level filter responses. The goal of the convolutional layer is to learn features that are consistent for image regions which look similar.

Formally, let  $\mathbf{W}$  be a set of learnable filters of dimension  $k \times k$  using the input  $\mathbf{X}$ , and  $\mathbf{b}$  the bias term. Output image  $\mathbf{C}$  after the convolution layer is computed as

$$\begin{aligned} \mathbf{C} &= \mathbf{X} * \mathbf{W} + \mathbf{b} \\ c_{(i,j)} &= \sum_{u=0}^{k-1} \sum_{v=0}^{k-1} x_{(i-u,j-v)} w_{(u,v)} + b_{(i,j)}. \end{aligned} \quad (2.33)$$

Here, we  $w \in \mathbf{W}$  defines a single learnable filter, however, each convolutional layer learns a set of kernels  $\mathbf{W}$ . Learning the filters requires the gradient to be computed w.r.t. the kernels  $\mathbf{W}$  and  $\mathbf{b}$  by applying the chain rule of gradients,

$$\begin{aligned} \frac{\partial c_{i,j}}{\partial w_{u,v}} &= \frac{\partial}{\partial w_{u,v}} \left( \sum_s \sum_t w_{(s,t)} x_{(i-s,j-t)} + b_{(i,j)} \right) \\ &= x_{(i-u,j-v)}. \end{aligned} \quad (2.34)$$

where, by expanding the summation and taking the partial derivatives for all the components results in zero values for all except the components where  $s = u$  and  $t = v$ . Similarly, the gradient w.r.t. the bias term can also be calculated.

**Pooling Layer:** The pooling layer is used to reduce the spatial size of the feature maps or the learned representation. Indeed, it reduces the amount of parameters and makes the computation fast in the network. Furthermore, it also controls the overfitting of parameters to the data. The pooling layers do not have any parameters, hence no learning is performed at this step.

The most common pooling methods include *max-pooling* and *average pooling*. During the forward pass through the deep network, an input of size  $N \times M$  is given to the pooling layer. The pooling layer takes a fixed size pooling window, *e.g.*,  $2 \times 2$ , and selects a single value according to the specified operation, *i.e.*, max or avg. The index of this value is kept, because it is important when computing gradient w.r.t. the error of this layer. In max-pooling, the error is propagated to just the index at which the maximum value was observed, however, the error is divided among the pooling window in case of average pooling.

**Fully-connected Layer:** The fully-connected layer is also known as the inner product layer. The idea of fully connected layer is to learn a complex function that transforms the high level feature maps, usually the output of the convolutional layers, into simple concepts. This is accomplished by flattening the feature maps and learning a non-linear combinations of these features, as

$$f_{out} = \mathbf{W}^T \mathbf{X} + \mathbf{b}. \quad (2.35)$$

The benefit of using fully-connected layers is that they can learn transformation functions from high-dimensional to low-dimensional feature space.

### 2.3.2.2 Activation Layers

The most common used activation functions are *sigmoid*, *softmax* and *rectified linear units (ReLU)*.

**Sigmoid layer:** A logistic function, also known as sigmoid, is the most common activation function, and is defined as

$$\begin{aligned} f(t) &= \frac{1}{1 + e^{-t}}, \\ \frac{df(t)}{dt} &= f(t)(1 - f(t)) = f(t)f(-t). \end{aligned} \quad (2.36)$$

The sigmoid function is a simple to use function and can be used as a separate layer inside a network. However, it can only be used for binary classification or regression when used as the last layer.

**Softmax Layer:** The most common function used for learning a multi-class classification probability is the softmax function. It is defined as

$$y_i = \frac{e^{x_i}}{\sum_{j=1}^C e^{x_j}} \quad \text{for } c = 1 \cdots C, \quad (2.37)$$

where  $C$  is the number of classes and the denominator  $\sum_{j=1}^C e^{x_j}$  acts as a regularizer to make it a proper probability distribution  $\sum_{c=1}^C y_c = 1$ . Furthermore, gradient of the softmax function w.r.t. its inputs can be calculated as,

$$\begin{aligned} \text{if } i = j : \frac{\partial y_i}{\partial x_i} &= y_i(1 - y_i) \\ \text{if } i \neq j : \frac{\partial y_i}{\partial x_j} &= -y_i y_j. \end{aligned} \quad (2.38)$$

**ReLU Layer:** The goal of a rectified linear unit (ReLU) layer is to perform a rectifier activation or a thresholding operation. This layer sets any input value which is less than zero to zero. That is,

$$y(x) = \max(0, x). \quad (2.39)$$

The max operation is a non-linear function and the gradient is  $\frac{\partial y}{\partial x} = 0$  for  $x < 0$  and 1 for  $x > 0$ . The gradient is not defined for  $y(0)$  because of non-differentiability at zero. However, the activation function is slightly modified to be  $y(x) = \max(x, ax)$  for  $a \leq 1$ .

The benefits of using the ReLU layer is that it enforces sparsity and reduces the likelihood of vanishing gradient. This is accomplished because the gradient is constant for  $x > 0$ , which is in contrast with the *sigmoid* or *softmax* function. Also, sparsity arises when  $x < 0$ , which makes further processing more efficient than with dense representations.

### 2.3.2.3 Utility Layers

The most widely used layers to improve the generalization performance of a network are the *dropout* and the *batch normalization* layers.

**Dropout layer:** It is often required to train large networks with multi-million parameters. This could be a problem when learning with a small dataset or highly correlated samples. This phenomena is generally known as overfitting, and is a serious problem in such networks. The technique to address this challenge is to incorporate dropout which prevents layers from co-adapting too much. It builds upon the key idea of randomly dropping connections from the neural network during training.

It is observed that problems arising due to overfitting to data can be significantly reduced by using dropout. Also, it is possible to achieve major improvements over other regularization techniques [Srivastava et al., 2014].

**Batch Normalization layer:** Deep networks are usually trained with very diverse datasets. Batch normalization is a powerful technique that guarantees that the output features of the layer will be normalized, *i.e.*, zero mean and unit variance. This process has trainable parameters to adapt to the diverse datasets. A major benefit of using batch-normalization is that it keeps the gradients in a proper range, and thus improves gradient flow. This is required for very deep networks such as residual networks [He et al., 2015]. Furthermore, it helps training using higher learning rates and is not sensitive to initialization.

#### 2.3.2.4 Loss Layers

Loss layers are the most important part of any deep network since they define how the error is computed. A loss layer must be based on a fully differential function. The most used loss layers include *Euclidean*, *log*, *softmax* and *cross entropy* loss layers.

**Euclidean loss layer:** Given a ground-truth target  $t$  and its predicted value  $y$ , the Euclidean loss layer computes the sum of squares of differences of its two inputs,

$$\mathcal{L}(y, t) = \frac{1}{2m} \sum_{i=1}^m \|y_i - t_i\|^2. \quad (2.40)$$

and the gradient w.r.t. the input is computed as

$$\frac{\partial \mathcal{L}}{\partial y_i} = \frac{1}{2} \cdot 2 \cdot (y_i - t_i) = (y_i - t_i) \quad (2.41)$$

$$\delta y = \left[ \frac{\partial \mathcal{L}}{\partial y_i} \right]_i = y - t. \quad (2.42)$$

**Log loss layer:** The log loss measures the deviation of the predicted value  $y$  from the ground-truth targets  $t$  as

$$\mathcal{L}(y, t) = - \sum_{i=1}^m t_i \log y_i. \quad (2.43)$$



and the gradient w.r.t. the input is computed as

$$\frac{\partial \mathcal{L}}{\partial y_i} = -\frac{t_i}{y_i} \quad (2.44)$$

$$\delta y = \left[ \frac{\partial \mathcal{L}}{\partial y_i} \right]_i = -t \odot \frac{1}{y}. \quad (2.45)$$

**Softmax loss layer:** This computes the multinomial logistic loss of the softmax of its inputs. This is identical to a softmax layer followed by a multinomial logistic loss layer, and when combined together provides a numerically stable gradient, as in Equation 2.38.

### 2.3.3 Optimization using Gradient Descent

In neural network, the most popular approach to optimize the overall objective function is *Gradient descent*. The idea is to start from an initial location in the solution space and estimate the best direction to take a small step. Ideally, the direction of steepest descent, also known as the opposite direction of gradient, is mathematically proven to be the best direction that leads to the global minimum of the convex objective function. Therefore, the best model parameters  $\theta^*$  are estimated by taking the gradient of the objective function  $\nabla_{\theta} J(\theta)$  w.r.t. to all the parameters, and by taking a small step in the opposite direction of the gradient. The step size is dependent upon the learning rate that is usually adjusted for the learning problem.

Practically, multiple variants of gradient descent algorithms are used based on the amount of data used to compute the gradient of the objective function. This is important, especially when dealing with large scale datasets, where all data cannot fit into the memory or when more frequent gradient updates are required.

**Batch gradient descent:** The naive gradient descent, usually referred to as batch gradient descent, performs each parameter update after estimating the gradient of the objective function w.r.t. to the parameters  $\theta$  for the entire training dataset

$$\theta = \theta - \eta \cdot \nabla_{\theta} J(\theta), \quad (2.46)$$

where  $\eta$  is the learning rate for the parameter update. Sometimes calculating the gradient can be very expensive or intractable if the dataset is very large or it cannot fit into memory. Furthermore, it will take a long time to converge for very non-convex objective functions when the initial estimate is far from optimal solution. Another important downside of naive gradient descent is that it can easily get trapped in local

optima. Therefore, clever optimization techniques, such as *stochastic gradient descent* with momentum, are required.

**Stochastic gradient descent (SGD):** In contrast to naive gradient descent, in stochastic gradient descent (SGD), a parameter update is performed for each training example  $\mathbf{X}_i$  and label  $\mathbf{y}_i$ , such as

$$\theta = \theta - \eta \cdot \nabla_{\theta} J(\theta; \mathbf{X}_i; \mathbf{y}_i). \quad (2.47)$$

Stochastic gradient descent is very fast and suitable for online learning algorithms because it performs one update at a time. The drawback of these frequent updates is that they generally cause large variations in the objective function. It is possible to reach at global minimum (though not guaranteed), however, frequent parameter updates complicate convergence.

**Mini-batch gradient descent:** Mini-batch gradient descent is a trade-off between batch gradient descent and stochastic gradient descent, and typically is the algorithm of choice for training large networks. The idea is to perform a parameter update for every mini-batch of  $b$  training samples, where  $b$  is neither too large nor too small.

$$\theta = \theta - \eta \cdot \nabla_{\theta} J(\theta; \mathbf{X}_1, \dots, \mathbf{X}_b; \mathbf{y}_1, \dots, \mathbf{y}_b). \quad (2.48)$$

The benefits to computing the gradient on a batch of samples is that it converges smoothly by reducing the variance of the parameter updates. Usually, a batch size of 128 or 256 samples is used depending on the size of the neural network. However, it can vary for different learning problems based on the memory requirements. Vanilla mini-batch gradient descent approaches are further optimized for better convergence using a better choice of *learning rate*, *momentum*. The learning rate  $\eta$  is the weight of the negative gradient. The momentum  $\gamma$  is the weight of the previous update. Equation 2.48 with momentum becomes

$$\begin{aligned} \phi_t &= \gamma \phi_{t-1} + \eta \nabla_{\theta} J(\theta) \\ \theta &= \theta - \phi_t, \end{aligned} \quad (2.49)$$

where  $\phi_t$  represents the previous weight update. The momentum  $\gamma$  is usually set to 0.9. Additionally, most of the deep learning frameworks including Caffe [Jia et al., 2014], Torch [Collobert et al., 2002] and TensorFlow [Abadi et al., 2015] provide a GPU optimized version of batch SGD to be used on a single machine.

## 2.4 Popular Deep Networks

In this section, we highlight few of the most frequently-used deep networks which are just a combination of the layers in a specific order. Following is a summary of networks that we used in our experiments to compute deep features as well as to build new layers to improve the detection, localization and segmentation accuracy.

### 2.4.1 Alex Net

Learning features using deep architectures have been achieved since the emergence of back-propagation for image recognition [LeCun et al., 1989]. However, Alexnet [Krizhevsky et al., 2012] is widely regarded as the most influential deep network architecture that won the ILSVRC 2012 imagenet large-scale visual recognition challenge [Russakovsky et al., 2015]. Alexnet is a simple and very small convolutional neural network, compared to latest architectures, and it only has 5 convolutional layers with max-pooling and dropout layers, and 3 fully-connected layers. This network was carefully designed with large 7x7 filters to handle 15 million annotated images which were labeled for one thousand categories in a multiclass classification setting. The detailed network structure is shown in Figure 2.5.

The main idea of this network is to combat the over-fitting problem, and was handled by implementing the dropout layers. It is important to note that the data augmentation techniques, including random image translations, reflections, and cropping, were also used to generate more data. Even though Alexnet is a relatively small network, it took five to six days of training on two GTX 580 GPUs. However, this training time was greatly reduced to a few hours by subsequent improvements in high-performance computing.

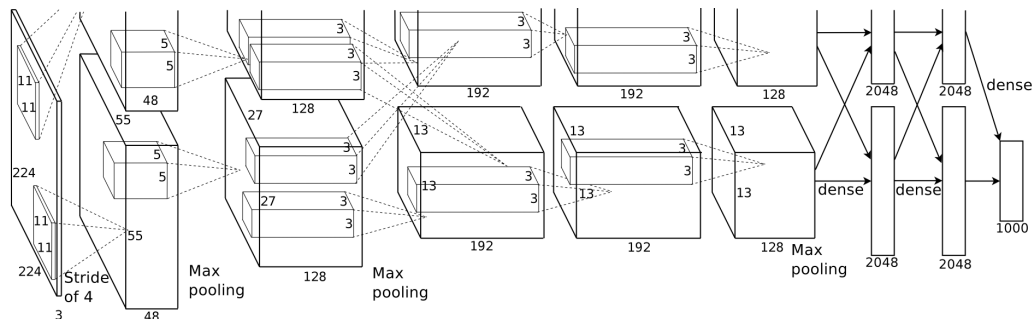


Figure 2.5: An illustration of the Alexnet architecture used in [Krizhevsky et al., 2012].

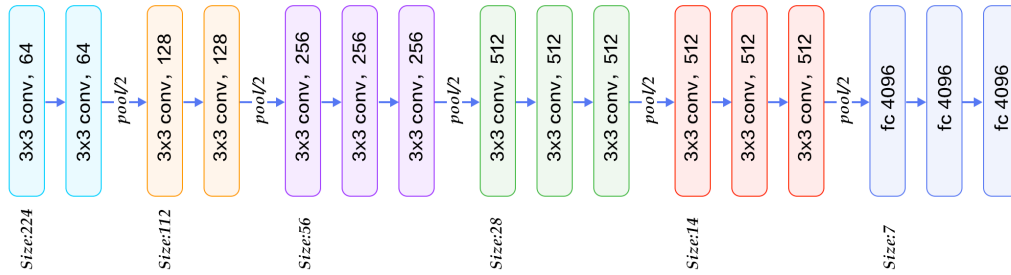


Figure 2.6: An illustration of the visual geometry group network VGG16 architecture as used in [Simonyan and Zisserman, 2015].

### 2.4.2 VGG Network

The VGG [Simonyan and Zisserman, 2015] network architecture is based on the idea of hierarchical representation with a large number of layers but small filter sizes. In contrast to AlexNet [Krizhevsky et al., 2012], the VGG strictly uses small filters of size 3x3 and applies 2x2 max-pooling afterwards. A pad of 1 is required for filters to have the same-sized output feature map. These small-sized filters are used in a hierarchical manner such that a combination of any two consecutive convolutional layers have an effective receptive field of 5x5. Thus, having a deep network will increase the overall receptive field around a single pixel in the image. There are two variants of VGG networks: one is VGG16 which has 16 layers in total and the other is VGG19 which contains 19 layers. An illustration of VGG16 is shown in Figure 2.6.

Interestingly, VGG networks double the learnable convolutional filters after applying max-pooling, which enforces the idea of shrinking spatial dimensions, but growing depth to learn higher level abstract concepts. These networks take two to three weeks on four relatively advanced Nvidia Titan Black graphics processing units to be trained on Image net [Russakovsky et al., 2015].

### 2.4.3 Residual Network

The ultra-deep residual network architecture builds upon two fundamental principles: *residual learning* and *modular architecture*.

The idea of residual learning is that each block of layers in the deep network is only responsible for fine tuning the output from a previous block of layers by just adding a residual to it. This is in contrast to previously discussed traditional approaches like Alexnet [Krizhevsky et al., 2012] and VGG [Simonyan and Zisserman, 2015] that generate completely new output after each layer.

Also, the benefit of having a modular network is that its depth can be increased by just repeating the same network block over and over again. This common set of

layers are called a *residual building block* [He et al., 2015] and is shown in figure 2.7.

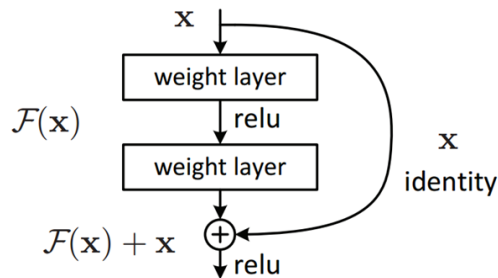


Figure 2.7: **An illustration of the residual building block** as used in residual neural network [He et al., 2015].

The residual neural network has a depth of either 34, 50, 101 or 152 layers [He et al., 2015] and have a minimum of only 3.6% recognition error. Indeed, optimization using the residual mapping is much easier than to optimize for original mapping [He et al., 2015]. Furthermore, the gradients can easily pass through the residual blocks because of the identity mappings, and is less prone to vanishing gradient problem.

## 2.5 Object Detection and Instance Segmentation

In this section, we provide an overview of most commonly used techniques for *object detection* and *instance-level semantic segmentation*.

### 2.5.1 Deformable Part-based Models (DPM)

In the context of object detection, deformable part-based models have become one of the most important supervised learning methods that use hand-engineered features [Felzenszwalb et al.]. In the naive algorithm, the main idea is to reduce the object detection problem to a binary classification problem where a sliding window approach is used on an image to extract patches and subsequently label them into foreground-background classes. Histogram of oriented features [Dalal and Triggs, 2005] are computed for each image patch and then a support vector machine classifier is used to rank them. However, this sliding window approach leads to a highly imbalanced positive and negative training dataset because of very few positive samples. Hard negative mining techniques are often used to carefully select the representative negative samples [Felzenszwalb et al., 2010]. Following is a brief overview of deformable part-based model techniques.

**Pictorial structures:** Main idea of pictorial structures [Fischler and Elschlager, 1973] is to model the local appearance of each object by its parts, and the part deformations

as the spring connections between those parts. Intuitively, this leads to a better performance since it can accurately capture local appearance of the parts and generalizes to previously unseen configurations.

Formally, let  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$  be a graph where the nodes  $\mathcal{V}$  represent the parts and the deformation costs are represented by edges  $\mathcal{E}$ . The part configuration score is defined as

$$\mathbf{S}(v_1, \dots, v_p) = \sum_{i=1}^p m_i(v_i) - \sum_{(i,j) \in \mathcal{E}} d_{ij}(v_i, v_j), \quad (2.50)$$

where  $m(\cdot)$  is the part matching score and the  $d(\cdot, \cdot)$  are the deformation costs. Intuitively, the idea is to learn an unconstrained part model where deformations are learned from the data and the parts can have any configuration. Furthermore, this provides an efficient matching algorithm for deformable part-based models.

**Star-structured:** In a star-structured deformable part based models [Felzenszwalb et al., 2010], the part locations are modeled as a star graph. There is a root node and all the parts are connected to that root node via deformation connections. This leads to a slightly modified objective function, given by

$$\mathbf{S}(\mathbf{x}, v_0) = \max_{v_1, \dots, v_p} \sum_{i=0}^p m_i(\mathbf{x}, v_i) - \sum_{i=1}^p d_i(v_0, v_i), \quad (2.51)$$

where the root node is represented by  $v_0$  and the deformation costs are computed from the root node. Furthermore, it maximizes over all parts by selecting them as a root node one-by-one.

**Mixture models:** The mixture models are data driven models which tried to optimize the structures of objects by learning the aspect ratios, occlusion modes, subclasses relations. In a mixture model, each object model consists of a mixture of two multi-scale deformable part models [Felzenszwalb et al., 2010]. In each deformable part-based model, part locations are trained on feature vectors representing distance of each part from the root node. In particular, classifier that scores an image  $\mathbf{x}_i$  with label  $y_i$  is given by

$$\mathbf{S}_\beta(\mathbf{x}_i) = \max_{z \in Z(\mathbf{x}_i)} \sum_{j=1}^p \beta_j \cdot m_j(\mathbf{x}_i, z), \quad (2.52)$$

where  $\beta$  are model parameters and  $z$  are latent values.

**Learning in Mixture Models:** Here, learning objective is to find optimal model parameters such that

$$y_i \cdot \mathbf{S}_\beta(\mathbf{x}_i) > 0. \quad (2.53)$$

This is learned by minimizing the following latent support vector machine (SVM) objective function

$$\mathcal{L}(\beta) = \frac{1}{2} \|\beta\|^2 + C \sum_{i=1}^n \max(0, 1 - y_i \cdot \mathbf{S}_\beta(\mathbf{x}_i)). \quad (2.54)$$

where  $\max(0, 1 - y_i \cdot \mathbf{S}_\beta(\mathbf{x}_i))$  is convex for negative examples, *i.e.*,  $y_i = -1$ . It is also convex if latent values for positive examples are fixed. It makes optimizing  $\beta$  a convex optimization problem, even though the latent values for the negative examples are not fixed. Furthermore, hard negative mining is used with stochastic gradient descent to train these models. The resulting detections are further re-scored using contextual information for each object class independently.

Although star-structured models and mixture models perform well in detecting objects, they generally predict many detections around one true object. Non-maximum suppression is generally applied as a post processing technique.

**Non-Maximum Suppression:** Non-maximum suppression (NMS) is a post-processing algorithm responsible for reducing the clutter around the true prediction and merging all detections that belong to the same object. Note that if the detection score is at a peak in a certain image region, the probability for that region to be a true positive is high. Similarly, the detector response should be low if the IoU overlap with a true object is small. Based on these assumptions, non-maximum suppression is achieved by fusing the nearby overlapping detections, at the same scale, based on the final detection score.

### 2.5.2 Region-based Convolutional Neural Networks (R-CNN)

Region-based CNN [Girshick et al., 2014] combines category-independent region proposals with rich features computed by a convolutional neural network (CNN) to tackle object detection task. In contrast to methods based on hand-crafted features, this is the first CNN-based object detection method that significantly improves object detection performance on benchmark datasets. Given an image, the goal of R-CNN is to identify the regions that contain distinct objects. In particular, R-CNN performs the following steps:

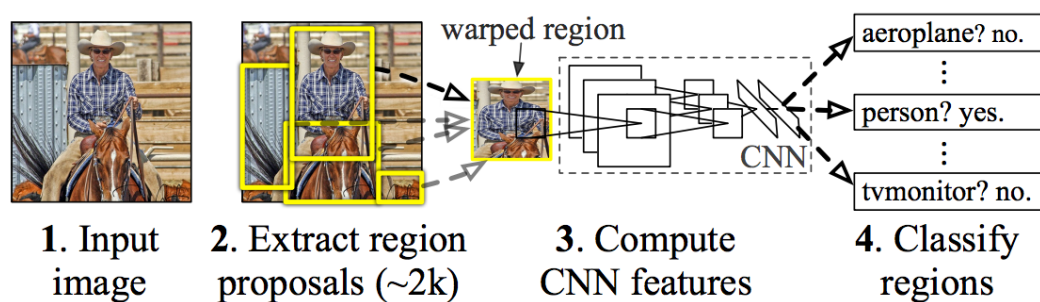


Figure 2.8: **R-CNN**: Object detection using regions-based CNN [Girshick et al., 2014].

- Generate a set of class-independent object proposals (also called bounding boxes) as potential candidate regions with different sizes and aspect-ratios.
- Compute 4096-dimensional feature vector for each bounding-box using  $fc7$  layer of AlexNet [Krizhevsky et al., 2012].
- Use features to train a support vector machine (SVM) classifier, and classify each bounding-box.
- Learn a linear regression model for fine bounding-box localization.
- Apply non-maximum suppression (NMS).

In R-CNN, a set of bounding-boxes are generated using Selective search [Uijlings et al., 2013]. Furthermore, all bounding-boxes are warped to a fixed size ( $224 \times 224$ ) before computing features using the deep network. This is illustrated in the Figure 2.8. In practice, R-CNN is hard to train because of its multi-stage pipeline. Also, all individual bounding-boxes are passed through the network one-by-one and this makes R-CNN extremely slow.

### 2.5.3 Fast Region-based Convolutional Neural Networks (Fast R-CNN)

Fast R-CNN [Girshick, 2015] is an enhanced version of R-CNN [Girshick et al., 2014] that not only improves training and testing speed but also increases the detection accuracy. Indeed, testing a VGG16 network [Simonyan and Zisserman, 2015] using Fast R-CNN is 213x faster as compared to R-CNN, also it gives 9x speedup in the training. This is achieved with a single-stage training using a multi-task loss, thus no disk storage is required for feature caching. Also, shared feature map computation is introduced for overlapping regions. This is illustrated in Figure 2.9.

In Fast R-CNN, similar to R-CNN, a set of bounding-boxes are generated using Selective search [Uijlings et al., 2013]. However, full image is provided to the network



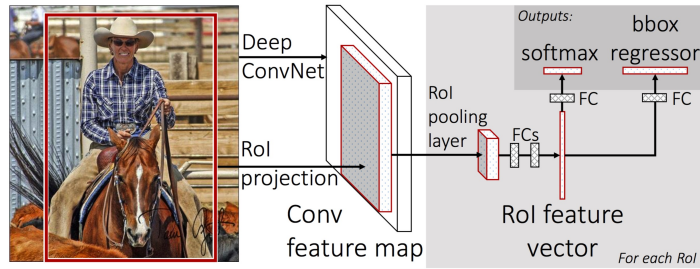


Figure 2.9: **Fast R-CNN**: Efficient object detection using regions-based CNN [Girshick, 2015].

as input and a shared feature map is estimated. Furthermore, a Region of interest (RoI) pooling layer is introduced to extract features for each bounding-box.

**Region of interest pooling:** Each bounding-box is defined by its top-left corner  $(r, c)$  and its height and width  $(h, w)$ . The idea of RoI pooling layer is to jointly obtain CNN features of all bounding-boxes by selecting the corresponding regions from the full image feature map. These features are then pooled using max-pooling. In particular, max-pooling is applied independently to each feature map channel by dividing the RoI into an  $H \times W$  grid (usually  $7 \times 7$ ) of sub-windows. This is in contrast with R-CNN [Girshick et al., 2014] where a separate forward pass is required for each bounding-box. The back-propagation for RoI pooling layer is performed by taking the argmax of gradients inside the bounding-box overlap region.

**Multi-task loss:** Fast R-CNN combines both the classification and localization losses into a single *multi-task loss* in order to speed up training. In particular, a softmax layer is used in the network instead of a separate SVM classifier to predict class labels. Also, bounding box regression is estimated using a fully-connected layer. The multi-task loss is defined as

$$\mathcal{L}(p, u, t^u, v) = \mathcal{L}_{cls}(p, u) + \lambda[\mu \geq 1] \mathcal{L}_{loc}(t^u, v); \quad (2.55)$$

where  $u$  and  $v$  represents ground-truth label and bounding-box regression target, respectively. The  $p$  represents the output probability estimated by a softmax layer, and  $t^u$  is the class-specific bounding-box tuple. This method is further improved in Faster R-CNN [Ren et al., 2015].

### 2.5.4 Faster R-CNN

Generating quality object proposals is a slow process, and it was a major bottleneck for most object detection methods including R-CNN [Girshick et al., 2014] and Fast R-CNN [Girshick, 2015]. Faster R-CNN [Ren et al., 2015] addresses this issue by simultaneously generating candidate regions inside the network. In particular, the idea of Faster R-CNN is to use convolutional feature maps used by region-based detectors for generating nearly cost-free region proposals. This is accomplished by adding a fully-convolutional sub-network, called *Region Proposal Network (RPN)*, on top of the shared features of the CNN, as illustrated in Figure 2.10 (Left).

**Region Proposal Network (RPN):** Given an image, the goal of RPN is to generate object proposals of various sizes and aspect ratios. In particular, a sliding window is passed over the CNN feature map and at each sliding-window location, multiple bounding-boxes and corresponding objectness scores are predicted. Note that for  $k$  object windows, also called anchors,  $4k$  values are predicted for box locations and  $2k$  values are predicted for objectness scores. This is illustrated in Figure 2.10 (Right).

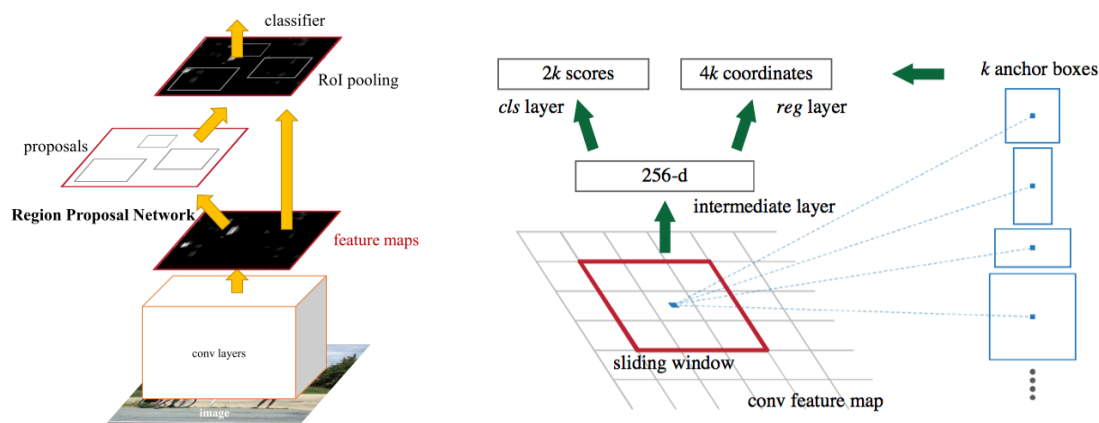


Figure 2.10: **Faster R-CNN:** Towards real-time object detection with region proposal networks [Ren et al., 2015]. **Left:** A single, unified network for object detection, **Right:** Region Proposal Network (RPN).

### 2.5.5 Multi-task Network Cascades (MNC)

MNC [Dai et al., 2016b] is a multi-stage multi-task network that performs instance-level semantic segmentation using convolutional neural networks. This is an extension of Faster R-CNN [Ren et al., 2015] that can only provide bounding-box around an object. Also, this is in contrast with semantic segmentation approaches that perform pixel-level labeling but are unable to identify object instances.

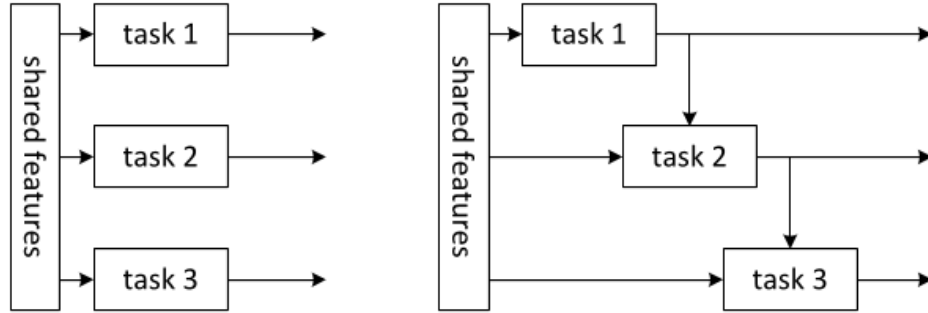


Figure 2.11: **MNC**: Multi-task network cascade [Girshick et al., 2014]. **Left**: Traditional multi-task learning, **Right**: A 3-stage cascade as in MNC.

MNC provides an end-to-end multi-stage trainable framework. The idea of MNC is to decompose the task of instance-level semantic segmentation into multiple related sub-tasks, which can be solved using shared features. In particular, MNC performs the following tasks in a single network:

1. Generate box-level object proposals using RPN [Ren et al., 2015].
2. Regress pixel-level segmentation mask for each object proposal.
3. Categorize object instances.

MNC tasks and its comparison with traditional multi-task learning is illustrated in Figure 2.11. In MNC, the multi-task loss is a combination of each individual task loss and is defined as

$$\mathcal{L} = \mathcal{L}_{RPN}(B(\theta)) + \mathcal{L}_{Mask}(M(\theta) | B(\theta)) + \mathcal{L}_{class}(C(\theta) | B(\theta), M(\theta)). \quad (2.56)$$

where  $\theta$  are network parameters and  $B$ ,  $M$ , and  $C$  represents the bounding-box coordinates, masks and classification scores, respectively. MNC also provides a differentiable RoI warping layer that can compute the gradient w.r.t. predicted box positions. The idea is to crop feature map regions corresponding to the bounding-boxes  $B$  and warp it into a target size (usually  $14 \times 14$ ) by interpolation. A standard max pooling is performed afterwards. The binary masks are learned using sigmoid cross-entropy loss and a standard softmax loss is minimized for classification. We refer the reader to [Dai et al., 2016b] for further details.

## 2.6 Large Scale Image Datasets

Datasets are the main driving force of all the learning algorithms described in the previous sections. Large scale datasets are generally required to learn rich features and high level representations which are not possible using hand-crafted features. This section presents a brief overview of the large-scale image detection, recognition and instance segmentation datasets which we will use in our experiments.

### 2.6.1 Pascal VOC

The Pascal visual object categories (VOC) dataset [Everingham et al., 2010] is a standard benchmark dataset used to evaluate object detection, classification and image segmentation techniques. Each image contains more than one object and the dataset contains 20 common object classes. This dataset is extensively labeled with bounding-boxes, object parts and pixel-level semantic segmentations.

The objects are categorized into four super-categories: 1) *Humans* (includes the person class only); 2) *Animals*, *i.e.*, bird, cat, cow, dog, horse, sheep; 3) *Vehicles*, *i.e.*, aeroplane, bicycle, boat, bus, car, motorbike, train; and 4) *Indoor objects*, *i.e.*, bottle, chair, dining table, potted plant, sofa, tv/monitor. Multiple versions have been introduced for this dataset but the most commonly used versions are 2007 and 2012. The

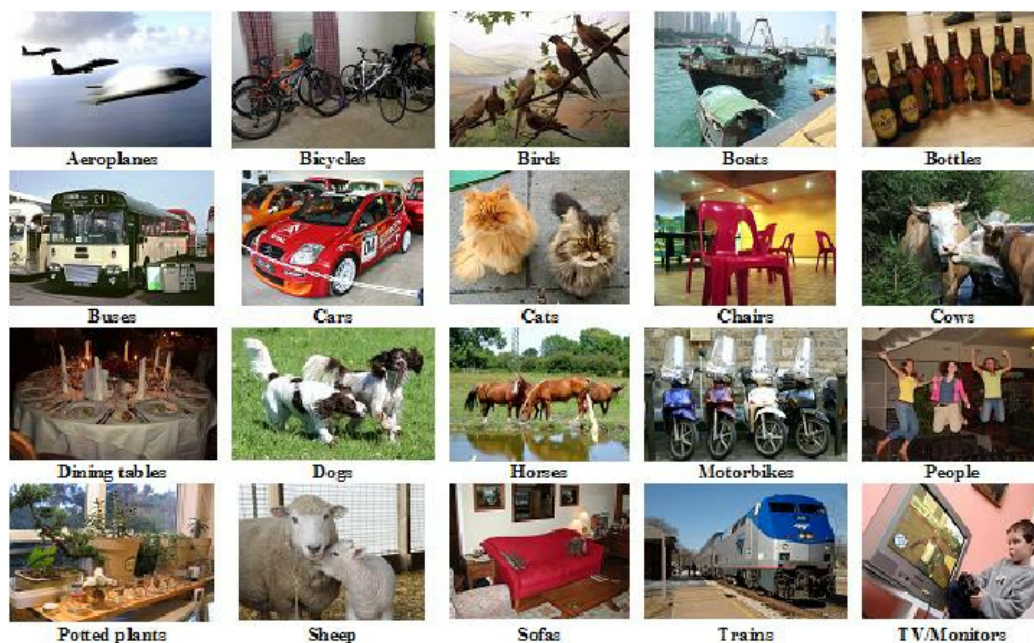


Figure 2.12: An example image for each category in the Pascal 2012 dataset [Everingham et al., 2010].

Pascal 2007 dataset has standard train, validation and test splits with a total 9,963 images containing 24,640 annotated objects. However, the Pascal 2012 dataset has a total of 11,530 train and validation images containing 27,450 bounding-box annotated objects and 6,929 segmentations. Furthermore, the Pascal 2012 dataset has a private test split of 10991 images for which annotations are not publicly available, thus an online submission is required to test the method's performance. Examples of each category are shown in the Figure 2.12.

### 2.6.2 Image-Net

The ImageNet dataset [Russakovsky et al., 2015] was created as part of the visual recognition challenge, which was the first of its kind for large scale multi-category benchmarking of object detection and recognition techniques. This dataset consists of more than a million images with fine-grained object class labels. Each image contains a per-image category label as well as the tight object bounding-box around it. The limitation of this dataset is that it only contains single objects mostly centered in the image, and thus cannot be used for rigorous object localization experiments. However, Imagenet has been widely used to pre-train a deep network, and the low level image filters learned using this dataset have proven their usefulness for multiple visual recognition tasks. The filters learned by the Alexnet [Krizhevsky et al., 2012] in its first convolutional layer are shown in figure 2.13. We only use this dataset for initializing our deep networks before fine-tuning for the specific experiments.

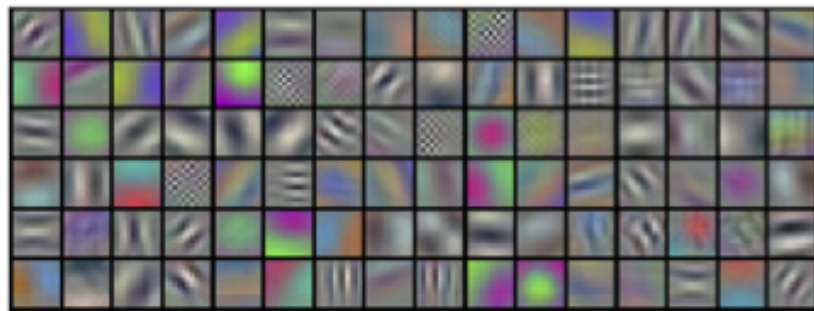


Figure 2.13: An illustration of the low-level filters learned using AlexNet [Krizhevsky et al., 2012] when trained on Imagenet dataset [Russakovsky et al., 2015].

### 2.6.3 Microsoft COCO

The Microsoft common objects in context dataset [Lin et al., 2014] extends the traditional object detection and classification datasets into more challenging object cate-



gories in their natural environments, where context is very important. This dataset contains 80 object categories which are labeled for 82,783, 40,504, 81,434 images in training, validation and test sets respectively. An important feature of this data is that it contains pixel-wise instance segmentations for over 886k objects in which 270k are human instances. This dataset is very challenging since many categories do not appear very frequently in the images, and the per image object density is very high compared to Pascal VOC [Everingham et al., 2010].

#### 2.6.4 Cityscapes

The Cityscapes dataset [Cordts et al., 2016] is the first high resolution dataset with an aim of high precision pixel-level instance and semantic segmentations. This dataset contains a large, diverse set of stereo video sequences recorded in streets from 50 different cities. Each 20th frame in the video sequence is manually labeled with very fine instance level segmentations. The rest of the 20k image frames have coarse instance segmentations, those are useful for semi-supervised learning or learning from weakly supervised data. A semantic segmentation ground-truth is illustrated in Figure 2.14.

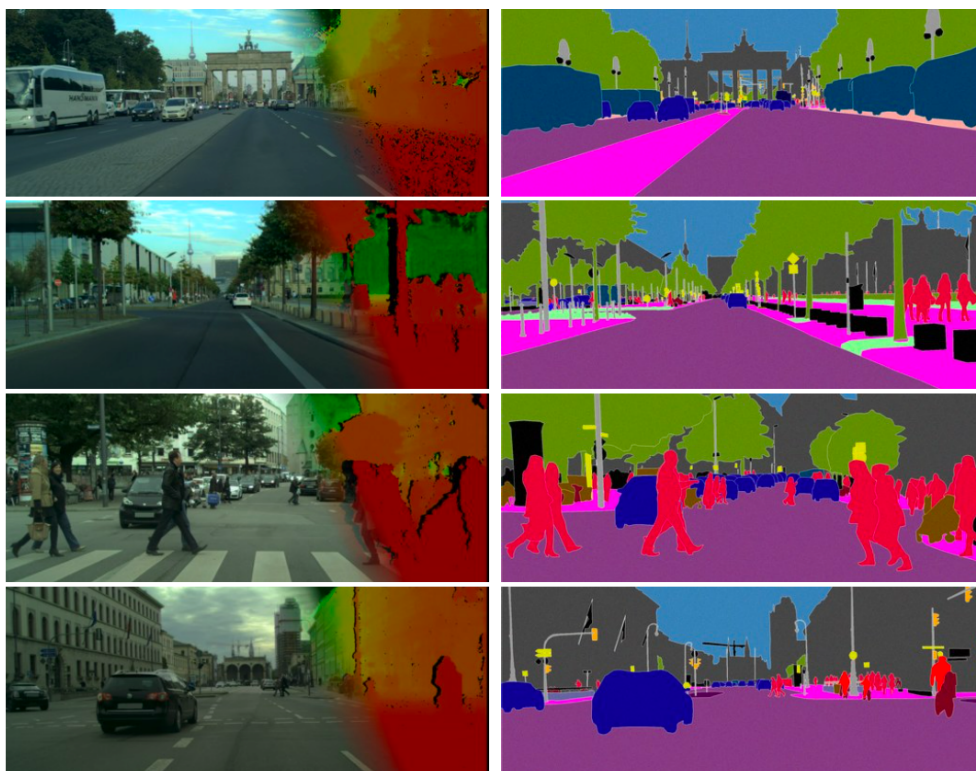


Figure 2.14: Few examples of Cityscapes dataset [Cordts et al., 2016] images and pixel-level level semantic segmentations.

## 2.7 Evaluation Metrics

An important aspect of model selection is to evaluate performance on a held-out subset of the data, or the validation set. A number of evaluation metrics are used to quantify how well an algorithm works. Almost all large-scale datasets provide a standardized toolbox to gauge a method's performance so as to compare it against different published results even without having those method's predictions.

In this section, we discuss a few important evaluation metrics used for object classification, detection, localization and instance-level semantic labeling. It is important to note that all the methods discussed in this thesis are expected to produce a continuous confidence score (or a probability estimate) rather than just a binary value. This is required to rank different possible solutions. An example of this is to produce a probability estimate for each of  $N$  classes while predicting the class label for an image. This probability estimate can be used to estimate the top-k accuracy for classification problems. Similarly, for the detection task, a number of bounding-boxes are generally required to be predicted with an associated confidence score. The provision of this confidence score allows results to be ranked such that the trade-off between false negatives and false positives can be evaluated.

**Precision:** Precision is a quantitative measure which tells how many correct predictions the model returns. This is usually estimated by computing the *Intersection over Union (IoU)* that in itself is an evaluation metric used to measure the accuracy of an object detector prediction with its ground-truth. This is also known as the *Jaccard index* and the Jaccard similarity coefficient. It is given by

$$\begin{aligned} P(A, B) &= \frac{A \cap B}{A \cup B} \\ &= \frac{A \cap B}{A + B - A \cap B} . \end{aligned} \tag{2.57}$$

The purpose is to assess the performance of an object detector (but with a threshold which will be discussed later in this section). In semantic labeling, a pixel-wise IoU is estimated instead of a bounding-box overlap.

**Recall:** Recall is a quantitative measure which tells how many of the positives the model returns. This is usually computed as

$$R(A, B) = \frac{A \cap B}{B} , \tag{2.58}$$

where  $A$  is the total number of predicted bounding-boxes and  $B$  the total number of positive boxes in an object detection dataset. The recall is usually expressed as a percentage.

**F1 score:** This is generally calculated by computing the harmonic average of precision and recall, and is defined as

$$F1(A, B) = \frac{2 \times P(A, B) \times R(A, B)}{P(A, B) + R(A, B)}, \quad (2.59)$$

where  $P(\cdot)$  and  $R(\cdot)$  represent precision and recall, respectively.

**Average precision (AP):** The average precision is computed by calculating the area under the precision curve for multiple recall thresholds. Generally, the area under the curve applies to binary classifiers that have a notion of a decision threshold internally. In an object detection system, generally a threshold of 50% is applied on the IoU to check if the predicted bounding-box is a *true* or *false* prediction. In order to estimate this curve, multiple thresholds are applied on the recall interval and a precision value is estimated to generate this curve. The area under this curve provides a single number that is useful to quantify the overall performance of a detector. Also, this curve highlights the areas of recall over which the detector performs best. Average precision is a more comprehensive metric and is the standard measure for all the detection and instance segmentation applications.

**Average recall (AR):** The average recall is computed by calculating the area under the recall curve for multiple IoU thresholds. In an object detection system, generally a varying threshold from 50% to 95% is applied on the IoU to estimate the recall in that interval. The area under this curve provides a single number that is useful to quantify the overall recall of the detector. Also, this curve highlights the areas of IoU over which the detector performs best.

**Mean Average Precision (mAP):** This metric is used to quantify the performance of all the classes present in the dataset. First of all, average precision is estimated for each individual category and then an average is estimated for all the categories to estimate a single number for the overall dataset.



---

## 2.8 Summary

In this chapter, we provided a brief introduction to supervised learning problem and probabilistic graphical modeling. In particular, we discussed the conditional random fields to incorporate structure and contextual information. We also presented techniques to compute hand-engineered representations as well as automatically learned deep representations using popular deep convolutional neural network architectures. In the end, an overview of large scale challenging image datasets and respective evaluation metrics was also provided. In the next chapter, we will present a method to learn class-specific object similarity.



---

# Learning Category-Specific Object Similarity

---

In this chapter, we address object co-detection problem that aims to leverage collective power of multiple images to achieve object detection simultaneously in all the images of a set. To this end, we introduce our category-specific object similarity learning technique. We formulate object co-detection as inference in a fully-connected conditional random fields (CRF) whose edges model the similarity between object candidates. We then learn a similarity function that allows us to efficiently perform inference in this fully-connected graph, even in the presence of many object candidates. This is in contrast with existing co-detection techniques that rely on exhaustive or greedy search, and thus do not scale well. Furthermore, the effectiveness of our approach is demonstrate via extensive experiments on several co-detection datasets. This chapter is based on our work [Hayder et al., 2014] and a substantial extension for multi-class joint similarity learning [Hayder et al., 2015] is available in Chapter 4.

## 3.1 Introduction

Object detection has been a central problem in modern computer vision, and has seen a surge of interest in recent years, which has lead to increasingly effective techniques. Much progress has been made in recent years, as demonstrated by the PASCAL challenge [Everingham et al., 2010] and the ImageNet challenge [Krizhevsky et al., 2012]. These techniques, however, still mostly perform detection based on local evidence in the input image. While some progress has been made towards exploiting scene context, the resulting methods typically only consider a single image at a time. Intuitively, however, the information contained jointly in multiple images should help overcoming phenomena such as occlusion and poor resolution. Whether working at instance [Lowe, 1999] or category [Felzenszwalb et al., 2010] level, most of the research has focused on detecting objects in a single image and in a sliding win-

dow manner. It is widely acknowledged, however, that such a myopic view is too restrictive as it ignores all contextual information [Galleguillos and Belongie, 2010]. On their own, the appearance cues of an object instance are often ambiguous due to poor resolution, occlusions, or challenging lighting conditions.

Previous work on object detection with context mainly exploits the 2D or 3D scene context observed in the same image as the detected objects [Hoiem et al., 2008; Desai et al., 2009]. Recently, simultaneously exploiting multiple images has been proposed as a means to gather broader contextual information for detection. The resulting *object co-detection* techniques [Bao et al., 2012; Guo et al., 2013; Shi et al., 2013] aim to jointly detect multiple instances of an object class from a pool of images. Intuitively, object co-detection leverages the weak appearance cues of object instances seen in multiple images to improve the robustness of object detection.

A critical challenge in object co-detection is to incorporate many object hypotheses from multiple images while keeping the joint classification of those object hypotheses tractable. Typically, the problem is formulated as that of inferring the (binary) activation labels of object candidates, which is a combinatorial search problem. The existing methods rely on either exhaustive search [Bao et al., 2012], or ad hoc greedy search [Shi et al., 2013]. While these strategies are effective for a small number of images, they are in general suboptimal, and become impractical when considering large image pools or number of classes.

In this chapter, we introduce a principled and efficient inference method for object co-detection. Given a pool of object candidates obtained by applying a pre-trained detector with a high recall rate (e.g., the Deformable Part-based Model (DPM) [Felzenszwalb et al., 2010]), we construct a fully-connected Conditional Random Field (CRF) where the nodes represent the candidate labels, and the edges encode the appearance similarity between two candidates. Inference in this CRF lets us predict the labels of all the object candidates simultaneously.

For our formulation to remain tractable, we need to be able to leverage efficient inference techniques in fully-connected CRFs. To this end, we model the similarity between two candidates as a linear combination of Gaussian kernels defined on multiple image features. The weights of this combination can be efficiently learned from training data. We make use of this similarity in the edge potentials of our CRF, which encode a data-dependent Potts model. The form of these potentials lets us utilize the efficient mean field inference algorithm of [Krähenbühl and Koltun, 2011], which not only yields the candidate labels, but also confidence in our predictions.

We evaluate our method on three benchmark co-detection datasets: the Pedestrian dataset [Ess et al., 2007], the Ford Car dataset [Bao et al., 2012] and the Human Co-Detection dataset [Shi et al., 2013]. In all three cases, our approach outperforms

---

the state-of-the-art co-detection methods, thus demonstrating the benefits of adequately modeling the relations between all object candidates via our fully-connected CRF formulation.

## 3.2 Putting Objects into Context

A natural perspective to improve the robustness of detectors to phenomena such as poor resolution, occlusions, and challenging lighting conditions, consists in putting objects into context. To this end, the scene properties of the target image are exploited to boost the object detection performance. For instance, Desai et al. [Desai et al., 2009] propose to jointly detect multiple object classes by defining a CRF on top of DPMs. In [Barinova et al., 2012], multiple instances of the same object class are jointly detected to address the occlusion problem. Hoiem et al. [Hoiem et al., 2008] consider the geometric context of the scene to improve detection by reducing the number of false positives. While these approaches have proven more effective than context-free detectors, they focus on exploiting the context from a single image. Intuitively, however, the information available in multiple images should be helpful to disambiguate detection.

### 3.2.1 Limitations of Existing Approaches

Object co-detection methods [Bao et al., 2012; Guo et al., 2013; Shi et al., 2013] were recently introduced to exploit the collective power of a set of images. In particular, the term *co-detection* was coined by [Bao et al., 2012], who tackle the problem by exhaustively searching for matching object instances in a set of object candidates. Generally speaking, co-detection has been considered for both 2D and 3D object models, as well as at category- and instance-levels. Category-level co-detection involves matching objects belonging to the same class (e.g., a person with another person) and appearing either in the same image, or in multiple images. In contrast, instance-level co-detection compares specific object instances (e.g., a specific person) that appear simultaneously in a pool of input images (e.g., [Shi et al., 2013]). While the original work of [Bao et al., 2012] could only handle pairs of images, [Guo et al., 2013] introduced a robust approach to multi-image co-detection that builds a shared low-rank representation of the object instances in multiple feature spaces.

### 3.2.2 Our Idea

Unlike existing object co-detection works, our method is based on a principled CRF formulation. Therefore, it enables us to perform joint inference efficiently for many

object instances extracted from multiple images. Our work is inspired by the fully-connected CRF model for semantic labeling of [Krähenbühl and Koltun, 2011; Vineet et al., 2012; Zhang and Chen, 2012]. By restricting the functional form of the pairwise potentials to a weighted mixture of Gaussian kernels defined on the input feature space, inference in this fully-connected CRF can be performed efficiently as a filtering operation. Here, instead of labeling the pixels in an image, we aim to label object candidates from multiple images in a principled and yet efficient manner. To the best of our knowledge, our work is the first attempt to extend the fully-connected CRF model of [Krähenbühl and Koltun, 2011] to another vision problem domain.

In this context, we propose to learn the pairwise potentials in our CRF by fitting a linear combination of kernels to a target similarity measure. This bears some connections with the multiple kernel learning literature [Gönen and Alpaydm, 2011; Vedaldi et al., 2009]. However, our objective is not to build a kernel-based similarity classifier as in [Shi et al., 2013], since this would not yield a mixture of Gaussian kernels adapted to our fully-connected CRF framework. Instead, our similarity learning approach is closer to metric learning [Weinberger et al., 2006]. In contrast, however, we jointly consider multiple kernels defined on separate feature spaces, thus yielding more flexibility than the single linear transformation typically used in metric learning methods.

### 3.3 A Fully-Connected CRF for Co-Detection

We now present our method for object co-detection, in which we aim to detect simultaneously all the instances of an object class in a group of  $S$  input images  $\mathcal{I} = \{I^1, \dots, I^S\}$ . As in [Bao et al., 2012; Guo et al., 2013; Shi et al., 2013], we dealing with  $C > 1$  object classes, we handle each class separately. Note that, while we discuss the case of category-level detection, our framework also applies to detecting the instances of specific objects (instance-level). Furthermore, we do not assume that each image contains only a single instance of an object class. Fig. 3.1 depicts an overview of our framework.

For each object class, our approach consists of two stages: first we generate a pool of object candidates in the form of bounding boxes obtained with a pre-trained object detector; then we formulate co-detection as a two-class labeling problem, where each candidate must be assigned either to the current object class of interest, or to a background class. These two steps are described in more detail in the remainder of this section.

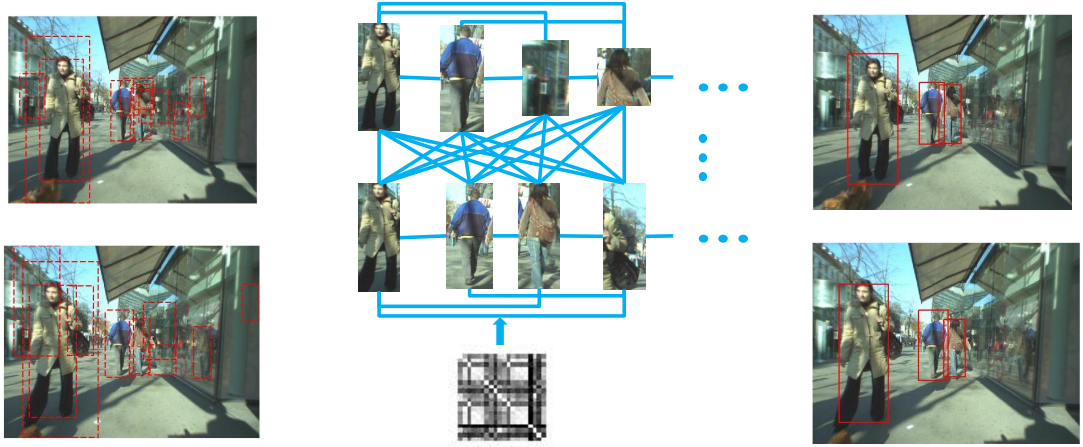


Figure 3.1: **Overview of our category-specific object similarity learning method.** Left: Original input images and candidates generated with a DPM; Middle: Fully-connected CRF on the candidates and corresponding learned pairwise similarities; Right: Jointly detected objects by efficient inference in the fully-connected CRF (actual result).

### 3.3.1 Object Candidate Generation

Following [Bao et al., 2012], given a target object class  $c$ , we first apply a pre-trained DPM [Felzenszwalb et al., 2010] to each input image and extract a set of object candidates, denoted by  $\mathcal{X}^c = \{\mathbf{X}_1^c, \dots, \mathbf{X}_{N_c}^c\}$ . To prevent entirely missing some objects in this first stage, we adjust the threshold of each detector and the non-maximum suppression parameters so as to achieve a high recall rate for all target classes. Note that, while we employ DPMs, any object detector that outputs a bounding box can be employed in our candidate generation stage. Fig. 3.2 shows some examples of object candidates generated for three different object co-detection datasets.

Given the set of candidates  $\mathcal{X}^c$  for class  $c$ , we then adopt a part-based representation as in the DPM. An object candidate  $\mathbf{X}_i^c$  is represented by its root  $\mathbf{r}_i^c$  and a set of  $k$  parts  $\mathcal{P}_i^c = \{\mathbf{p}_{i,1}^c, \dots, \mathbf{p}_{i,k}^c\}$ , together with the image window  $\mathbf{W}_i^c$  corresponding to the object bounding box. For each candidate  $\mathbf{X}_i^c$ , we also compute a set of appearance features from its image window  $\mathbf{W}_i^c$ . These features, denoted by  $\mathbf{f}_{i,s}^c$  for a specific feature type  $s$ , capture the color and texture properties of the candidate.

### 3.3.2 CRF Formulation

Given the candidate pool  $\mathcal{X}^c$ , we formulate object co-detection as the problem of jointly labeling the candidates with the corresponding object or background class. More specifically, we introduce a label variable  $y_i^c$  for each object candidate  $\mathbf{X}_i^c$ , which

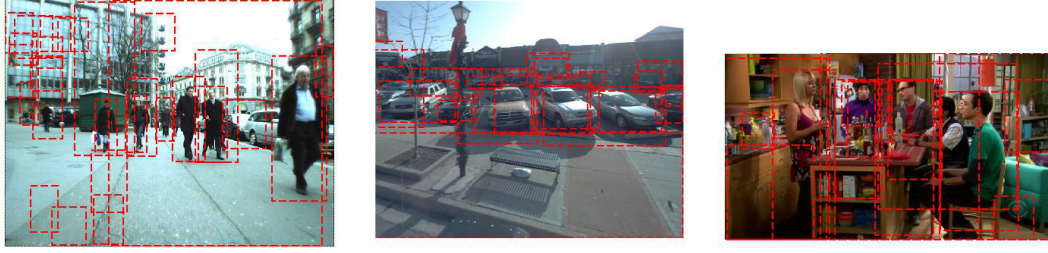


Figure 3.2: **Sample object candidates generation results.** Left: Pedestrian dataset [Ess et al., 2007]; Middle: Ford Car dataset [Bao et al., 2012]; Right: Human Co-detection dataset [Shi et al., 2013].

takes either the object class label  $l^c$ , or the background label  $l^0$ .

To appropriately capture the dependencies of our object candidates, we build a fully-connected Conditional Random Field (CRF) on the label variables  $\mathcal{Y}^c = \{y_1^c, \dots, y_{N_c}^c\}$ . Each node in the CRF corresponds to the label of one object candidate, and any pair of two candidates are connected by an edge that encodes their relationship. Formally, we define the joint distribution over the label variables  $\mathcal{Y}^c$  given the observed candidates  $\mathcal{X}^c$  as

$$P(\mathcal{Y}^c | \mathcal{X}^c) = \frac{1}{Z(\mathcal{X}^c)} \exp \left( - \sum_{i=1}^{N_c} \phi_u(y_i^c | \mathbf{X}_i^c) - \alpha \sum_{i=1}^{N_c} \sum_{j>i} \psi_p(y_i^c, y_j^c | \mathbf{X}_i^c, \mathbf{X}_j^c) \right), \quad (3.1)$$

where  $Z(\cdot)$  is the partition function,  $\alpha$  is a weight learned by cross-validation, and  $\phi_u$  and  $\psi_p$  are the unary and pairwise potential functions, respectively. The unary potential  $\phi_u$  encodes how likely a candidate is to be associated with each class, while the pairwise potential  $\psi_p$  measures the affinity between the different possible class assignments of two candidates.

Object co-detection then boils down to inferring the optimal label configuration of this CRF model, which jointly labels all the object candidates. In our work, we do not put any restriction on the number of input images. Consequently, we may have a large number of object candidates (nodes) in our CRF. Inference in such a large, fully-connected CRF is in general intractable and difficult to approximate. The key challenge therefore lies in finding an efficient inference procedure in our fully-connected CRF.

To address joint inference in a principled way, we rely on the formulation of [Krähenbühl and Koltun, 2011] to design our CRF model. The main requirement of this formulation is that the pairwise potentials must have the form of a mixture of Gaussian kernels. In the following, we discuss the potential functions employed in our model, and, in particular, introduce pairwise potentials that meet this mixture-of-Gaussian-



kernels requirement and, as we will show, are effective for co-detection.

### 3.3.2.1 Unary Potentials.

The unary potentials measure the likelihood that a candidate  $\mathbf{X}_i^c$  belongs to the object class and to the background class. Following [Bao et al., 2012], we use a rescaled DPM score as unary potential. This lets us write our unary term as

$$\phi_u(y_i^c | \mathbf{X}_i^c) = \begin{cases} E_r(\mathbf{r}_i^c, \mathbf{W}_i^c) + \sum_{j=1}^k \left( E_p(\mathbf{p}_{i,j}^c, \mathbf{W}_i^c) + E_d(\mathbf{r}_i^c, \mathbf{p}_{i,j}^c) \right) & \text{if } y_i^c = l^c \\ 0 & \text{if } y_i^c = l^0, \end{cases} \quad (3.2)$$

where  $E_r$  and  $E_p$  are the unary potentials for the root and part filters respectively.  $E_d$  encodes the deformation cost between the root  $\mathbf{r}_i^c$  and each part  $\mathbf{p}_{i,j}^c$ .  $E_r$ ,  $E_p$  and  $E_d$  are directly defined as in the original DPM [Felzenszwalb et al., 2010]. Note that, in principle, if the candidate was generated by another detector, we could still extract the DPM model parameters from  $\mathbf{W}_i^c$  and make use of this unary potential.

### 3.3.2.2 Pairwise Potentials.

The pairwise potential  $\psi_p$  is a data-dependent smoothing term that encourages similar hypotheses to share the same object label. As in [Krähenbühl and Koltun, 2011], we restrict our pairwise potential to take the form of a weighted mixture of Gaussian kernels, which can be expressed as

$$\psi_p(y_i^c, y_j^c | \mathbf{X}_i^c, \mathbf{X}_j^c) = \mu(y_i^c, y_j^c) \sum_{m=1}^M w^m k^{(m)}(\mathbf{f}_i^c, \mathbf{f}_j^c), \quad (3.3)$$

where  $\{w^m\}$  are the weights of the Gaussian kernels  $\{k^{(m)}\}$ , and  $\mu$  is a label compatibility function. In particular, we make use of this function to encode a data-dependent Potts model, i.e.,  $\mu(y_i, y_j) = \mathbf{1}_{y_i=y_j}$ .

The mixture of Gaussian kernels measures the appearance similarity between two object candidates. To this end, we use multiple feature types, as well as multiple kernel parameters. For each feature type  $\mathbf{f}_s$ , we construct a series of kernel functions of the form

$$k(\mathbf{f}_{i,s}^c, \mathbf{f}_{j,s}^c; t, \sigma_s) = \exp\left(-\frac{\|\mathbf{f}_{i,s}^c - \mathbf{f}_{j,s}^c\|^2}{2^t \sigma_s^2}\right), \quad (3.4)$$

where  $\sigma_s$  is the minimum kernel width and  $t$  is an integer. We enumerate the value of  $t$  from 1 to  $T$  to define our series of kernels. Using kernels with different widths

provides us with more flexibility in the representation of the similarity. The mixture of Gaussian kernels in Eq. 5.6 is then obtained by summing over all feature types and all values of  $t$  in each type. As will be discussed in Section 3.3.3, to avoid having to manually tune the weights  $\{w^m\}$  of this mixture, we propose an efficient supervised learning procedure to estimate these weights.

### 3.3.2.3 Efficient Co-detection

Given our fully-connected CRF model, we jointly detect the object instances in the input images by performing maximum posterior marginal inference. Following [Krähenbühl and Koltun, 2011], we adopt a fast mean field approximation algorithm to compute the marginals. Given the current mean field estimates  $\{Q_i\}$  of the marginals, the update equation can be written as

$$Q_i(y_i^c = l) \propto \exp \left( -\phi_u(y_i^c) - \sum_{l' \neq l} \sum_{j \neq i} Q_j(y_j^c = l') \psi_p(y_i^c, y_j^c) \right). \quad (3.5)$$

Due to the mixture of Gaussian kernels form of the pairwise term, the updates can be computed in parallel by convolution with Gaussian kernels. This can be achieved efficiently by exploiting fast Gaussian filtering techniques, such as the permutohedral lattice-based method of [Baek et al., 2013].

After convergence, we obtain an (approximate) posterior distribution of object labels for each node (i.e., object candidate). To obtain the final co-detection results, we can then compute the most likely label for each object candidate,  $\hat{y}_i^c = \arg \max_{y_i^c} Q_i(y_i^c)$ . Furthermore, we can also exploit the mean field approximate marginal probability  $Q_i(\hat{y}_i^c)$  as a detection score.

Note that, with our pairwise potential and since we treat each class separately, our CRF models a binary problem with a submodular energy function. As such, it could in principle be solved exactly by the graph-cut algorithm [Boykov et al., 2001; Kolmogorov and Zabini, 2004]. However, to achieve efficiency, the conventional graph-cut algorithm [Boykov et al., 2001] relies on the sparse connectivity of the graph. As will be shown in Section 3.4, a graph-cut solution to our inference problem becomes significantly slower than our efficient filtering-based mean field solution when dealing with large densely connected random fields. Furthermore, note also that the MAP estimate from graph-cut does not provide a confidence score for the detection. Finally, despite that in this work we focus on a binary labeling problem, our formulation easily extends to the multi-class scenario.

### 3.3.3 Learning Object Similarity

Recall that our pairwise potentials encode the appearance similarity between two object candidates as a mixture of Gaussian kernels. To suitably adjust the weights of the mixture to the problem at hand, we can exploit training data and learn the weights that minimize the deviation from an ideal similarity measure. Here we formulate kernel weights estimation as a least-squares regression problem, where the ground-truth (binary) similarity is directly defined by the compatibility of the labels of two object instances.

More specifically, we build a training set of object pairs,  $\mathcal{D} = \{(\mathbf{X}_{i,1}, \mathbf{X}_{i,2}, s_i)\}$ , where  $s_i$  is the ground-truth similarity, taking value 1 if  $\mathbf{X}_{i,1}$  and  $\mathbf{X}_{i,2}$  belong to the same class (excluding background) and 0 otherwise. The weights of the kernels can then be estimated by solving the optimization problem

$$\hat{\mathbf{w}} = \underset{\mathbf{w}}{\operatorname{argmin}} \frac{1}{|\mathcal{D}|} \sum_{i=1}^{|\mathcal{D}|} \left( s_i - \sum_{m=1}^M w^m k^{(m)}(\mathbf{f}_{i,1}, \mathbf{f}_{i,2}) \right)^2, \quad (3.6)$$

where  $\mathbf{w} = \{w^1, \dots, w^M\}$  contains the weights of all kernels for all feature types. Note that this is a least-squares problem, and that its solution can therefore be obtained in closed-form.

To compute ground-truth similarities for category-level co-detection, we employ the following procedure. For each object class  $c$ , we first apply the same pre-trained object detector with high recall on the training images, and compute the intersection-over-union (IOU) of each detected bounding box with respect to the ground-truth bounding boxes. A detected bounding box is said to belong to class  $c$  if its maximum IOU w.r.t. ground-truth bounding boxes is larger than 50%. Otherwise, it is labeled as background. The training set  $\mathcal{D}$  can then be constructed by collecting all possible pairs of detected bounding boxes and setting their similarity to 1 if they were both found to belong to class  $c$ , and 0 otherwise. In practice, the number of such pairs grows quickly, and we therefore randomly subsample the dissimilar pairs to build a balanced training set. Note that, as will be shown in our experiments, this procedure can also make use of instance-level labels when available, even if the final task remains category-level co-detection. In this scenario, two bounding boxes are considered similar only if they depict the same instance from the general category  $c$ .

For the similarity, we used the square loss due to its simplicity (closed-form solution). The hinge loss would also be a possible choice. However, the logistic function could not be employed, since we need the similarity to take the form of a mixture of Gaussian kernels.

## 3.4 Experiments

In this section, we study the effectiveness of our approach and compare it against state-of-the-art co-detection baselines.

### 3.4.1 Datasets and Setup

We evaluate our framework on several standard object co-detection datasets. These datasets include the Ford Car dataset of [Bao et al., 2012] and the Pedestrian dataset of [Ess et al., 2007], which provide category-level labels for the bounding boxes. Furthermore, we also employ the Human Co-detection (HCD) dataset of [Shi et al., 2013], which provides instance-level annotations of the bounding boxes. Note, however, that the task for HCD remains that of category-level co-detection, but, as suggested in [Shi et al., 2013], the instance-level annotations can be employed to better model object similarities.

In our experiments, we used the version 4 of DPMs [Felzenszwalb et al.], since it was also employed in [Guo et al., 2013; Shi et al., 2013]. This version provides one root and eight parts for each object. For each object candidate, we computed a 59 dimensional Local Binary Patterns (LBP) feature and a 32 dimensional color histogram feature on the H channel of the HSV color-space. Note that the dimensionality of these features can be reduced using PCA to speed up inference. Our method has three hyper-parameters: the number of kernels, the widths  $\sigma_s$  and the pairwise weight  $\alpha$ . These parameters were obtained by two-fold cross validation.

We compare our method with the DPM baseline [Felzenszwalb et al., 2010] and the following state-of-the-art co-detection approaches: 1) Object Co-detection [Bao et al., 2012]; 2) Multi-feature Joint Low-Rank Reconstruction [Guo et al., 2013]; 3) Human Co-detection and Labeling [Shi et al., 2013]. To study the effect of using different image features in our similarity kernels, we also consider two simpler versions of our approach, each of which uses only one feature type (either LBP [Ojala et al., 2002] or color histograms). We refer to these two systems as LBP-CRF and Color-CRF, respectively, and to our full system as Joint-CRF.

In our results, each bounding box is labeled based on the mean field approximate marginal probability of the object class. This lets us compute precision-recall curves, as opposed to a single point on these curves if we used the MAP estimate. We report average precision (AP) at category level following the evaluation metric in the PASCAL VOC challenge. For a bounding box to be considered correct, it must have at least 50% overlap with one of the ground truth bounding boxes in that image. This also has the advantage of making our results directly comparable to previously-

Methods with Stereo Pairs	Ped(all)	Ped(h>120)
DPM [Felzenszwalb et al.]	59.7	55.4
Obj. Co-detection [Bao et al., 2012]	62.7	63.4
Robust Obj. Co-detection [Guo et al., 2013]	67.8	70.1
Human CoDeL [Shi et al., 2013]	74.4	73.8
LBP-CRF	77.04	79.43
Color-CRF	77.99	79.70
Joint-CRF	<b>78.73</b>	<b>81.25</b>

Table 3.1: **Pedestrian co-detection with stereo pairs:** Comparison of our approach with state-of-the-art co-detection methods on Pedestrian dataset using stereo pairs.

reported ones. Therefore, for the baselines, we directly quote results reported in [Bao et al., 2012; Guo et al., 2013; Shi et al., 2013]. Our results for all experiments were averaged over 10 random training/test partitions.

### 3.4.2 Results and Discussion

#### 3.4.2.1 Pedestrian Dataset.

The Pedestrian dataset consists of 476 training images and 374 test images from two video sequences of street scenes acquired with a stereo setup. Each image has a resolution of  $640 \times 480$  and contains multiple people. To evaluate our co-detection framework, we follow the same scenario as other co-detection work, which address the problem of jointly detecting people in a pair of images. Note that this dataset provides ground truth labels only for the left images in the stereo pairs. To mimic the stereo scenario, we therefore follow the same strategy as [Guo et al., 2013] and generate pseudo-stereo pairs by randomly drawing pairs of images that are no more than 3 frames apart in the left sequences. We generate 476 training pairs from the left training sequence and 300 test pairs from the left test sequence in this manner.

In Tables 3.1 and 3.2, we report the results of our approach and the baselines on this dataset. Note that our approach significantly improves the results of the

Methods with Random Pairs	Ped(all)	Ped(h>120)
DPM [Felzenszwalb et al.]	59.7	55.4
Obj. Co-detection [Bao et al., 2012]	58.1	58.1
Robust Obj. Co-detection [Guo et al., 2013]	67.7	70.3
Joint-CRF	<b>77.7</b>	<b>80.43</b>

Table 3.2: **Pedestrian co-detection with random pairs:** Comparison of our approach with state-of-the-art co-detection methods on Pedestrian dataset using random pairs.

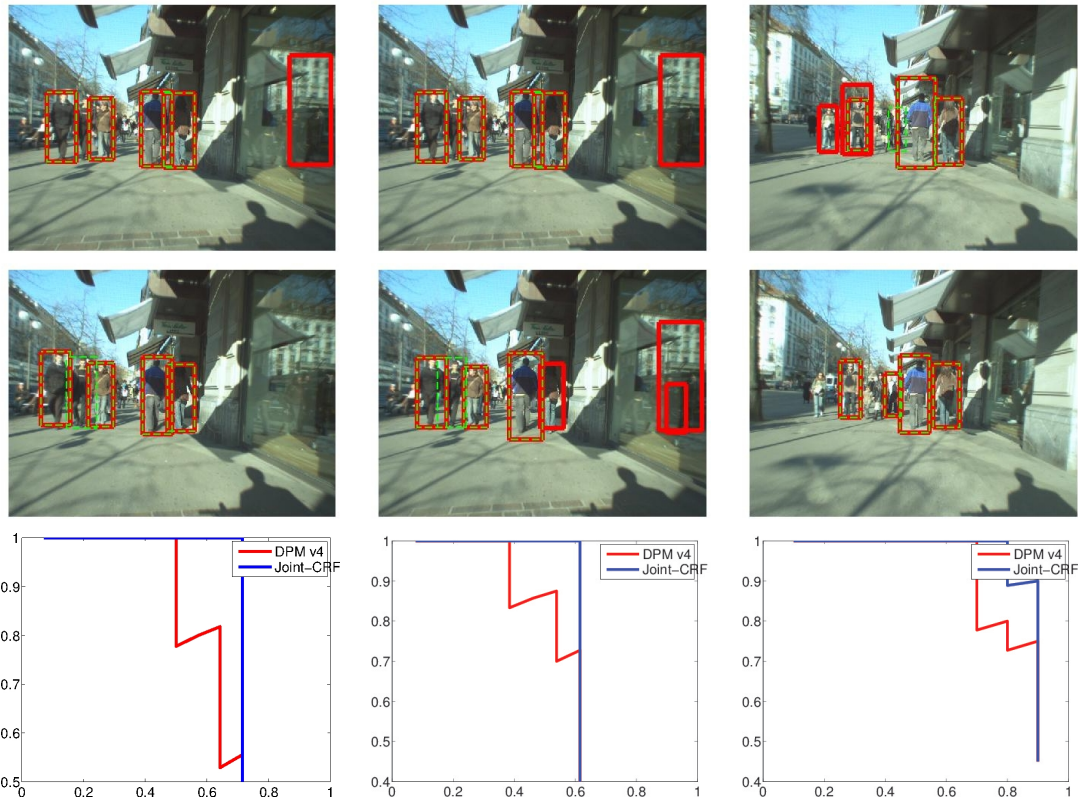


Figure 3.3: **Pedestrian co-detection results:** Examples of our co-detection results on test pairs of the Pedestrian dataset. Top two rows: Input image pairs (Green dash: our results, Red solid: DPM results); Bottom row: Precision-recall curves of our method and DPM for the three image pairs.

DPMs. More importantly, we also outperform all the baselines, even [Shi et al., 2013] that is specifically dedicated to the human co-detection case. This is also true for the single-feature versions of our model (LBP-CRF and Color-CRF). Combining these two features in our Joint-CRF model nonetheless lets us further improve performance. Sample co-detection results are given in Fig. 3.3. The overall precision v.s. recall curve is shown in Fig. 3.7.

We then study the quality of the similarity function learned from the training pairs using the method described in Section 3.3.3. To this end, in Fig. 3.4, we compare the similarity matrix obtained by applying the learned function to the candidates in one test pair with the corresponding ground-truth similarity computed from the correct labels (pedestrian vs background). Note that the predicted similarity depicts a similar pattern to the ground-truth one. This is further evidenced by the histogram that shows that pedestrian candidates have a high similarity score.

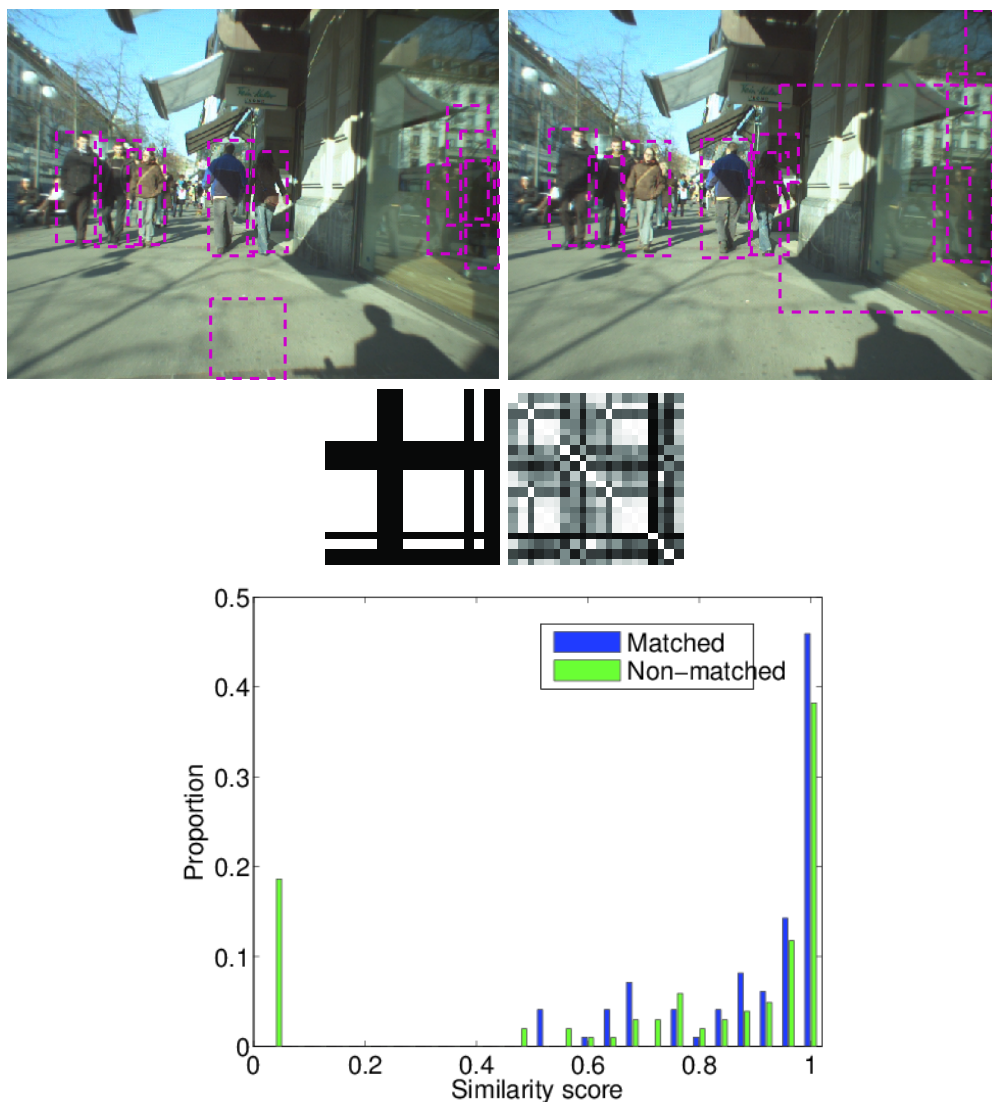


Figure 3.4: **Predicting similarity:** Sample similarity matrix obtained by applying our learned similarity function to one test pair of the Pedestrian dataset. Top: Input image pair; Middle: (Left) target (ground truth) similarity matrix, (Right) learned similarity matrix (brighter means more similar); Bottom: Normalized histograms of similarity scores for matched and non-matched candidate pairs.

### 3.4.2.2 Ford Car Dataset.

The Ford Car dataset consists of five scenes, each of which contains 86 stereo images. Each image has a resolution of  $781 \times 601$  and depicts multiple instances of cars at different scales and orientations. We made use of the 300 pseudo-stereo pairs provided by [Bao et al., 2012], which were generated in the same manner as described above for the Pedestrian dataset. Since no training pairs are provided, we extracted them

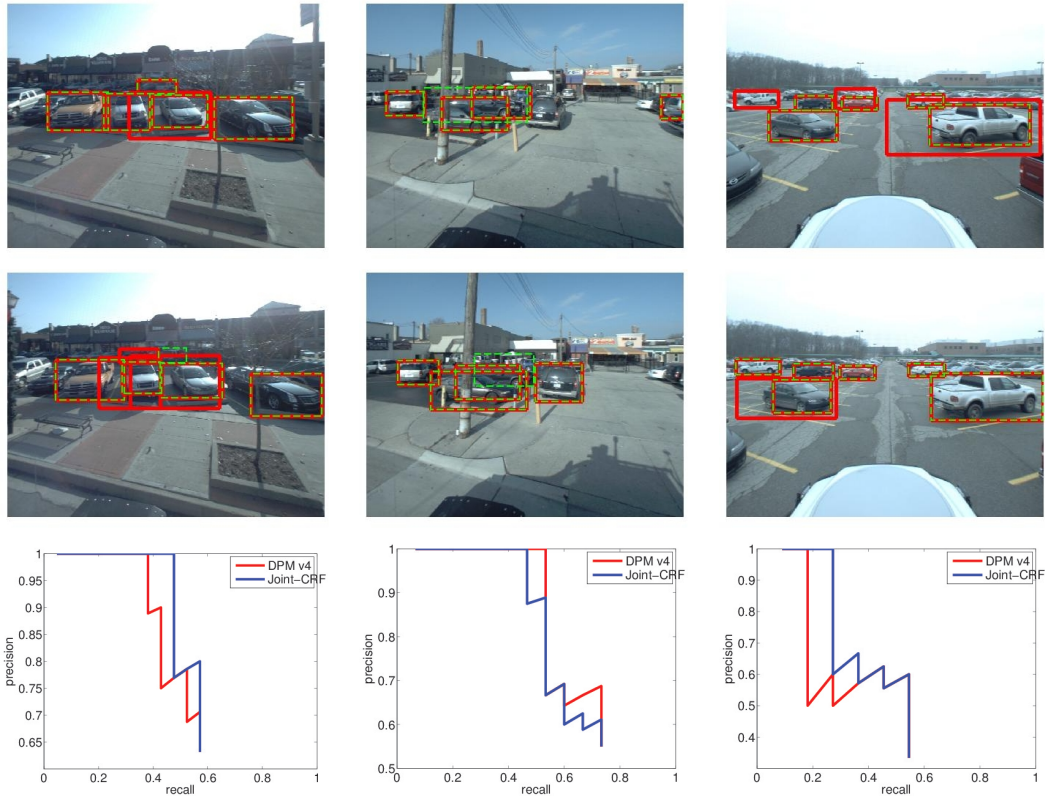


Figure 3.5: **Car co-detection Results:** Examples of our co-detection results on test pairs of the Ford Car dataset. Top two rows: Input image pairs (Green dash: our results, Red solid: DPM results); Bottom row: Precision-recall curves of our method and DPM for the three image pairs.

in the same fashion, while ensuring no overlap with the test pairs. This resulted in a total of 410 training pairs.

The results of our approach and the baselines on this dataset are reported in Table 3.4. Note that, since it is dedicated to human co-detection, the method of [Shi et al., 2013] does not apply to this dataset. As in the Pedestrian case, our approach

Method with Stereo Pairs	Ford(all)	Ford(h>80)
DPM [Felzenszwalb et al.]	49.8	47.1
Obj. Co-detection [Bao et al., 2012]	53.5	55.5
Robust Obj. Co-detection [Guo et al., 2013]	55	57.5
LBP-CRF	60.13	61.67
Color-CRF	59.44	60.45
Joint-CRF	<b>60.77</b>	<b>61.45</b>

Table 3.3: **Car co-detection with stereo pairs:** Comparison of our approach with state-of-the-art co-detection methods on the Ford Car dataset using stereo pairs.



Method with Random Pairs	Ford(all)	Ford(h>80)
DPM [Felzenszwalb et al.]	49.8	47.1
Obj. Co-detection [Bao et al., 2012]	50.0	49.1
Robust Obj. Co-detection [Guo et al., 2013]	55.1	57.5
Joint-CRF	<b>62.49</b>	<b>59.13</b>

Table 3.4: **Car co-detection with random pairs:** Comparison of our approach with state-of-the-art co-detection methods on the Ford Car dataset using random pairs.

yields a significant performance improvement over the baselines. This is the case both with the single-feature models and with our Joint-CRF model. Sample co-detections are provided in Fig. 3.5. The overall precision v.s. recall curve is shown in Fig. 3.7.

Similarly to the Pedestrian case, in Fig. 3.6, we illustrate the quality of the learned similarity function by depicting the similarity matrix obtained when applying this function to one test pair. We can again see that the predicted similarity correctly reflects the ground-truth one.

### 3.4.2.3 Human Co-Detection Dataset.

The HCD [Shi et al., 2013] comprises 387 images separated into 26 sets. Each image may contain multiple people, and the appearance of these people is consistent within one set. As opposed to the Ford Car and Pedestrian datasets where only two images with relatively small viewpoint difference are employed for co-detection, all the images in one set of HCD are considered simultaneously and typically depict large viewpoint changes. For our experiments, we followed a leave-five-sets-out strategy, which amounts to using roughly 80% of the images as training data and the remaining 20% (coming from five independent sets) as test images. For this dataset, we employed the provided instance-level labels to learn our similarity function. Note however that the main task remains category-level (human) co-detection.

We report our results on this dataset in Table 3.5. Note that the only reported

	HCD Dataset
DPM [Felzenszwalb et al.]	69.64
Human CoDeL [Shi et al., 2013]	74.94
LBP-CRF	78.81
Color-CRF	79.19
Joint-CRF	<b>79.41</b>

Table 3.5: **Human co-detection:** Comparison of our approach with state-of-the-art co-detection methods on the HCD dataset

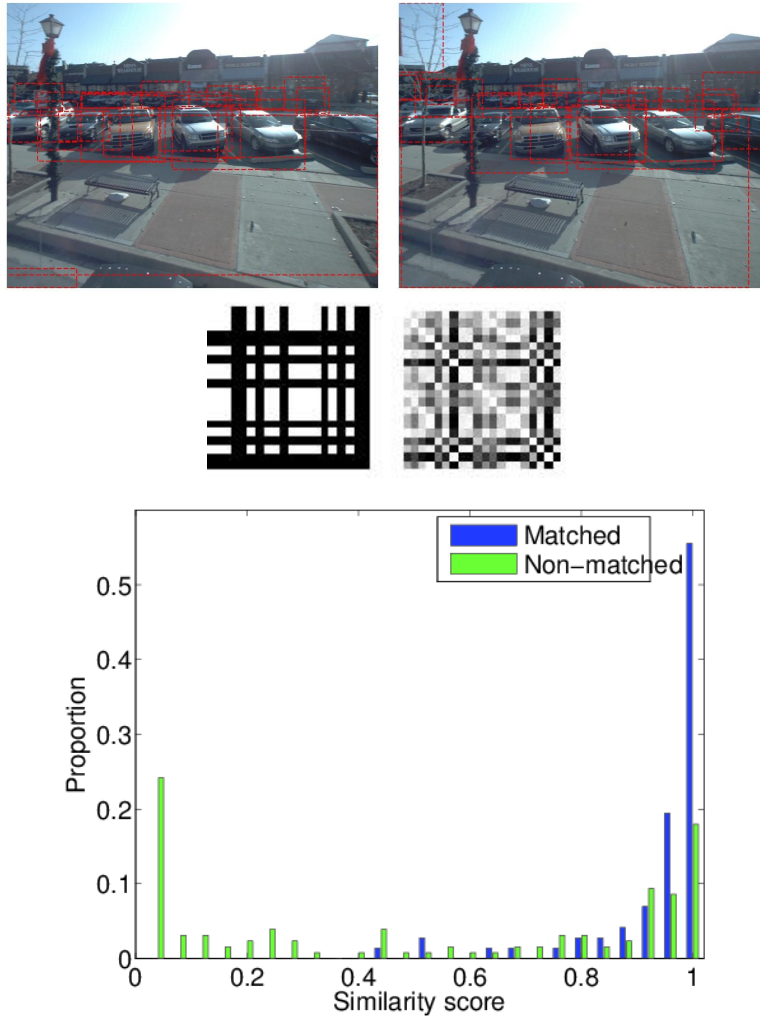


Figure 3.6: **Predicting similarity:** Sample similarity matrix obtained by applying our learned similarity function to one test pair of the Ford Car dataset. Left: Input image pair; Middle: (Top) target (ground truth) similarity matrix, (Bottom) learned similarity matrix (brighter means more similar); Right: Normalized histograms of similarity scores for matched and non-matched candidate pairs.

results on HCD are those of [Shi et al., 2013]. As for the other datasets, we outperform the baselines, whether using a single feature type or multiple ones. Fig. 3.8 depicts some of our co-detection results on HCD. For this dataset, we also compare our results with those of a sparse CRF constructed by connecting only the first  $k$  nearest neighbors (based on our similarity) of each node, with  $k$  ranging from 1 to 50. The best F1 score of this sparse CRF (over all values of  $k$ ) is 80.45%. This is clearly outperformed by our F1 score of 85.3%.

In Fig. 3.9, we also show the predicted similarity function for one of the sets in the dataset. Note that, because of the instance-level annotations, the similarity matrix is

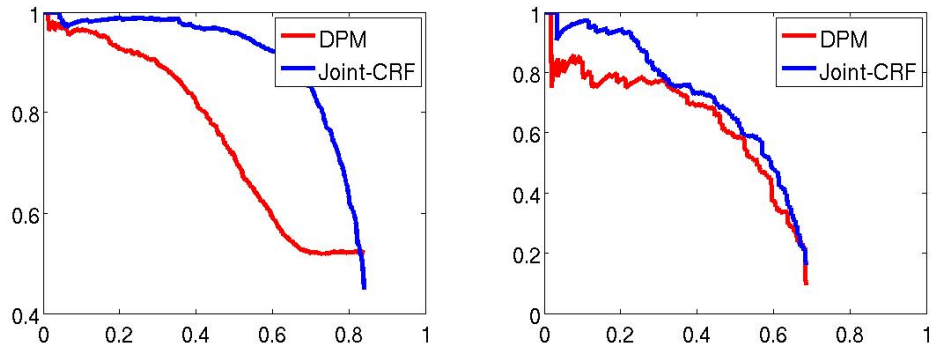


Figure 3.7: **Category-specific object co-detection precision-recall curves:** The precision-recall curves over all pairs (Blue solid: our results, Red solid: DPM results). Left: Pedestrian dataset ; Right: Ford car dataset.

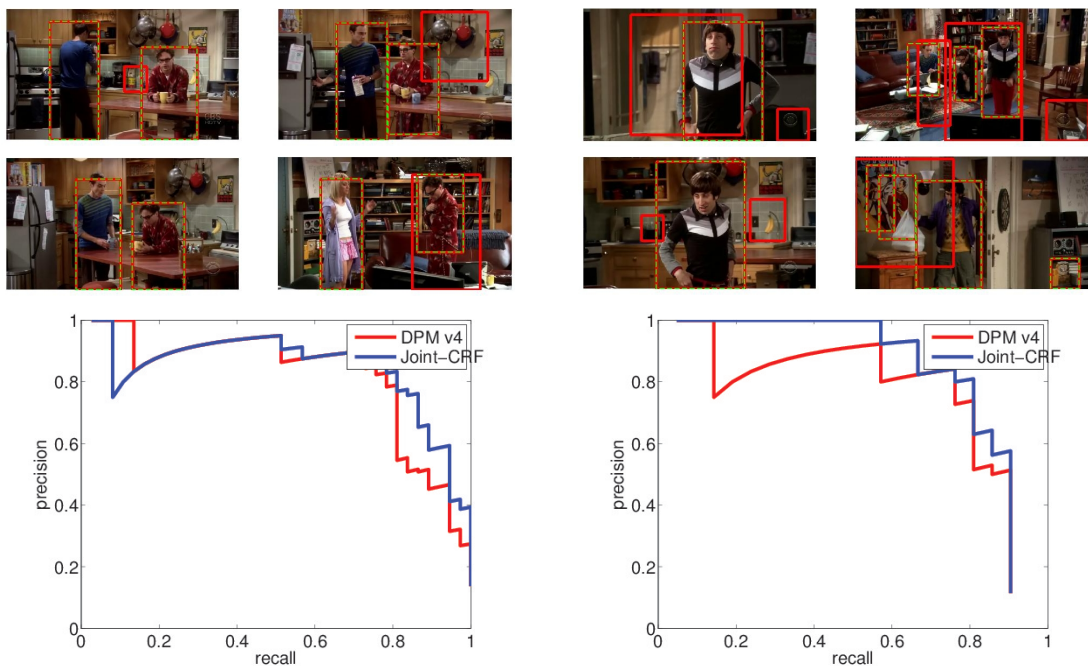


Figure 3.8: **Sample Results:** Examples of our co-detection results on two sets from the Human Co-detection dataset. Top: Input image set overlaid with detection output (Green dash: our results, Red solid: DPM results); Bottom: Precision-recall curves of our method and DPM for the two sets.

more complex than before. Nonetheless, our predicted similarity matrix still yields a good approximation of the ground-truth one.

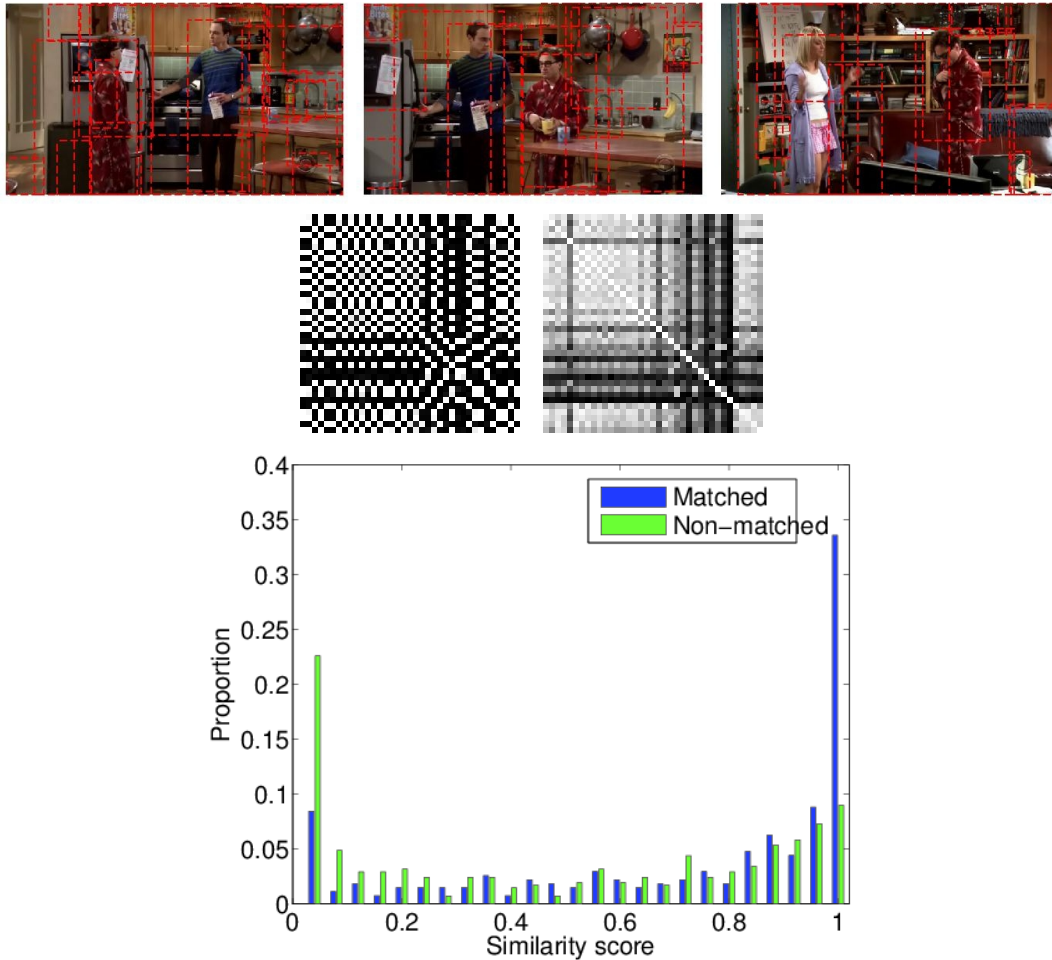


Figure 3.9: **Predicting similarity:** Sample similarity matrix obtained by applying our learned similarity function to one test set in the Human Co-Detection dataset. Top: Input images (three examples); Middle: (Left) target (ground truth) similarity matrix, (Right) learned similarity matrix (brighter means more similar); Bottom: Normalized histograms of similarity scores for matched and non-matched candidate pairs.

#### 3.4.2.4 Scaling up.

As mentioned earlier, inference in our model could in principle also be performed using the Graph-cut algorithm [Boykov et al., 2001]. Here, we study the scalability of both inference strategies with respect to the number of images considered jointly at test time. There are two reasons not to employ graph-cut with a sparse graph structure: (i) For co-detection, determining the sparse graph connectivity must rely on a heuristic; (ii) Graph-cut only generates a MAP estimate, which does not directly yield confidence values. Our method provides the marginal probabilities as confidence scores. To this end, we build two fully-connected CRF models with dif-

---

ferent numbers of test images from the HCD dataset. The first CRF has 330 nodes. In this case, our mean field filtering inference takes 6.5 seconds and the Graph-cut only takes 0.6 second (this includes the time to compute the potential functions). However, when the size of the CRF increases by a factor 10, our method takes 8.5 seconds, which is only mildly slower compared to the previous setting. In contrast, the Graph-cut spends 30 seconds in potential calculation and 60 seconds in inference. This shows that the Graph-cut algorithm does not scale up to larger test sets for fully connected graphs, and thus confirms our choice of inference strategy.

### 3.5 Conclusion

In this chapter, we have introduced a formulation of object co-detection from a pool of images that expresses the problem as inference in a fully-connected CRF whose nodes represent object candidates. We have then shown that modeling the similarity between pairs of candidates as a weighted mixture of Gaussian kernels allowed us to efficiently perform inference in our graph, while yielding an effective representation for co-detection. Importantly, the efficient inference strategy has been applied to semantic pixel labeling, we believe that adapting it to a novel problem domain is an interesting contribution in itself, which led to the additional challenge of designing suitable, yet effective pairwise potentials. Our experimental evaluation has demonstrated that our approach could effectively leverage the information in multiple images to improve detection accuracy, thus outperforming existing co-detection techniques on benchmark datasets. In the next chapter, we study how our framework can be applied to jointly co-detecting different object categories, thus leveraging the collective power not only of multiple images, but also of multiple classes.



---

# Multiclass Structural Kernel Boosting

---

Exploiting contextual relationships across images has recently proven key to improve object detection. In Chapter 3, we have introduced a simple model to learn category-specific object similarity which leverages the collective power of multiple images to achieve object detection simultaneously. This is in contrast with existing co-detection methods [Bao et al., 2012; Guo et al., 2013; Shi et al., 2013] that implicitly assume a fully-connected graph between the object proposals, but do not provide the means to perform inference in such models. Indeed, we believe that two major factors contribute to the performance improvement: (i) A principled inference method works better than the heuristic search strategies employed by the baselines; (ii) Our multiple kernel similarity is more accurate than simple similarity scores. However, in previous chapter, we only tackle class-specific object co-detection. Note that, the resulting object co-detection algorithm fails to exploit the correlations between multiple classes and, for scalability reasons are limited to modeling object instance similarity with relatively low-dimensional hand-crafted features.

In this chapter, we extend our model to multiclass object co-detection. Also, we study how our framework can be applied to jointly co-detecting different object categories with automatically learned features, thus leveraging the collective power not only of multiple images, but also of multiple classes. Indeed, we address the problem of multiclass object co-detection for large scale datasets. To this end, we formulate co-detection as the joint multiclass labeling of object candidates obtained in a class-independent manner. To exploit the correlations between objects, we build a fully-connected CRF on the candidates, which explicitly incorporates both geometric layout relations across object classes and similarity relations across multiple images. We then introduce a structural boosting algorithm that lets us exploit rich, high-dimensional deep network features to learn object similarity within our fully-connected CRF. Our experiments on PASCAL VOC 2007 and 2012 evidence the ben-

efits of our approach over object detection with region-based convolutional neural networks (RCNN) [Girshick et al., 2014; Girshick, 2015], single-image CRF methods [Desai et al., 2009] and state-of-the-art co-detection algorithms [Bao et al., 2012; Guo et al., 2013; Shi et al., 2013] including the work discussed in the previous chapter. This chapter is based on our work on multi-class joint similarity learning [Hayder et al., 2015].

## 4.1 Object Co-detection: Single Class to Multiclass

Exploring contextual relations is one of the key factors to improve object detection under challenging viewing conditions and to scale up recognition to large numbers of object classes. Object co-detection, which jointly detects object instances in a set of related images, constitutes an important step towards utilizing large-scale context beyond individual images [Bao et al., 2012]. Recent efforts have achieved promising results on challenging detection benchmarks by learning instance similarity within object classes [Hayder et al., 2014; Shi et al., 2013; Guo et al., 2013].

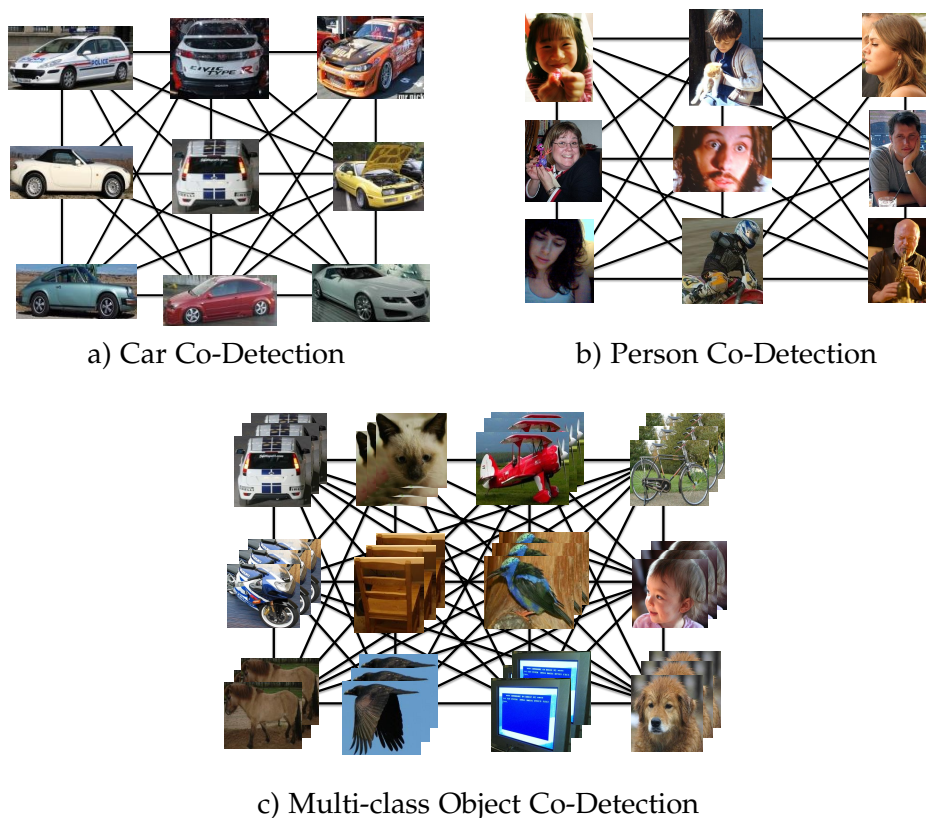


Figure 4.1: **Top:** Conventional single-class object co-detection vs. **Bottom:** our multi-class object co-detection approach.



---

Despite this progress, most existing object co-detection methods focus on single-class object detection and treat the multiclass scenario as a set of unrelated tasks. As such, they are unable to capture the correlations between co-occurring object categories, or exploit their spatial and semantic relations, which leads to suboptimal performances. In addition, due to their approach to learning instance similarities, these techniques are confined to employing relatively low-dimensional hand-crafted object features, such as color or LBP histograms. Unfortunately, such simple feature descriptors lack the necessary representation power to capture rich characteristics and similarities between instances of multiple object categories. In addition, pre-trained object detectors, such as DPMs, have been adopted to propose potential object candidates for joint inference. While this pre-processing step helps reducing the search space, it scales poorly to large number of object classes and limits the final performance according to the quality of the object detector.

## 4.2 Multi-class Objects in Context

Putting objects into context has been widely studied to improve the robustness of detectors. In this section, we present the existing approaches for multi-class object detection which exploits the context to improve the object detection performance. Furthermore we highlight their limitations and discuss how our proposed method can overcome these shortcomings. Existing multi-class object detection approaches which benefit from context falls into two categories:

**Single image techniques:** The pioneering work of [Hoiem et al., 2008; Galleguillos and Belongie, 2010] improve the robustness of detectors by exploiting the co-occurrence of objects and scene properties within an image. In particular, object-object relations have been integrated into several multiclass object detection systems. For example, Desai et al. [Desai et al., 2009] propose to jointly detect multiple object classes by defining a CRF on top of DPMs, in which the relative geometric relationships among 20 classes are captured. Choi et al. [Choi et al., 2012] build a tree-structured model to encode both the co-occurrence statistics and relative spatial locations of multiple object classes. While these methods have shown the benefits of object context, they focus on modeling the context from a single image. Our approach incorporates contextual information from all the images.

**Multi image techniques:** An overview of object co-detection techniques is provided in section 3.2.1. In Chapter 3, we propose to learn a category-level similarity function based on color and LBP histograms. These similarity learning approaches,

however, scale poorly to the dimensionality of the features, and are thus mostly limited to using relatively low-dimensional hand-crafted features. Instead, here, we design a learning framework that lets us make use of a rich representation of objects from different classes from a deep neural network. More importantly, while all existing co-detection methods consider a single object class at a time, we also model the object relations across multiple classes.

Our work is inspired by the fully-connected CRF model and its learning algorithm [Krähenbühl and Koltun, 2011, 2013; Vineet et al., 2012; Zhang and Chen, 2012]. The fully-connected CRF restricts the functional form of the weights in its pairwise potential to a weighted mixture of Gaussian kernels, which allows efficient inference based on fast Gaussian convolution [Baek et al., 2013]. In practice, the efficiency critically depends on the dimensionality of the input feature space and deteriorates quickly with higher dimensional features. Our work develops a structural boosting approach based on functional gradient descent [Munoz et al., 2009; Ratliff et al., 2009], in which we incrementally learn a set of weighted Gaussian kernels defined on low-dimensional feature spaces. In Chapter 3, we also learn a mixture of weighted kernels for a fully connected CRF. However, in that work, learning is treated as a separate regression problem without considering the CRF framework. Furthermore, our previous learning strategy does not scale up to large-scale multiclass object co-detection. Note that, here, our goal is not to build a kernel-based similarity classifier as in [Gönen and Alpaydm, 2011; Vedaldi et al., 2009; Shi et al., 2013], since this would not yield a mixture of Gaussian kernels adapted to our fully-connected CRF framework.

Built on the recent success of deep learning in object recognition [Krizhevsky et al., 2012; Girshick et al., 2014], our approach exploits the output of the RCNN [Girshick et al., 2014] as unary potentials. Note, however, that other context-free object detectors or deep network classifiers [Szegedy et al., 2015; Simonyan and Zisserman, 2015; Ouyang et al., 2017; Shen and Xue, 2014] could also be adopted in our framework, and may further improve the performance. Finally, [Zhu et al., 2015] also proposes to incorporate context information within a deep network based detection framework. However, the resulting method focuses only on exploiting the context around an object hypothesis, and thus does not exploit the collective information contained in a set of images.

#### 4.2.1 Our Idea

As illustrated in Fig. 4.1, we tackle the multiclass object co-detection problem at large scale. To this end, we take a hypothesize-and-classify approach and introduce a joint labeling framework that addresses the limitations of the single-class co-detection.

---

Given a large set of class-independent object candidates, we formulate multiclass object co-detection as the task of assigning each object candidate to either one of the target classes, or background. To exploit the correlation between objects, we explicitly incorporate two types of object relations: geometric layout relations between object classes within an image; and similarity relations between object instances across multiple images. To capture complex relationships between objects, we make use of learned CNN features, and introduce a principled approach to learning similarities with such high-dimensional descriptors.

More specifically, we generate class-independent object candidates based on an objectness measure [Uijlings et al., 2013] that does not require any pre-trained detectors, and build a fully-connected Conditional Random Field (CRF) on these candidates. The unary potentials of this CRF are then obtained via a deep convolutional neural network trained to generate class confidence scores [Girshick et al., 2014]. Furthermore, we make use of two types of edge potentials: one that encodes the class-dependent spatial relations between two object instances in an image [Desai et al., 2009], and one that encourages a similarity-based label smoothness between two object instances across the entire set of candidates.

An important focus of this work is the design of an object similarity measure, and more precisely, the study of how high-dimensional deep learning features, that have proven highly discriminative, can be effectively employed to model the similarity of object instances in our pairwise potential. Indeed, while inference in fully-connected CRFs can be performed efficiently by making use of Gaussian kernels in the pairwise potentials, this strategy scales poorly with the dimensionality of the features used in the kernels. To address this issue, we adopt a structural boosting approach to learning our pairwise potentials. Our learning strategy incrementally adds low-dimensional Gaussian kernels to the CRF model so as to optimize the overall labeling performance. We develop two structural boosting algorithms that encode different measures of such performance: one based on a max-margin criterion with Hamming loss, and one that minimizes the KL-divergence between the marginals of the CRF and the overlap ratios of the object candidates with the ground-truth. The resulting pairwise potential, in the form of a weighted sum of low-dimensional Gaussian kernels, allows us to perform inference efficiently, which is critical for multiclass object co-detection.

Furthermore, we evaluate our method on two large-scale object detection datasets: Pascal VOC 2007 [Everingham et al., 2010] and 2012 [Everingham et al.]. Our experiments demonstrate that our approach outperforms state-of-the-art methods on several standard metrics. This evidences the importance of learning rich similarity measures to account for the contextual relations across object classes and instances.

### 4.3 Large-scale Object Co-detection

Given a set of images, we aim to jointly detect object instances of multiple classes in all the images. To this end, we employ a two-stage strategy: We first generate a set of object hypotheses in each image, and then formulate co-detection as the problem of jointly labeling the hypotheses from all the images as either one of the target object classes or background. To address the labeling task, we build a fully-connected Conditional Random Field (CRF) that captures the spatial relationships of the objects within each image and the object similarity across all images. Our object similarity measure exploits high-dimensional features from which we learn a compact pairwise potential that allows efficient inference with a large number of images and of object hypotheses. We introduce our CRF model in the remainder of this section, and discuss our structural boosting approach to learning its pairwise potential in Section 4.4.

#### 4.3.1 Object Hypotheses Generation

For each image  $I_m$  in a given image set  $\mathcal{I}$ , we generate a set of object hypotheses  $\mathcal{X}^m$  based on a class-independent objectness measure [Endres and Hoiem, 2010; Alexe et al., 2010]. Specifically, we adopt the Selective Search method [Uijlings et al., 2013] to extract  $N_m$  object hypotheses in  $I_m$ , represented by a set of bounding boxes, with a high-recall rate. We then apply a fine-tuned RCNN model [Girshick et al., 2014] to prune down the number of hypotheses based on the SVM score. Finally, following [Girshick et al., 2014], we train a regression model to adjust the position of each remaining bounding box. We denote by  $\mathcal{X} = \cup_m \mathcal{X}^m = \{\mathbf{X}_1, \dots, \mathbf{X}_N\}$  the set of all hypotheses, where  $\mathbf{X}_i$  represents the bounding box parameters of the  $i$ -th object hypothesis, and  $N = \sum_m N_m$ . We then extract an object feature descriptor  $\mathbf{f}_i \in \mathbb{R}^K$  for each bounding box. In particular, we make use of the  $f7$ -layer features of the fine-tuned RCNN mentioned above.

#### 4.3.2 CRF for Multiclass Object Co-detection

Given the set of object hypotheses  $\mathcal{X}$ , our goal is to classify each object candidate into either one of the foreground object classes  $\mathcal{C}$  or background  $\emptyset$ . We follow the CRF formulation in Equation 3.1 and describe each potential function below.

#### 4.3.3 Unary Potential

The unary potentials  $\phi(y_i = c | \mathbf{X}_i)$  encodes the cost of assigning the candidate  $\mathbf{X}_i$  to the object class  $c$ . To this end, we train a CNN model on normalized bounding

boxes with  $|\mathcal{C}| + 1$  categories and take the output of the  $f7$ -layer as feature vectors for all the bounding boxes, and then train a set of linear SVM classifiers for  $|\mathcal{C}| + 1$  classes. The classifier for the background class is trained in an inverted fashion by performing hard-negative mining. In other words, this classifier treats all the ground-truth bounding boxes from the first  $|\mathcal{C}|$  classes as positive examples, and we make use of the negative of its output score as a score for the background class. Viewing the scores of all classifiers as log probabilities, we then define our unary term as

$$\phi_u(y_i|\mathbf{X}_i) = - \sum_{c=1}^{|\mathcal{C}|+1} (s_{ic}) \mathbf{1}_{(y_i=c)} \quad (4.1)$$

where  $s_{ic}$  represents the score of class  $c$  for each sample bounding-box  $i$ .

#### 4.3.4 Pairwise Potential

The pairwise potential  $\psi(\cdot)$  captures the relationship between pairs of object hypotheses, and measures the cost of the different possible label assignments for each pair. We incorporate two types of relationship in the pairwise term: (i) the spatial, or geometric, relationships between different object classes within each image; (ii) the similarity between any two object candidates in the image set. We denote these two pairwise potentials as  $\psi_g(\cdot)$  and  $\psi_s(\cdot)$ , respectively.

For  $\psi_g$ , we follow the definition of spatial relations of [Desai et al., 2009], which groups the relative locations of two object hypotheses into  $D$  canonical relations. The pairwise potential is defined as

$$\psi_g(y_i, y_j|\mathbf{X}_i, \mathbf{X}_j) = \mathbf{w}_{y_i, y_j}^T \mathbf{d}(\mathbf{X}_i, \mathbf{X}_j), \quad (4.2)$$

where  $\mathbf{w}_{y_i, y_j}$  are weights for all possible (i.e.,  $D$ ) geometric configurations of labels  $y_i$  and  $y_j$ . The binary vector  $\mathbf{d}(\mathbf{X}_i, \mathbf{X}_j)$  encodes the geometric relation of  $\mathbf{X}_i$  and  $\mathbf{X}_j$  (i.e., 1 for the correct relation, 0 for the other ones). Note that  $\mathbf{d} = \mathbf{0}$  when  $\mathbf{X}_i$  and  $\mathbf{X}_j$  are from different images.

The pairwise potential  $\psi_s$  is a data-dependent Potts model, encouraging similar hypotheses to share the same object label. We restrict the data-dependent term to take the form of a weighted mixture of Gaussian kernels defined on image features. This can be written as

$$\psi_s(y_i, y_j|\mathbf{X}_i, \mathbf{X}_j) = \sum_{t=1}^T w_t K_t(\mathbf{f}_i, \mathbf{f}_j; \sigma_t) \mu(y_i, y_j), \quad (4.3)$$

where each  $K_t$  is a Gaussian kernel with variance  $\sigma_t^2$  defined on the object features  $\mathbf{f}_i$

and  $\mathbf{f}_j$ , and  $\mu$  is a label compatibility function, which we define as a Potts model, *i.e.*,  $\mu_s(y_i, y_j) = \mathbf{1}_{y_i \neq y_j}$ .<sup>1</sup>

In essence, the mixture of Gaussian kernels encodes the appearance similarity between two object candidates. Such a representation was employed by [Krähenbühl and Koltun, 2011] to design an efficient inference algorithm for semantic labeling in a fully-connected CRF. The computational efficiency of this algorithm, however, critically depends on the dimensionality of the object features  $\mathbf{f}_i, \mathbf{f}_j$ . In this work, we seek to exploit the powerful high-dimensional features extracted by CNNs. Directly employing these features in each kernel would make inference impractically slow. To overcome this issue, we propose to make use of one-dimensional Gaussian kernels of the form

$$K_t(\mathbf{f}_i, \mathbf{f}_j, \sigma_t, \kappa_t) = \exp\left(-\frac{(\mathbf{f}_i^{\kappa_t} - \mathbf{f}_j^{\kappa_t})^2}{\sigma_t^2}\right) \quad (4.4)$$

where  $\mathbf{f}_i^{\kappa_t}$  extracts the  $\kappa_t$ -th dimension of the feature vector. While each kernel can now be efficiently evaluated, with the kind of CNN features that we would like to utilize, considering all feature dimensions would yield several thousands of kernels. This would therefore still make inference intractable. We address this issue in Section 4.4, where we introduce an approach to learning the kernels that are important for our task.

#### 4.3.5 Efficient Inference for Co-detection

Efficient inference in our fully-connected CRF remains challenging since each image contains a large number of object candidates and the spatial pairwise potential  $\psi_g$  does not have a form that allows us to use the efficient algorithm of [Krähenbühl and Koltun, 2011]. We therefore design a two-step cascaded inference procedure to obtain the marginal posterior probabilities of each object candidate.

We first start with a model that does not contain the potential  $\psi_s$  defined on the dense connections. The resulting model thus decomposes into  $M$  individual CRFs, one for each image. We make use of the max-margin procedure of [Desai et al., 2009] to learn the parameters of this model (*i.e.*, a single set of parameters that will be used for all  $M$  images). At inference, we apply the greedy forward search algorithm of [Desai et al., 2009] and compute the marginals of each object node, which we denote by  $\phi_g(y_i | \mathbf{X}_i)$ .

We then integrate the marginals from the first step with our fully-connected pair-

---

<sup>1</sup>General compatibility functions can also be used in our model.

wise potential  $\psi_s$ , which yields a CRF with energy function

$$\tilde{E}(\mathcal{Y}, \mathcal{X}) = \sum_{i=1}^N \phi_g(y_i | \mathbf{X}_i) + \sum_{i=1}^N \sum_{j>i} \psi_s(y_i, y_j | \mathbf{X}_i, \mathbf{X}_j). \quad (4.5)$$

The marginals of each object candidate can then be obtained efficiently using the approximate mean-field inference method of [Krähenbühl and Koltun, 2011], which relies on fast Gaussian filtering. As mentioned above, however, inference is efficient only as long as the number of Gaussian kernels remains relatively small. This problem will be addressed in the next section.

## 4.4 Kernel Learning for Fully-connected CRFs

In this section, we introduce an approach to learning the kernels that define the pairwise potential of our fully-connected CRF. We also learn a transformation of unary scores because unary scores are not scaled across different classes. As briefly discussed in Section 4.3, the efficiency of our inference strategy depends on the compactness of our pairwise potential  $\psi_s$ , or, more precisely, since we employ one-dimensional Gaussian kernels, on the number of kernel functions in  $\psi_s$ . Unfortunately, the feature descriptors that have proven the most discriminative in practice, such as CNN features, are typically very high dimensional. Employing one kernel per feature would then not be practical. Here, we propose to overcome this issue by learning the kernels that are relevant to our goals. To this end, we introduce a structural boosting approach that allows us to select a small subset of distinctive object features and learn their corresponding kernel functions. In the remainder of this section, we first discuss our structural boosting framework and then describe the two different loss functions to learn our pairwise potentials.

### 4.4.1 Structural Boosting for Fully-connected CRFs

We now present our structural boosting framework. This framework follows the general functional gradient descent approach introduced in [Munoz et al., 2009]. In this general context, however, we design two novel algorithms specialized to the problem of kernel learning in fully-connected CRFs, with emphasis on the task of multi-class object co-detection.

Let  $(\hat{\mathcal{X}}, \hat{\mathcal{Y}})$  be a training set containing pairs of object bounding boxes  $\mathbf{X}_i$  with corresponding ground-truth label  $\hat{y}_i$ . Furthermore, let  $R[\psi_s | \hat{\mathcal{X}}, \hat{\mathcal{Y}}]$  denote an empirical loss function defined with respect to the pairwise potential  $\psi_s$  and the training examples. Our approach exploits the fact that, as shown in Eq. 4.3, our pairwise

potential  $\psi_s$  has an additive form,

$$\psi_s(\cdot) \sim \sum_t w_t h_t(\cdot) \quad (4.6)$$

Each  $h_t \in \mathcal{H}$  can thus be thought of as a weak learner belonging to a family  $\mathcal{H}$ . In our case,  $\mathcal{H}$  is defined as a family of pairwise potential functions indexed by  $(\sigma, \kappa)$ , such that each weak learner has the form

$$h(\cdot; \sigma, \kappa) = \sum_{i=1}^N \sum_{j>i} K(\mathbf{f}_i, \mathbf{f}_j, \sigma, \kappa) \mu(y_i, y_j), \quad (4.7)$$

where  $K(\cdot)$  is a one dimensional kernel as shown in Eq. 4.4.

Our structural boosting algorithm then works as follows. At each step  $t$ , we first compute the negative functional gradient  $-\nabla_f R[\psi_s | \hat{\mathcal{X}}, \hat{\mathcal{Y}}]$  with respect to  $\psi_s$  and evaluate it at the previous pairwise potential estimate  $\psi_s^{t-1}$ , and find its maximum projection onto the weak learner space  $\mathcal{H}$ . This can be expressed as computing

$$h_t^* = \operatorname{argmax}_{h \in \mathcal{H}} \langle -\nabla_f R[\psi_s | \hat{\mathcal{X}}, \hat{\mathcal{Y}}] |_{\psi_s^{t-1}}, h \rangle. \quad (4.8)$$

Given the best weak learner  $h_t^*$ , we then use line-search to determine the corresponding weight as

$$w_t^* = \operatorname{argmin}_{w_t \in \mathbb{R}} R[\psi_s^{t-1} + w_t h_t^* | \hat{\mathcal{X}}, \hat{\mathcal{Y}}]. \quad (4.9)$$

To maximize the projection of the functional gradient in Eq. 4.8, we discretize the parameter space of Gaussian kernel widths and enumerate all combinations of discrete width and feature dimension in  $\mathbf{f}$ . The resulting optimal kernel at step  $t$  is denoted by  $K_t(\cdot, \cdot, \sigma_t, \kappa_t)$ . Due to the approximate nature of our inference procedure, the functional gradient is also approximate for the boosting process. We stop the structural boosting process when the change in accuracy on a validation set is less than a certain threshold between the current boosting iteration and previous boosting iteration.

Below, we introduce two special cases of the general algorithm described above: One based on a max-margin learning approach, and one based on a direct loss minimization strategy. In each case, we first discuss the corresponding empirical loss functions  $R$ . We will derive the functional gradient projection for max-margin learning, propose a numerical approximation for the direct loss minimization. The steps of our approach are outlined in Algorithm 4.1.

The time complexity of the learning algorithm is  $O(KSNT^2)$ , where  $N$  is the total number of candidate boxes in the pool, and  $T, K, S$  are defined in Algorithm 4.1. The space complexity is  $O(NT)$ . At test time, the time complexity and the space complexity are  $O(NT)$ .



### 4.4.2 Max-margin Structural Boosting

Let us first consider the case of max-margin learning. In this scenario, the empirical loss  $R$  can be expressed as

$$R[\psi_s | \hat{\mathcal{X}}, \hat{\mathcal{Y}}] = \max_{\mathcal{Y}} (-\tilde{E}(\hat{\mathcal{X}}, \mathcal{Y}) + \mathcal{L}(\mathcal{Y}, \hat{\mathcal{Y}})) + \tilde{E}(\hat{\mathcal{X}}, \hat{\mathcal{Y}}), \quad (4.10)$$

where  $\tilde{E}(\hat{\mathcal{X}}, \mathcal{Y})$  is the energy function defined in Eq. 4.5, and  $\mathcal{L}(\mathcal{Y}, \hat{\mathcal{Y}})$  is the Hamming loss between a label configuration  $\mathcal{Y}$  and the ground-truth  $\hat{\mathcal{Y}}$ ,

$$\mathcal{L}(\mathcal{Y}, \hat{\mathcal{Y}}) = \sum_{i=1}^N \mathbf{1}_{(\hat{y}_i \neq y_i)} \quad (4.11)$$

The negative functional (sub-)gradient of this loss function with respect to  $\psi_s$  can be written as

$$\begin{aligned} -\nabla_f R[\psi_s | \hat{\mathcal{X}}, \hat{\mathcal{Y}}] &= \nabla_f E(\hat{\mathcal{X}}, \mathcal{Y}^*) - \nabla_f E(\hat{\mathcal{X}}, \hat{\mathcal{Y}}) \\ &= \delta_{\mathcal{Y}=\mathcal{Y}^*} - \delta_{\mathcal{Y}=\hat{\mathcal{Y}}} \end{aligned} \quad (4.12)$$

where  $\delta$  is the Kronecker delta function, and  $\mathcal{Y}^*$  is the label configuration that determines the value of the first term in Eq. 4.10. The negative functional gradient is positive at locations where they are assigned their ground truth labels and negative where they are assigned their inferred labels. These impulses will cancel each other when ground truth and inferred labels agree.

In other words,

$$\mathcal{Y}^* = \operatorname{argmax}_{\mathcal{Y}} \{-E(\hat{\mathcal{X}}, \mathcal{Y}) + \mathcal{L}(\mathcal{Y}, \hat{\mathcal{Y}})\}, \quad (4.13)$$

and can thus be estimated approximately using loss augmented inference in our CRF. We use the same mean field algorithm with loss augmented unary terms to find the marginal posterior probability and labels  $\mathcal{Y}^*$ .

The optimal weak learner can then be computed by maximizing the projection of the functional gradient as in Eq. 4.8, which can be written as

$$\begin{aligned} h_t^* &= \operatorname{argmax}_{h \in \mathcal{H}} \langle -\nabla_f R[\psi_s | \hat{\mathcal{X}}, \hat{\mathcal{Y}}], h \rangle \\ &= \operatorname{argmax}_{\sigma, \kappa} \sum_{i=1}^N \sum_{j>i} K(\mathbf{f}_i, \mathbf{f}_j, \sigma, \kappa) \left( \mu(y_i^*, y_j^*) - \mu(\hat{y}_i, \hat{y}_j) \right). \end{aligned} \quad (4.14)$$

In each boosting step, we compute the projection using the same fast Gaussian con-

volution as for inference, and enumerate all the  $(\sigma, \kappa)$  pairs. Algorithm 4.1 outlines the steps to learn these structural kernels.

#### 4.4.2.1 Space-Time Complexity

Our inference algorithm is a mean-field approximate method, and therefore the learning algorithm is also approximate. The time complexity of the learning algorithm (which is dominated by Algorithm 4.1) is  $O(K * S * N * T^2)$ , where  $N$  is the total number of candidate boxes in the pool, and  $T, K, S$  are defined in Algorithm 4.1. The space complexity is  $O(N * T)$  [Krähenbühl and Koltun, 2013]. At test time, the time complexity is  $O(N * T)$  and the space complexity is  $O(N * T)$ .

---

#### Algorithm 4.1 Structural Kernel Learning Algorithm

---

**Input:**

- 1: Training data:  $\mathbf{D} = (\mathcal{X}, \hat{\mathcal{Y}})$
- 2: Unary potentials:  $\phi_g$
- 3: CNN FC7 feature dimension:  $\mathbf{K}$
- 4: No. of Iterations:  $\mathbf{T}$
- 5: Loss Function:  $\mathcal{L}$
- 6:  $\mathbf{f} = \sigma = \mathbf{w} = \emptyset$

**Iteration:**

- 8: **for**  $t = 1 \dots \mathbf{T}$  **do**
- 9:   Build Dense CRF Graph using Unary  $\phi_g$  and  $(t - 1)$  Structural Gaussian kernels  $\mathbf{f}$
- 10:   **for**  $k = 1 \dots \mathbf{K}$  **do**
- 11:     Select a range of  $\mathbf{S}$   $\sigma$  values
- 12:     **for**  $s = 1 \dots \mathbf{S}$  **do**
- 13:       Generate weak learner  $h_t$  using feature index  $k$  and variance  $\sigma^s$
- 14:       Efficient Mean-field inference
- 15:       Compute Loss using Eq. 4.10 or Eq. 4.15
- 16:     **end for**
- 17:   **end for**
- 18:   Select best structural kernel  $h_t^*$  using variance  $\sigma_t^*$  and feature index  $f_t^*$
- 19:    $\sigma = [\sigma \cup \sigma_t^*]$
- 20:    $\mathbf{f} = [\mathbf{f} \cup f_t^*]$
- 21:   Line Search for kernel weight  $\mathbf{w}^*$ ,  $\mathbf{w} = [\mathbf{w} \cup \mathbf{w}^*]$
- 22:    $\psi_s \leftarrow [\psi_s \cup h_t]$
- 23: **end for**

**Output:**

- 24: Learned boosted structural kernel features:  $\mathbf{f}$
  - 25: Kernel widths:  $\sigma$
  - 26: Kernel weights:  $\mathbf{w}$
  - 27: Pairwise potentials:  $\psi_s$
-

### 4.4.3 Direct-loss Structural Boosting

Typically, detection algorithms are evaluated using the mean average-precision error, and thus do not only care about the predicted label for each object, but rather about some notion of score obtained for each class. Therefore, the max-margin framework described above might not be optimizing the best loss. To overcome this issue, we introduce a direct-loss minimization approach.

More precisely, instead of predicting a unique class label for each object hypothesis, we aim to estimate the overlap between the proposed object bounding box and the ground-truth annotations of each class. In the following, we refer to this overlap as *the target overlap distribution*. Here, we propose to directly minimize the KL divergence between the target overlap distribution and the marginal probability from our fully-connected CRF. Let us denote the target overlap distribution of object hypothesis  $\mathbf{X}_i$  as  $\mathbf{p}_i$  and the corresponding marginal output of the CRF as  $\mathbf{q}_i$ . We then define the empirical loss  $R$  as

$$R[\psi_s | \hat{\mathcal{X}}, \hat{\mathcal{Y}}] = \sum_{i=1}^N \sum_{k=1}^{|\mathcal{C}|+1} \mathbf{p}_i(k) \log(\mathbf{p}_i(k) / \mathbf{q}_i(k)) \quad (4.15)$$

where  $\mathbf{q}_i(k)$  is a function of  $\psi_s$ , which is recursively defined by the mean field update equation

$$\mathbf{q}_i(k) \propto \exp \left( -\phi_g(k | \mathbf{X}_i) - \sum_{j \neq i} \sum_{y_j} \psi_s(k, y_j | \mathbf{X}_i, \mathbf{X}_j) \mathbf{q}_j(y_j) \right). \quad (4.16)$$

Computing the functional gradient and its maximum projection is expensive due to the recursion in Eq. 4.16. We therefore follow a finite difference approach to approximately estimate the functional gradient. At each step  $t$ , this translates to searching for the weak learner that maximizes the decrease in the empirical loss according to the finite difference gradient, which can be expressed as

$$h_t^* = \operatorname{argmax}_{h \in \mathcal{H}} R[\psi_s^{t-1} | \hat{\mathcal{X}}, \hat{\mathcal{Y}}] - R[\psi_s^{t-1} + \epsilon h | \hat{\mathcal{X}}, \hat{\mathcal{Y}}], \quad (4.17)$$

where  $\epsilon$  is a small constant value. As in the max-margin case, we enumerate all possible  $h$ . Since inference can be performed efficiently, this search remains tractable.

## 4.5 Experiments

We now demonstrate the effectiveness of our method on large scale multiclass object co-detection. To this end, we evaluate our approach on two challenging large datasets with multiple object classes, i.e., PASCAL VOC 2007 [Everingham et al., 2010] and

2012 [Everingham et al.], and compare our results with those of the state-of-the-art methods.

#### 4.5.1 Datasets and Setup

As discussed earlier, the PASCAL VOC dataset [Everingham et al., 2010] contains 20 object classes, and, we use the standard trainval/test partitions for all these experiments. Our training procedure consists of four stages as follows,

1. Fine-tuning the parameters of a deep network.
2. Training class-specific SVM classifiers [Girshick et al., 2014].
3. Learning the per-image layout CRF [Desai et al., 2009].
4. Learning the similarity kernels via structural boosting in our fully-connected CRF.

We split the trainval images into a training and a validation set. The training set consists of randomly selected 75% of the trainval images, and the remaining 25% of the trainval images are kept as validation set. We use the training set to fine-tune a pre-trained deep network (AlexNet [Krizhevsky et al., 2012]) and train the 21 class-specific SVMs as in [Girshick et al., 2014]. We learn the layout CRF and the kernels in our fully-connected CRF based on the validation set. The unary and feature computation jointly take 9 sec per image on average and the learning algorithm takes approximately 72 hrs to train the full model.

At test time, we perform inference in our fully-connected CRF, which not only computes the marginal posterior probability but also generate the maximum-a-posteriori labeling for each object bounding box hypothesis. To evaluate per-class performance, we divide our bounding-box pool into different categories according to the MAP labeling and use the marginal probability as the detection scores to generate the precision-recall curves. The PASCAL VOC 2007 and 2012 datasets comprise 4952 and 10991 test images, respectively. In both cases, we jointly perform detection in all the test images via inference in our fully-connected CRF. Performing inference in our fully-connected CRF containing a total of 995937 nodes (VOC 2007) took 9.8052sec.

We compare our approach (CoDeT-G-LA: kernel selection using max-margin learning, and CoDeT-G-DL: kernel selection using direct loss minimization) with the state-of-the-art context-free detector RCNN [Girshick et al., 2014] (R-CNN and R-CNN-BB), the layout CRF [Desai et al., 2009] with RCNN as unary term (CoDeT-G) and the state-of-the-art single class object co-detection method described in Chapter 3. For this last baseline, and to have a fair comparison, we used CNN features to

generate the unary scores for each individual object detector. However, the approach of [Hayder et al., 2014] to learning object similarity does not scale to CNN features and to large training sets. Therefore, following our formulation in Chapter 3, we employed color and local binary pattern to define the pairwise potentials, and sub-sampled the training set to learn the similarity. The weight of the pairwise potential was learned using the validation set.

Empirically, the unary and R-CNN [Girshick et al., 2014] feature computation jointly take 9sec/image on average. The learning algorithm takes approximately 72 hrs to learn the geometric model and the structural kernels. At test time, given pre-computed unary terms, the geometric model takes 0.06 sec/image, and performing 5 iterations of inference in our fully-connected CRF containing a total of 995937 nodes (VOC 2007) take 9.8052 sec.

VOC 2007 (test)	MLRR	MCOL	R-CNN	R-CNN BB	Co-detection	CoDet-G (Ours)	Codet-G-LA (Ours)	CoDet-G-DL (Ours)
aero	34.1	28.8	64.2	68.1	65.4	70.1	70.2	<b>70.6</b>
bike	53.0	56.2	69.7	72.8	71.6	72.4	74.8	<b>76.3</b>
bird	12.4	3.2	50.0	56.8	54.2	58.8	60.7	<b>61.7</b>
boat	18.9	14.2	41.9	43.0	41.9	43.5	43.6	<b>44.7</b>
bottle	31.2	29.4	32.0	36.8	34.9	38.7	45.0	<b>45.6</b>
bus	43.2	38.7	62.6	66.3	66.7	67.2	67.0	<b>67.6</b>
car	52.7	48.7	71.0	74.2	74.4	74.8	74.9	<b>75.0</b>
cat	21.6	12.4	60.7	67.6	67.5	<b>68.3</b>	64.1	64.9
chair	22.8	16.0	32.7	34.4	35.1	34.1	34.2	<b>34.6</b>
cow	25.0	17.7	58.5	63.5	64.1	65.4	65.8	<b>66.3</b>
table	32.2	24.0	46.5	54.5	53.9	56.8	<b>58.7</b>	58.3
dog	10.6	11.7	56.1	61.2	61.4	63.4	62.1	<b>63.7</b>
horse	51.7	45.0	60.6	69.1	69.3	70.3	72.0	<b>72.3</b>
mbike	41.0	39.4	66.8	68.6	68.7	68.9	70.4	<b>70.4</b>
person	38.6	35.5	54.2	58.7	59.6	58.9	60.1	<b>60.4</b>
plant	19.2	15.2	31.5	33.4	33.2	33.2	35.4	<b>35.6</b>
sheep	27.3	16.1	52.8	62.9	62.5	63.9	<b>65.5</b>	65.4
sofa	32.5	20.1	48.9	51.1	50.8	<b>51.2</b>	47.5	48.6
train	41.3	34.2	57.9	62.5	61.0	<b>64.2</b>	61.1	63.8
tv	41.9	35.4	64.7	64.8	63.0	<b>65.2</b>	64.0	64.4
<b>mAP</b>	<b>32.5</b>	<b>27.1</b>	<b>54.2</b>	<b>58.5</b>	<b>58.0</b>	<b>59.5</b>	<b>59.9</b>	<b>60.5</b>

Table 4.1: **Detection average precision(%) on the PASCAL VOC 2007 test set using AlexNet.** Columns 1-2 shows the co-detection baselines i.e. MLRR [Guo et al., 2013] and MCOL [Desai et al., 2009]. Columns 3-4 provide the baseline state-of-the-art results for detection [Girshick et al., 2014]. R-CNN (without bounding-box regression); R-CNN BB (with bounding-box regression). Column 5 provides results of Chapter 3 co-detection method with deep network unary potentials. Columns 6-8 show co-detection performance. CoDet-G (R-CNN-BB with geometric context model learning); CoDet-G-LA (Kernel selection using max-margin learning); CoDet-G-DL (Kernel selection using direct loss minimization).

### 4.5.2 Results on VOC 2007

We report our results on the VOC 2007 test set using two different evaluation protocols: the standard per-class metric and the multiclass metric of [Desai et al., 2009].

#### 4.5.2.1 Per-class Scores

We follow the VOC performance evaluation protocol and report the Average Precision (AP) and mean of AP (mAP) for the 20 object classes in Table 4.1 using AlexNet [Krizhevsky et al., 2012] and in Table 4.2 using VGG16 [Simonyan and Zisserman, 2015].

VOC 2007 (test)	Fast R-CNN	CoDet-G-DL (Ours)
aero	74.5	75.8
bike	78.3	78.1
bird	69.2	69.3
boat	53.2	53.8
bottle	36.6	36.9
bus	77.3	77.5
car	78.2	79.0
cat	82.0	82.5
chair	40.7	40.1
cow	72.7	73.5
table	67.9	67.7
dog	79.6	81.4
horse	79.2	82.2
mbike	73.0	75.4
person	69.0	70.0
plant	30.1	33.4
sheep	65.4	65.4
sofa	70.2	70.0
train	75.8	74.3
tv	65.8	67.2
mAP	66.9	67.7

Table 4.2: **Detection average precision(%) on the PASCAL VOC 2007 test set using VGG16.** Column 1 shows the results of the Fast-RCNN baseline and Column 2 shows our method results using VGG16.

We can see that CoDet-G outperforms the R-CNN-BB results on 18 out of 20 classes, which suggests that learning a geometric model to incorporate inter-class objects configuration is vital to object detection when we have multiple objects appearing in the same image. By contrast, we observed that the results of the Chapter 3 yields virtually no improvement over the R-CNN-BB results. We conjecture that this is due to the limited scalability of this method, which forced us to subsample the training data when learning the object similarity and to employ less informative, but lower-dimensional, features. Our CoDet-G-LA algorithm yields a further

small improvement over CoDet-G on 14 classes, thus outperforming R-CNN-BB by 1.4%. Our CoDet-G-DL algorithm yields an improvement of 2% over the R-CNN BB baseline results. Another parallel work, Feature Edit method [Shen and Xue, 2014], outperformed the R-CNN BB baseline by a margin of 1.6%. We also evaluated our method using the VGG16 [Simonyan and Zisserman, 2015] network. We obtained an improvement of 0.8% over the Fast-RCNN VGG16 baseline [Girshick, 2015]. In other words, our CoDet-G-DL algorithm achieves state-of-the-art results on VOC 2007 (based on the same kind of deep network structure).

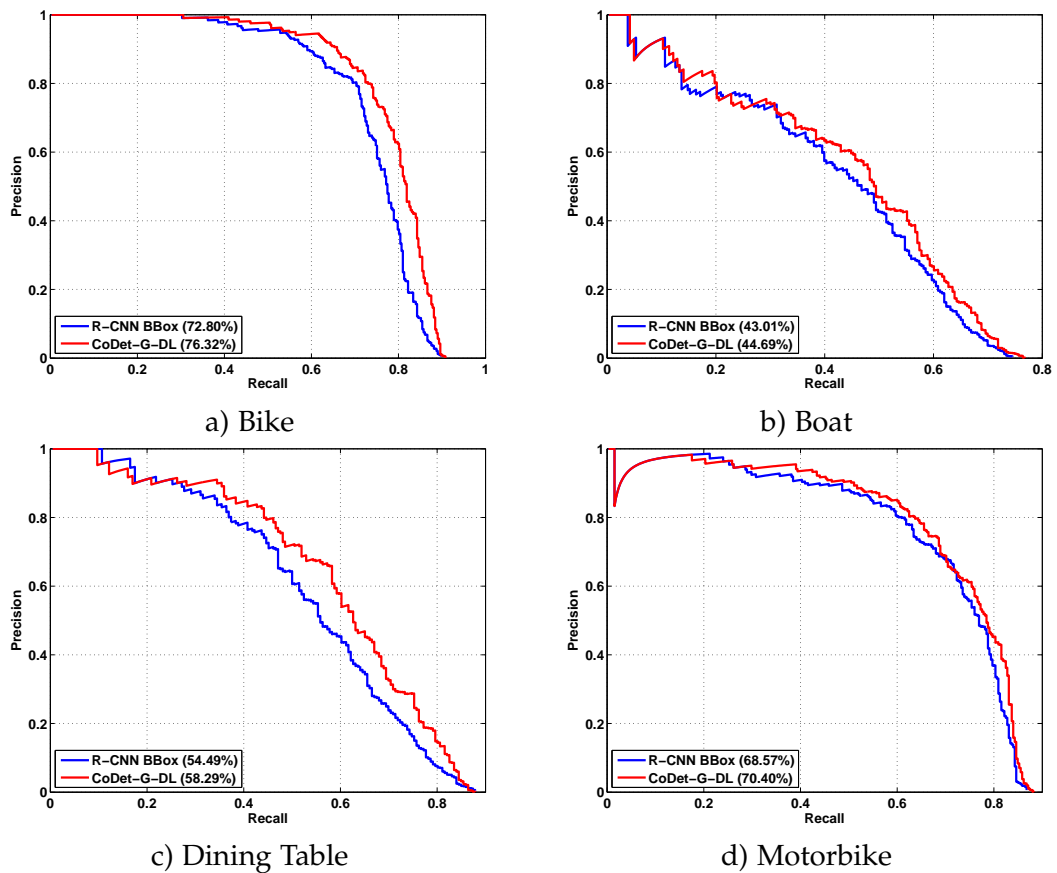


Figure 4.2: **Precision/Recall curves:** Performance comparison on four representative categories using the VOC 2007 dataset. Blue curve represents the R-CNN (without bounding-box regression) [Girshick et al., 2014], whereas, Red curve shows our method results.

We also computed Precision-Recall (PR) curves for comparison with the R-CNN-BB baseline. Fig. 4.2 shows the precision-recall curves for bicycle, boat, dining table and motorbike. These curves clearly indicate that our method improves over R-CNN-BB in the high-recall low-precision region. Following [Hoiem et al., 2012], we provide a detailed comparison of different performance criteria in Fig. 4.3. Our

method outperforms the R-CNN-BB baseline by 2.7% in average normalized precision  $AP_N$ . Finally, in Fig. 4.4, we also provide an analysis of the false-positives using the vehicles category group. False positives with confusion across similar object categories and different object categories are significantly reduced by our approach. This shows the strength of our multiclass object co-detection approach, which benefits from intra-class and inter-class similarity to help reduce the false positives.

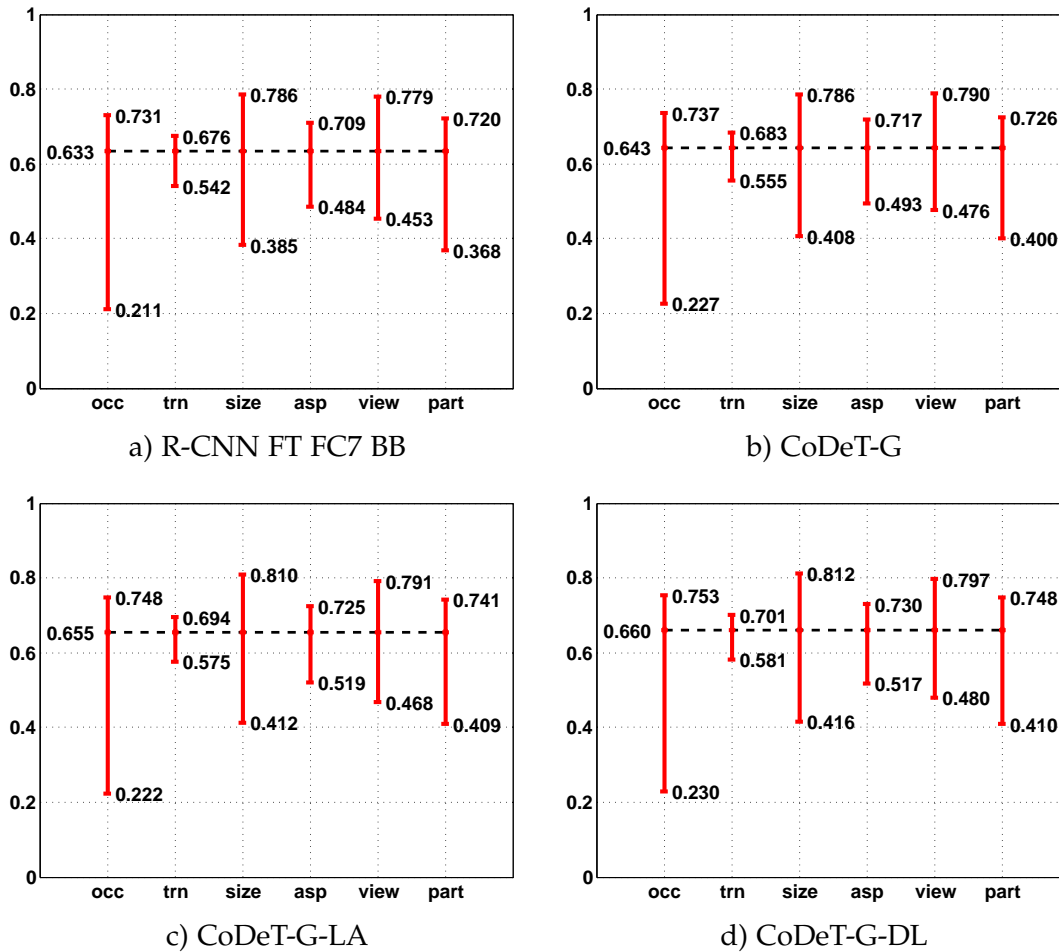


Figure 4.3: **Sensitivity and Impact Analysis:** Overall detailed performance comparison using different metrics (i.e., occlusion (acc), truncated (trn), size, aspect ratio (asp), view point and part visibility). The black dashed line indicates the overall average normalized precision  $AP_N$ .

#### 4.5.2.2 Multi-class Scores

Multi-class object detection performance is difficult to measure using per-class AP. Since our main goal is multi-class object co-detection, we also report the results ac-



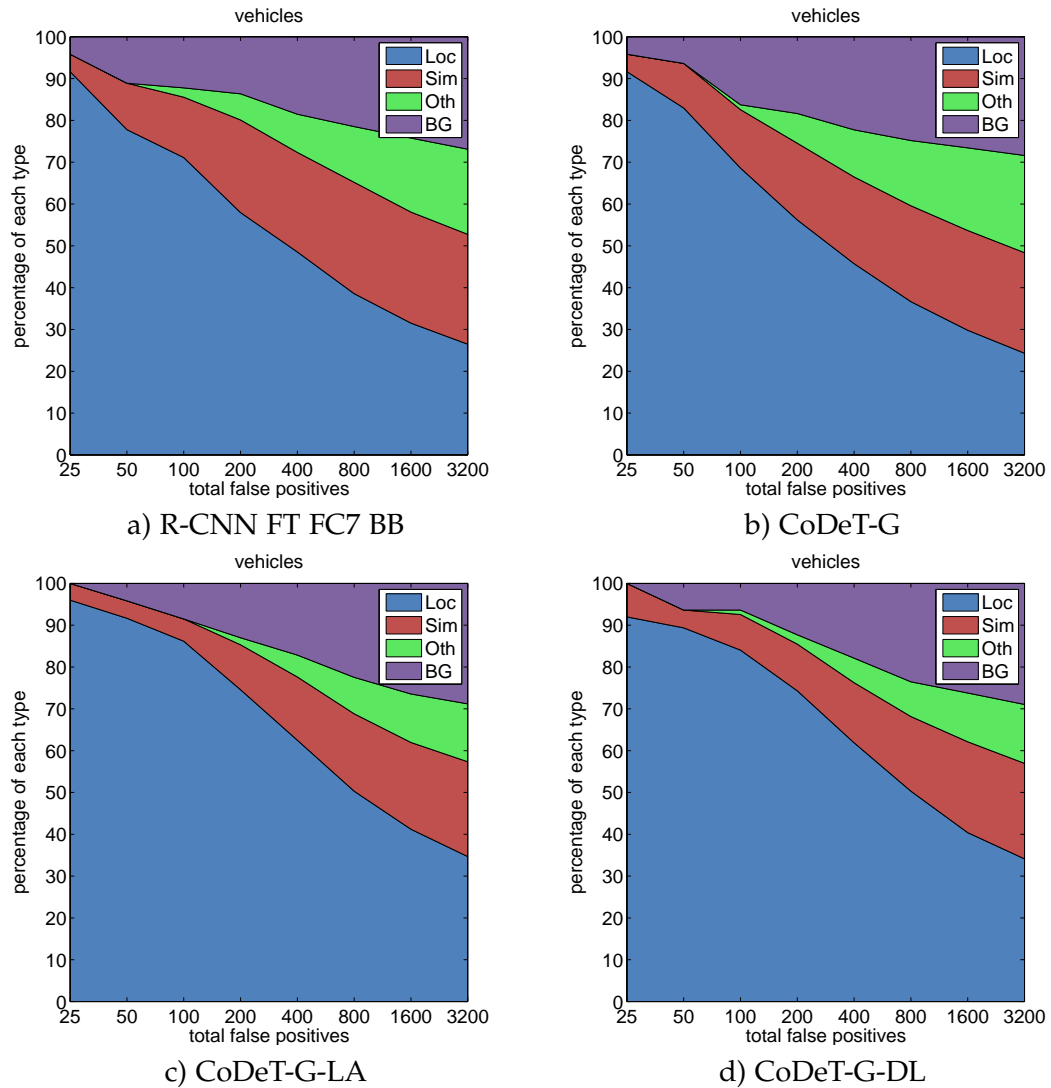


Figure 4.4: False positive analysis using all the vehicles category.

VOC 2007 (test)	Overall AP	Max-A-Posteriori Acc.
R-CNN BB	61.46	57.05
CoDet-G-LA	62.13	62.94
CoDet-G-DL	62.60	64.49

Table 4.3: **Multi-class average precision(%) on the PASCAL VOC 2007 test set.** CoDet-G-DL (Kernel selection using direct loss minimization). We constructed the baseline curve for R-CNN-BB (with bounding-box regression) [Girshick et al., 2014] by pooling the detections across all object classes and images when computing the PR curves. Our model clearly provides a noticeable boost in overall performance.

according to the multiclass detection metric of [Desai et al., 2009]. Following [Desai et al., 2009], we pool the detections across all the classes and all the images and generate a single precision-recall curve from which we compute the overall AP. We also compute the labeling accuracy based on the Maximum-A-Posteriori (MAP) estimation. The results are summarized in Table 4.3. We can see that our method achieves a large improvement over the baseline.

### 4.5.3 Results on VOC 2012

In Table 4.4, we report our results on the VOC 2012 test dataset using the standard per-class metric based on the summary statistics from the evaluation server. Precision-recall curves and multi-class performance metrics cannot be generated here, since we do not have access to the original labeling of the VOC 2012 test dataset.

VOC 2012 (test)	R-CNN	R-CNN BB	CoDet-G	CoDet-G-DL (Ours)
<b>aero</b>	68.1	71.8	<b>72.5</b>	69.8
<b>bike</b>	63.8	65.8	64.2	<b>67.5</b>
<b>bird</b>	46.1	52.0	51.4	<b>52.3</b>
<b>boat</b>	29.4	34.1	<b>34.8</b>	34.4
<b>bottle</b>	27.9	32.6	32.6	<b>35.7</b>
<b>bus</b>	56.6	59.6	62.5	<b>63.0</b>
<b>car</b>	57.0	60.0	60.3	<b>63.4</b>
<b>cat</b>	65.9	69.8	<b>71.2</b>	68.3
<b>chair</b>	26.5	27.6	27.8	<b>28.3</b>
<b>cow</b>	48.7	52.0	52.4	<b>54.0</b>
<b>table</b>	39.5	41.7	<b>42.4</b>	41.1
<b>dog</b>	66.2	69.6	<b>69.7</b>	67.7
<b>horse</b>	57.3	61.3	60.5	<b>65.2</b>
<b>mbike</b>	65.4	68.3	67.7	<b>70.7</b>
<b>person</b>	53.2	57.8	58.6	<b>60.6</b>
<b>plant</b>	26.2	29.6	30.0	<b>32.5</b>
<b>sheep</b>	54.5	57.8	57.7	<b>60.0</b>
<b>sofa</b>	38.1	40.9	40.0	<b>41.1</b>
<b>train</b>	50.6	59.3	<b>61.0</b>	54.7
<b>tv</b>	51.6	54.1	55.6	<b>57.9</b>
<b>mAP</b>	49.6	53.3	53.6	<b>54.4</b>

Table 4.4: **Detection average precision(%) on the PASCAL VOC 2012 test set.** Columns 1-2 provide the baseline state-of-the-art detection results! [Girshick et al., 2014]. R-CNN (without bounding-box regression); R-CNN BB (with bounding-box regression). Columns 3-4 show our co-detection performance. CoDet-G (R-CNN-BB with geometric context model learning); CoDet-G-DL (Kernel selection using direct loss minimization).

Our CoDeT-G-DL algorithm outperforms the R-CNN-BB baseline in 15 object categories, which, altogether, yields 1.1% overall gain in the mean average precision on VOC 2012 test dataset. This indicates that our approach achieves a consistent improvement over the baseline method. Note that, to obtain our results on VOC 2012, we simply re-used the kernels learned using the VOC 2007 trainval dataset.

## 4.6 Qualitative Results on PASCAL VOC

In this section, we analyze the detections produced by our method in more details. In particular, following [Hoiem et al., 2012], Fig. 4.5 depicts the detection trends of our CoDeT-G-DL algorithm for several categories as the number of detections increases. In Fig. 4.6, we compare the changes in scores between R-CNN-BB [Girshick et al., 2014] and our approach, i.e., before and after modeling the geometric relationships and instance similarity, for a few representative examples. Finally, in Fig. 4.7 and Fig. 4.8, we show the top-scoring correct detections and the top-scoring false positives obtained with our CoDeT-G-DL algorithm, respectively.

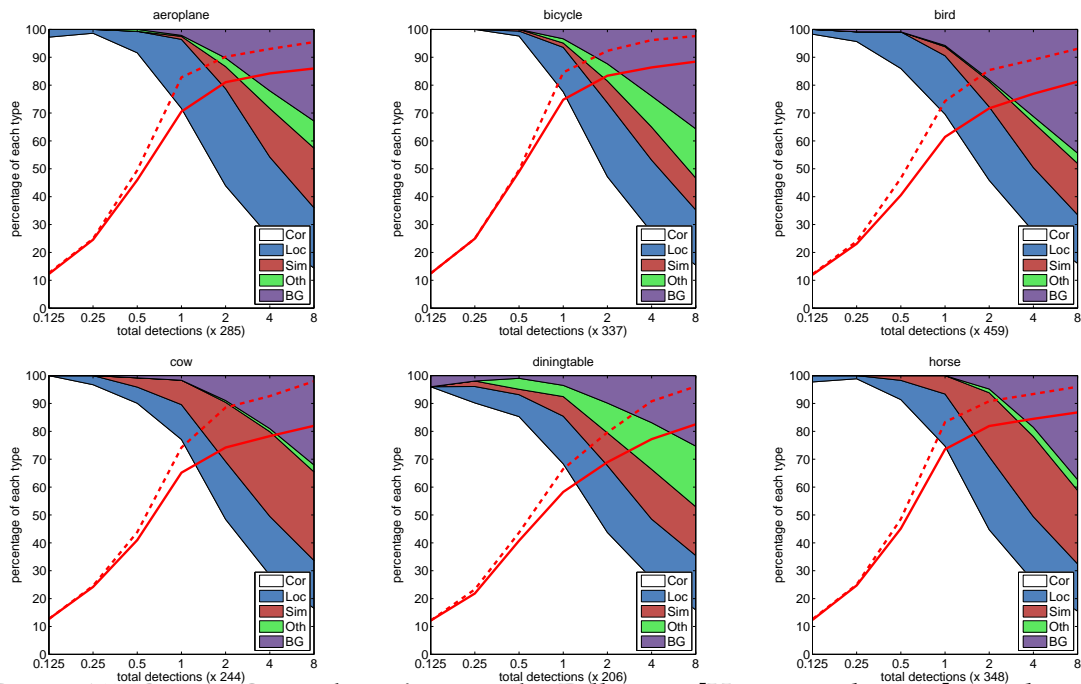


Figure 4.5: **CoDeT-G-DL detection trends.** Following [Hoiem et al., 2012], we show the evaluation of the type of detection as the number of detections increases; The white areas correspond to the correct detections; The blue areas represent the detections with localization error; The red areas correspond to confusion with a similar category; The green areas represent the confusion with a dissimilar category. Finally, the line plots show the recall as a function of the number of objects (dashed=weak localization, solid=strong localization).






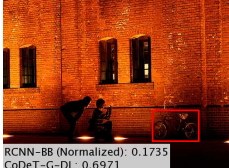



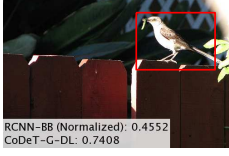
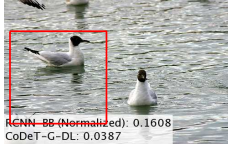
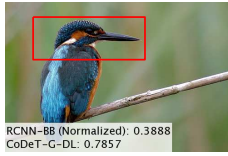


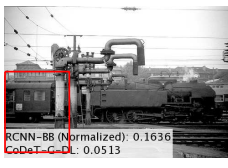





Outperformed	Outperformed	Outperformed	Underperformed
Score Increased True Positive	Score Increased True Positive	Score Decreased True Negative	False Positive
 RCNN-BB (Normalized): 0.6355 CoDeT-G-DL: 0.9784	 RCNN-BB (Normalized): 0.2510 CoDeT-G-DL: 0.5754	 RCNN-BB (Normalized): 0.5364 CoDeT-G-DL: 0.3830	 RCNN-BB (Normalized): 0.3934 CoDeT-G-DL: 0.8019
 RCNN-BB (Normalized): 0.4122 CoDeT-G-DL: 0.9802	 RCNN-BB (Normalized): 0.1735 CoDeT-G-DL: 0.6971	 RCNN-BB (Normalized): 0.3564 CoDeT-G-DL: 0.2972	 RCNN-BB (Normalized): 0.4557 CoDeT-G-DL: 0.6757
 RCNN-BB (Normalized): 0.3131 CoDeT-G-DL: 0.7384	 RCNN-BB (Normalized): 0.4552 CoDeT-G-DL: 0.7408	 RCNN-BB (Normalized): 0.1608 CoDeT-G-DL: 0.0387	 RCNN-BB (Normalized): 0.3888 CoDeT-G-DL: 0.7857
 RCNN-BB (Normalized): 0.4006 CoDeT-G-DL: 0.9741	 RCNN-BB (Normalized): 0.4266 CoDeT-G-DL: 0.8827	 RCNN-BB (Normalized): 0.1636 CoDeT-G-DL: 0.0513	 RCNN-BB (Normalized): 0.1563 CoDeT-G-DL: 0.6030
 RCNN-BB (Normalized): 0.3931 CoDeT-G-DL: 0.9089	 RCNN-BB (Normalized): 0.4422 CoDeT-G-DL: 0.9072	 RCNN-BB (Normalized): 0.2453 CoDeT-G-DL: 0.1342	 RCNN-BB (N): 0.1789 CoDeT-G-DL: 0.6357

Figure 4.6: CoDeT-G-DL vs. R-CNN-BB (Normalized) Scores. For each image, we show the changes in scores before and after performing dense CRF inference with our learned structured kernels. Columns 1-3 depict examples where our CoDeT-G-DL algorithm improved the scores of R-CNN-BB. Example bounding boxes with increased confidence are shown in column 1-2, column 3 shows the cases where bounding box confidence is decreased. Column 4 shows examples where the R-CNN-BB scores are better than ours. Note that, as in the false positive analysis of Fig. 4.8, these cases correspond to either parts of objects, or multiple objects of the same class.



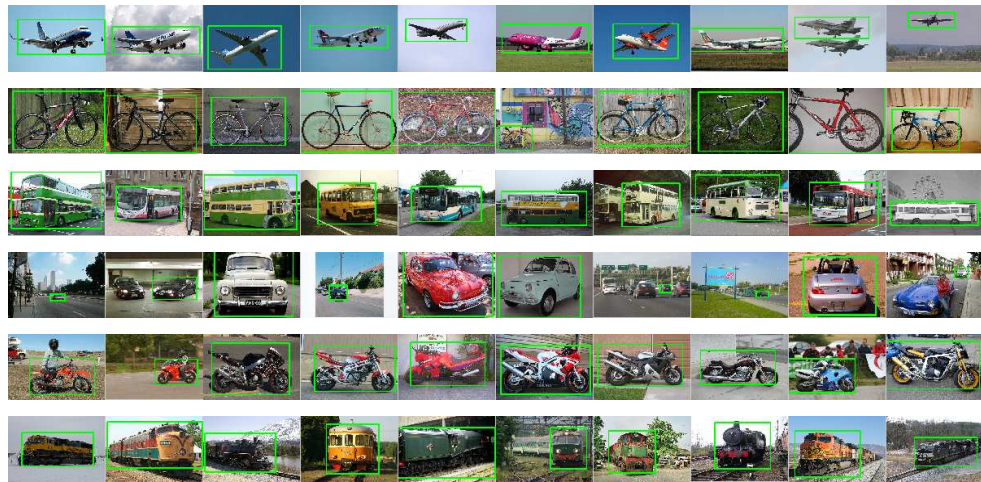


Figure 4.7: **CoDeT-G-DL top detections.** We show the top-scoring detections in decreasing order from left to right for 10 representative classes. If an image has multiple detections, only its top-scoring detection is shown. Note that our top-scoring detections correspond to accurate object localizations.

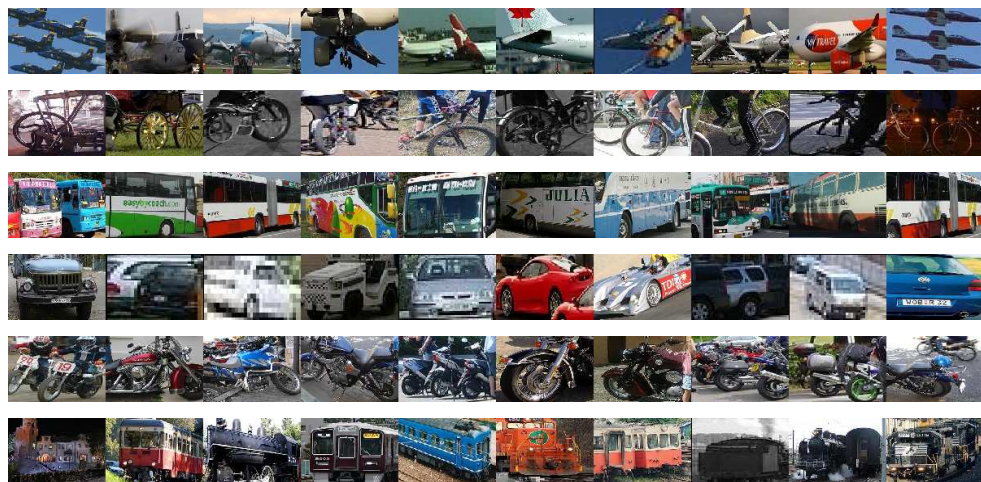


Figure 4.8: **CoDeT-G-DL top false positives.** We show the top-scoring false positives in decreasing order from left to right for 8 representative classes. Note that our top scoring false positives belong to two main categories: **(i)** Parts of objects are confused with complete objects due to high similarity with objects that are truly partially observed; and **(ii)** Bounding boxes containing multiple objects of the same category are confused with single objects due to high similarity with other boxes truly containing single objects. Note, however, that our top-scoring false positives do not contain truly absurd detections.

## 4.7 Conclusion

In this work, we have introduced a novel large scale object co-detection method that simultaneously considers multiple object classes. Our method uses a unified framework to jointly learn the CRF and the similarity, and thus yields better performance. This approach based on a fully-connected CRF allows us to incorporate contextual information within and across images, as well as within and across the classes. Furthermore, our structural boosting strategy lets us benefit from rich, high-dimensional features to learn the object relationships within our CRF framework. Our experiments have demonstrated the benefits of our approach over the state-of-the-art methods on PASCAL VOC 2007 and 2012, where we obtained state-of-the-art detection accuracies. An important limitation of proposed technique is that it relies on separately trained model and cannot be trained in an end-to-end manner. In the next chapter, we propose to learn the class-agnostic geometric model and instance similarity jointly as well as to exploit the instance segmentations within the bounding boxes to improve the accuracy of object detection and localization.

---

# Class-agnostic Similarity Learning with a Deep Structured Network

---

Generating object proposals has become a key component of modern object detection pipelines. Indeed, the performance of modern object detection techniques [Girshick et al., 2014; Hariharan et al., 2014; Hayder et al., 2015; Girshick, 2015] including methods discussed in Chapters 3 and 4, is upper bounded by the performance of the object proposals methods.

However, most existing methods generate the object candidates independently of each other. In this chapter, we present an approach to co-generating object proposals in multiple images, thus leveraging the collective power of multiple object candidates. In particular, we introduce a deep structured network that jointly predicts the objectness scores and the bounding box locations of multiple object candidates. Our deep structured network consists of a fully-connected Conditional Random Field built on top of a set of deep Convolutional Neural Networks, which learn features to model both the individual object candidates and the similarity between multiple candidates. To train our deep structured network, we develop an end-to-end learning algorithm that, by unrolling the CRF inference procedure, lets us backpropagate the loss gradient throughout the entire structured network. We demonstrate the effectiveness of our approach on two benchmark datasets, showing significant improvement over state-of-the-art object proposal algorithms.

## 5.1 Generating Object Proposals

Objectness has essentially lead to a paradigm shift in object detection. Instead of the traditional sliding window approach, objectness facilitates detection by proposing a smaller number of interesting candidate regions. These object proposals have now become ubiquitous in state-of-the-art detectors [Girshick et al., 2014; Hariharan et al., 2014; Girshick, 2015]. While object proposals have also been considered in the context

of depth images [Zheng et al., 2015b], in this chapter, we focus on methods designed for RGB images, which are more common for large-scale object detection.

### 5.1.1 Limitations of Existing Approaches

Generating object proposals has recently become one of the key components of modern object detection techniques [Girshick et al., 2014; Hariharan et al., 2014; Girshick, 2015] and methods discussed earlier in Chapters 3 and 4. By filtering out the irrelevant portions of the input image, these proposals hugely reduce the search space of object detectors, which has proven beneficial for both speed and accuracy.

Existing object proposal methods [Alexe et al., 2010; Cheng et al., 2014; Krähenbühl and Koltun, 2014; Pinheiro et al., 2015; Uijlings et al., 2013; Zitnick and Dollár, 2014; Hosang et al., 2016], however, all generate candidate detections one by one, independently of each other. By contrast, in a parallel line of research as shown in Chapters 3 and 4, object co-detection [Bao et al., 2012; Guo et al., 2013] has emerged as an effective approach to leveraging the information jointly contained in multiple images to improve detection accuracy. Unfortunately, to model the similarity of multiple objects, existing methods rely on either handcrafted features [Bao et al., 2012; Guo et al., 2013; Shi et al., 2013] and 3, or features learned for object recognition 4. As a consequence, they are ill-suited to handle general object proposals, whose appearance is subject to much larger variations than specific object classes.

Most objectness methods rely on well-engineered handcrafted features. For instance, Alexe et al. [Alexe et al., 2010] introduced a generic objectness measure using four image cues, including multi-scale saliency, color contrast, edge density and superpixel straddleness. Instead of performing exhaustive search over the image, the Selective Search method of Uijlings et al. [Uijlings et al., 2013] utilizes an image over-segmentation. This method achieves high accuracy, and is therefore widely used as an initial step for detection. Krähenbühl et al. [Krähenbühl and Koltun, 2014] introduced a fast method based on the geodesic distance transform, which can be computed in near-linear time and generates object proposals at different scales. To date, the fastest method to generate object proposals is that of Cheng et al. [Cheng et al., 2014]. To achieve speed, this method relies on a binary representation of gradient-based features. This speed, however, comes at some loss in object localization accuracy. By contrast, Zitnick et al. [Zitnick and Dollár, 2014] exploited the edges and edge groups at the object boundaries to better localize the objects and generate good-quality proposals.

Recently, Pinheiro et al. [Pinheiro et al., 2015] proposed to go beyond handcrafted features for object proposal generation. In particular, [Pinheiro et al., 2015] leverages the representation power of deep networks to learn a discriminative CNN that



---

generates boxes and segmentation proposals. Ultimately, this method achieves state-of-the-art accuracy and competitive speed.

While effective, all existing objectness methods essentially generate one proposal at a time, without considering interactions between the proposals beyond simple exclusion via non-maximum suppression. By contrast, object co-detection [Bao et al., 2012] attempts to simultaneously exploit the similarity between pairs of objects to perform detection jointly in multiple images. Most existing co-detection methods rely on simple handcrafted features to model object similarity [Bao et al., 2012; Guo et al., 2013; Shi et al., 2013] and 3. By contrast, in Chapter 4, we performed feature selection using pre-trained CNN features. In both cases, however, the resulting techniques are ill-suited to produce general object proposals, because the employed features are tuned to the problem of detecting specific objects. Here, instead, we co-generate object proposals from multiple images by introducing a deep structured network that lets us learn general object features, as well as pairwise features to model proposal similarity. To the best of our knowledge, leveraging the power of multiple images has never been achieved in the context of objectness.

As discussed in chapter 1, the idea of deep structured neural networks has nonetheless been exploited in the past and can be traced back to the 90s [LeCun et al., 1998]. More recently, such structured networks have been exploited for the task of object recognition and semantic segmentation [Chen et al., 2015b; Schwing and Urtasun, 2015; Zheng et al., 2015a]. In this context, Schwing et al. [Schwing and Urtasun, 2015] and Zheng et al. [Zheng et al., 2015a] have also proposed to exploit the efficient mean-field inference procedure of [Krähenbühl and Koltun, 2011]. However, these approaches, beside tackling a different problem than ours, still rely on simple handcrafted features in their pairwise term. By contrast, here, in addition to the unary features, we also learn pairwise features that are back-propagated throughout the entire network. As demonstrated by our experiments, by learning this similarity, we effectively exploit the joint information of multiple object candidates to produce high-quality object proposals.

### 5.1.2 Our Idea

In this chapter, we introduce an approach to co-generating object proposals in multiple images. To this end, we propose a deep structured network that lets us learn features to model (i) the appearance of individual object candidates; and (ii) the similarity between multiple object candidates. As a result, our model is able to leverage the collective power of multiple object candidates, while coping with the large appearance variability of general object proposals.

More specifically, given an initial pool of object candidates, our model consists

of a fully-connected Conditional Random Field (CRF) built on top of a set of deep Convolutional Neural Networks (CNNs), one for each candidate. The CNN module of each candidate predicts (i) an objectness score; (ii) a bounding box location; and (iii) a low-dimensional feature vector employed in the pairwise term of the CRF. This pairwise term takes the form of a Gaussian kernel, which allows us to perform inference efficiently [Krähenbühl and Koltun, 2011], even for large numbers of candidates. Altogether, the resulting deep structured model jointly produces improved objectness scores for multiple candidates and refined locations for the foreground objects.

We introduce an end-to-end learning algorithm to estimate the weights of our deep structured network. To this end, we follow a stochastic gradient descent procedure using mini-batches on which we define the CRF. By unrolling the iterations of our CRF inference strategy, we can backpropagate the gradient of our loss function throughout the entire structured network. This lets us learn the similarity of pairs of candidates, thus effectively benefitting from multiple candidates to co-generate high-quality object proposals.

We demonstrate the effectiveness of our approach on two benchmark datasets for object proposal generation: Pascal VOC 2007 [Everingham et al., 2010] and MS COCO [Lin et al., 2014]. Our experiments evidence the benefits of leveraging multiple images for object proposal generation over state-of-the-art methods that generate the proposals individually.

## 5.2 Co-Generating Object Proposals

We tackle the problem of jointly predicting a set of object proposals from multiple images<sup>1</sup>. Ultimately, our goal is to leverage the collective information contained in a set of object candidates to obtain high-quality object proposals. To this end, we start from an initial pool of object candidates and develop a deep structured network that improves their ranking and localization. This deep structured network models both the appearance of each object candidate and the interactions between these candidates.

Formally, let  $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$  be an initial pool of object candidates generated from a set of images  $\mathcal{I}$ , typically by an existing object proposal method, such as Bing or Selective Search. Each object candidate  $\mathbf{x}_i$  denotes an image window cropped from one of the images in  $\mathcal{I}$ . For each  $\mathbf{x}_i$ , we introduce (i) a binary variable  $y_i$ , which indicates whether  $\mathbf{x}_i$  is a foreground object ( $y_i = 1$ ) or background clutter ( $y_i = 0$ ); and (ii) a continuous real-valued vector  $\mathbf{t}_i = (t_{i,x}, t_{i,y}, t_{i,w}, t_{i,h})^T$  containing the offset

<sup>1</sup>Note that our approach also applies to the multiple proposals of a single image.

to the true object location if  $\mathbf{x}_i$  is a foreground object and 0 otherwise.

We formulate the co-generation of object proposals for image set  $\mathcal{I}$  as a multi-label prediction problem, in which we simultaneously predict the labels  $\mathbf{Y} = \{y_1, \dots, y_N\}$  and the location offsets  $\mathbf{T} = \{t_1, \dots, t_N\}$  of the object candidate set  $\mathbf{X}$ . To this end, we develop a deep structured network that defines a joint distribution over  $\mathbf{Y}$  and  $\mathbf{T}$  given  $\mathbf{X}$ , denoted by  $P(\mathbf{Y}, \mathbf{T} | \mathbf{X})$ . Our deep structured network consists of two components: One CNN for each object candidate, with weights shared across all candidates, which provides a general object representation, and one fully-connected CRF, which captures the similarity between every pair of object candidates.

Specifically, the joint distribution defined by the deep structured network can be written as

$$P(\mathbf{Y}, \mathbf{T} | \mathbf{X}) = \frac{1}{Z(\mathbf{X})} \exp \left( - \sum_{i=1}^N \phi(y_i, t_i | \mathbf{x}_i) - \sum_{i=1}^N \sum_{j>i}^N \psi(y_i, y_j | \mathbf{x}_i, \mathbf{x}_j) \right), \quad (5.1)$$

where  $Z(\cdot)$  is the partition function, and  $\phi, \psi$  are the unary and pairwise potential functions, respectively. The unary potential  $\phi$  encodes how likely a candidate  $\mathbf{x}_i$

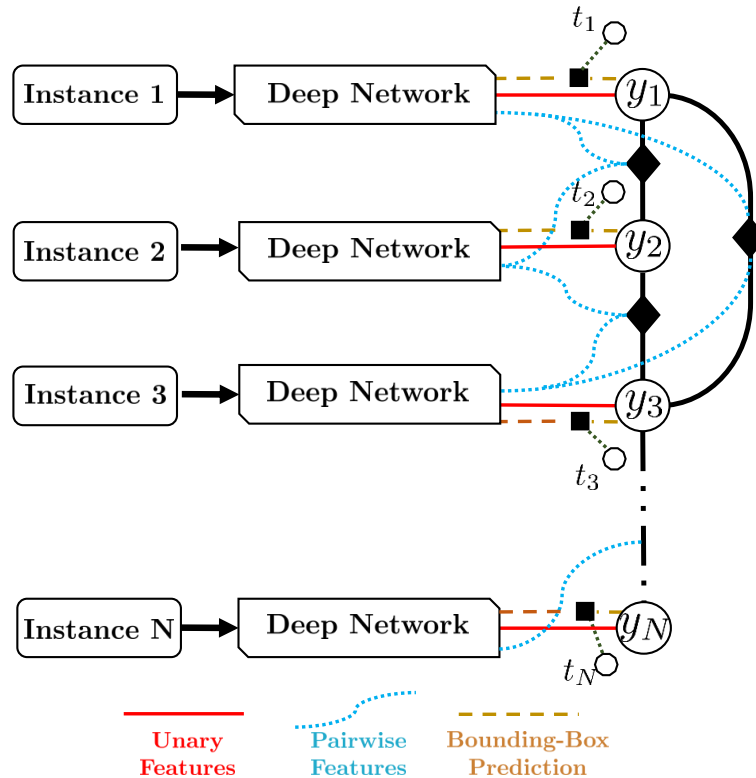


Figure 5.1: **Overview of our deep structured network for object proposal co-generation.** Our model consists of one deep CNN module per object candidate, linked by a fully-connected CRF.

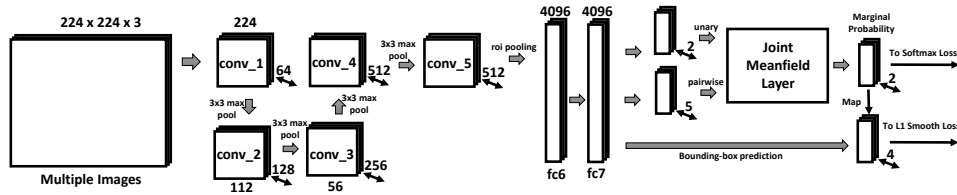


Figure 5.2: **Detailed architecture of our deep structured network for object proposal co-generation.** Each input image first goes through a series of convolutional layers, followed by Region-of-Interest (RoI) pooling corresponding to the different candidates in the image. Each candidate then passes through several fully-connected layers to predict unary features, pairwise features and bounding box location offsets. The features are finally employed in a fully-connected CRF. During training, our model makes use of a multi-task loss.

is to be assigned label  $y_i$  with location offset  $t_i$ , while the pairwise potential  $\psi$  is a symmetric term encouraging any two similar candidates to have the same label assignment.

To fully leverage the representative power of CNNs, we make both the unary and pairwise potentials depend on the CNN modules of our deep structured network. Intuitively, and as illustrated by Fig. 5.1, a CNN module corresponding to candidate  $x_i$  produces features for the unary term of  $x_i$  and features for the pairwise terms involving  $x_i$ . Furthermore, the pairwise term connects multiple CNN modules to form a structured prediction model. In the remainder of this section, we describe our network architecture and potential functions in details.

### 5.2.1 Deep CNNs for Individual Object Candidates

The network architecture constituting the CNN module for each individual object candidate  $x_i$  is depicted by Fig. 5.2. As mentioned above, this CNN module produces (i) a unary term consisting of an objectness score and of a refined object location; and (ii) a feature vector for the pairwise term. To this end, the output of this network consists of three sibling layers. The first one relies on a softmax layer to predict the foreground/background probabilities; the second one makes use of a regression layer that outputs the four real-valued coordinates of the foreground location offset; and the third one employs a fully-connected layer to generate a low-dimensional feature vector for the CRF pairwise terms.

More specifically, let us denote by  $f_i^{net}$  the output of the FC<sub>7</sub> layer of the CNN for

object candidate  $\mathbf{x}_i$ . The three network outputs are computed as

$$P_u(y_i) \propto \exp(\mathbf{w}_{u,y_i}^T \mathbf{f}_i^{net}) \quad (5.2)$$

$$\tilde{\mathbf{t}}_i = \mathbf{W}_r^T \mathbf{f}_i^{net} \llbracket y_i = 1 \rrbracket, \quad (5.3)$$

$$\mathbf{h}_i^p = \mathbf{W}_p^T \mathbf{f}_i^{net}, \quad (5.4)$$

where  $\mathbf{w}_{u,y_i}$ ,  $\mathbf{W}_r$  and  $\mathbf{W}_p$  denote the fully-connected weights to generate the label probabilities, the estimated object location offsets and the features for the pairwise potential, respectively. By  $\llbracket y_i = 1 \rrbracket$ , we mean that the estimated offset will be  $\mathbf{W}_r^T \mathbf{f}_i^{net}$  if the predicted label  $y_i = 1$ , and 0 otherwise. This lets us write our unary potential as

$$\phi(y_i, \mathbf{t}_i | \mathbf{x}_i) = -\mathbf{w}_{u,y_i}^T \mathbf{f}_i^{net} + \|\mathbf{t}_i - \mathbf{W}_r^T \mathbf{f}_i^{net} \llbracket y_i = 1 \rrbracket\|^2, \quad (5.5)$$

which measures the cost for a candidate  $\mathbf{x}_i$  to belong to the foreground/background class and have offset  $\mathbf{t}_i$ .

### 5.2.2 Fully-connected CRF for Candidate Similarity

On top of the deep CNN modules, we construct a fully-connected CRF, which models inter-candidate similarity. To this end, we define the pairwise potential  $\psi$  as a data-dependent smoothing term that encourages similar object candidates to share the same label. As in [Krähenbühl and Koltun, 2011], we restrict the data-dependent weight in the pairwise potential to take the form of a Gaussian kernel. This yields

$$\begin{aligned} \psi(y_i, y_j | \mathbf{X}_i, \mathbf{X}_j) &= \mu(y_i, y_j) k(\mathbf{h}_i^p, \mathbf{h}_j^p) \\ &= \mu(y_i, y_j) k(\mathbf{W}_p^T \mathbf{f}_i^{net}, \mathbf{W}_p^T \mathbf{f}_j^{net}), \end{aligned} \quad (5.6)$$

with

$$k(\mathbf{h}_i^p, \mathbf{h}_j^p) = \exp\left(-\frac{1}{2} \|\mathbf{h}_i^p - \mathbf{h}_j^p\|^2\right), \quad (5.7)$$

where  $\mathbf{h}^p$  is defined in Eq. 5.4, and  $\mu$  is a label compatibility function. While a general compatibility function can be learned [Krähenbühl and Koltun, 2013], in practice, we found that a Potts model, *i.e.*,  $\mu(y_i, y_j) = \llbracket y_i \neq y_j \rrbracket$ , was already effective.

Our deep structured model differs from the existing fully-connected CRFs for semantic labeling in several ways. First, since our nodes correspond to object proposals, our CRF implements multiple tasks, including labeling object candidates and refining their locations. In addition, we do not rely on the traditional bilateral kernels, which use manually selected features. Instead, we learn a low-dimensional feature representation that, when used in a Gaussian kernel, is able to encode candidate similarity. As a side effect, we do not need to define a covariance matrix for the kernel,

since the weights  $\mathbf{W}_p$  implicitly handle this. This simplifies the end-to-end learning of the full network.

### 5.2.3 Efficient Object Proposal Co-Generation

Given our deep structured model, we co-generate object proposals by taking a set of initial object candidates as input, and jointly inferring the MAP estimates of the objects' labels and location offsets, as well as the posterior marginal probabilities of the labels. Note that the MAP estimates of the labels and location offsets are decoupled. Indeed, for any label assignment  $y_i$ , the minimizer of the second term in Eq. 5.5 is given by

$$\mathbf{t}_i^* = \mathbf{W}_r^T \mathbf{f}_i^{net} \llbracket y_i = 1 \rrbracket, \quad (5.8)$$

since  $\mathbf{t}_i$  does not appear in the pairwise terms. Therefore, we can first compute the MAP label assignment and obtain the location offsets from Eq. 5.8.

To compute the MAP and posterior marginals of the label variables, we make use of the same efficient mean-field inference as in [Krähenbühl and Koltun, 2011]. Specifically, we approximate the joint label probability by a factorized distribution  $Q(\mathbf{Y}) = \prod_i \mathbf{q}_i(y_i)$ . The mean-field inference updates approximate the marginals iteratively as

$$\mathbf{q}_i^{(t)}(y_i) = \frac{1}{Z_i} \exp(\mathbf{w}_{u,y_i}^T \mathbf{f}_i^{net} - \sum_{j \neq i} \sum_{y_j} \mu(y_i, y_j) k(\mathbf{h}_i^p, \mathbf{h}_j^p) \mathbf{q}_j^{(t-1)}(y_j)), \quad (5.9)$$

where  $t$  denotes the iteration step, and  $Z_i$  is the approximate partition function (*i.e.*, the normalizer). These updates can be computed efficiently for a large number of object candidates using a fast Gaussian filtering technique. After performing inference, the MAP estimate of the object label is approximated by  $y_i^* = \arg \max_{y_i} \mathbf{q}_i(y_i)$ . We use the (approximate) marginal probabilities of the foreground class as scores to generate the final ranking of the object proposals.

## 5.3 Learning our Deep Structured Network

We now develop an end-to-end learning method to estimate the parameters of our multi-output deep structured network. To this end, let  $\mathcal{D} = \{\mathbf{X}, \hat{\mathbf{Y}}, \hat{\mathbf{T}}\} = \{(\mathbf{x}_i, \hat{y}_i, \hat{\mathbf{t}}_i)\}_{i=1}^N$  be a set of object candidates extracted from training images with ground-truth object bounding boxes. Note that this training set contains both foreground objects and background clutter, obtained with the same objectness algorithm as at test time. The offsets of these candidates are normalized by the size of the bounding boxes (scale-invariant) and expressed in log-space. Furthermore, to handle the iterative nature of

mean-field inference, we unroll its iterations and denote the output of the final one by  $\{\mathbf{q}_i^{(m)}\}_{i=1}^N$ .

To train our deep structured network, we employ a multi-task loss  $L$ , which combines a classification loss  $L_y$  for the object labels and a regression loss  $L_r$  for the box offsets. This loss can be expressed as

$$\begin{aligned} L(\mathcal{D}) &= L_y(Q^m, \hat{\mathbf{Y}}) + \lambda L_r(\mathbf{T}, \hat{\mathbf{T}}) \\ &= - \sum_i \left[ \log \mathbf{q}_i^{(m)}(\hat{y}_i) + \lambda \mathbb{I}[\hat{y}_i = 1] \|\mathbf{t}_i - \hat{\mathbf{t}}_i\|_{SL_1} \right], \end{aligned} \quad (5.10)$$

where  $\mathbf{T} = \{\mathbf{t}_i\}$  is the offset predictions from the network, and  $\lambda$  is a hyperparameter balancing the two tasks. For offset regression, Eq. 5.10 makes use of a smoothed  $L_1$  loss, denoted by  $\|\cdot\|_{SL_1}$  and defined as

$$\|\mathbf{z}\|_{SL_1} = \sum_k 0.5z_k^2 \mathbb{I}[|z_k| < 1] + (|z_k| - 0.5) \mathbb{I}[|z_k| \geq 1].$$

Our deep structured network has four sets of parameters, including the network weights  $\mathbf{W}_{cnn}$  before the  $\text{FC}_7$  layer of the CNN module, the unary term weights  $\mathbf{W}_u$ , the regression weights  $\mathbf{W}_r$  and the pairwise feature weights  $\mathbf{W}_p$ . We use stochastic gradient descent (SGD) to train those weights in an end-to-end manner. The overall training procedure consists of two steps: We first pre-train the CNN modules and then train the full structured model using mini-batches.

### 5.3.1 Pre-training the Deep CNN Module

In a first stage, we train the CNN module corresponding to the individual object candidates. In other words, in this stage, we ignore the pairwise features and the dense CRF layer. The CNN module has two outputs: the object label probability  $P_u(y_i)$  and the bounding box location offset  $\mathbf{t}_i$ . We train it using the same procedure as the Fast RCNN [Girshick, 2015].

More specifically, we start from a convolutional neural network (VGG-16 [Simonyan and Zisserman, 2015]) pre-trained on ImageNet, which gives us an initialization for the weights  $\mathbf{W}_{cnn}$ . We then define our training loss as

$$L_c = - \sum_i \left[ \log P_u(\hat{y}_i) + \lambda \mathbb{I}[\hat{y}_i > 0] \|\mathbf{t}_i - \hat{\mathbf{t}}_i\|_{SL_1} \right], \quad (5.11)$$

and adopt the same strategy of mini-batch sampling and back-propagation through RoI pooling layers as in [Girshick, 2015]. This pre-training step initializes the unary and regression weights ( $\mathbf{W}_u$  and  $\mathbf{W}_r$ ), and fine-tunes the network ones ( $\mathbf{W}_{cnn}$ ).

### 5.3.2 End-to-end Learning with Mini-batches

In a second stage, given the weight initializations described above, we learn our complete deep structured network. To this end, we follow an SGD procedure using mini-batches on which we define the fully-connected CRF. The main challenge of this procedure is to derive the gradient of the loss function  $L(\mathcal{D})$  with respect to the weight parameters and to compute this gradient efficiently. Below, we will focus on the gradient of  $L_y$ , since the second term  $L_r$ , not involving the CRF, can be handled in the same manner as in Fast RCNN (as in the pre-training of Section 5.3.1).

We derive a mean-field gradient method which computes the parameter gradients of our deep structured model recursively, similarly to [Krähenbühl and Koltun, 2013]. Note that, unlike [Krähenbühl and Koltun, 2013], we need to compute the gradient w.r.t. the unary *and pairwise* weights  $\mathbf{W}_u$  and  $\mathbf{W}_p$ , as well as the network features  $\mathbf{f}_i^{net}$  in order to backpropagate the gradient to the CNN modules.

As before, let us denote the marginals by  $\mathbf{q} = (\mathbf{q}_1^T, \dots, \mathbf{q}_N^T)^T$ . The gradient of the loss  $L_y$  w.r.t. a parameter  $\mathbf{w}$  can be written as

$$\frac{\partial L_y}{\partial \mathbf{w}} = \frac{\partial L_y}{\partial \mathbf{q}} \frac{\partial \mathbf{q}^T}{\partial \mathbf{w}}. \quad (5.12)$$

Let  $\mathbf{u} = (\mathbf{u}_1^T, \dots, \mathbf{u}_N^T)^T$  be the unary term for the label variables, where  $\mathbf{u}_i = -[\mathbf{w}_{u,0}, \mathbf{w}_{u,1}]^T \mathbf{f}_i^{net}$ , and  $\hat{\Psi}$  be the matrix form of the pairwise term, *i.e.*,  $\hat{\Psi} = \mathbf{K} \otimes \bar{\cdot}$ , where  $\mathbf{K} = [k_{ij}]_{N \times N}$  is the kernel matrix and  $k_{ij} = k(\mathbf{h}_i^p, \mathbf{h}_j^p)$ . Following [Krähenbühl and Koltun, 2013], the gradient can be recursively computed as

$$\frac{\partial L_y(\mathbf{q}^m(\mathbf{w}))}{\partial \mathbf{w}} = \sum_{t=1}^m \mathbf{b}^{(t)T} \left( \frac{\partial \mathbf{u}}{\partial \mathbf{w}} + \mathbf{q}^{(t-1)} \frac{\partial \hat{\Psi}}{\partial \mathbf{w}} \right), \quad (5.13)$$

where  $\mathbf{b}^{(t)} = (\mathbf{b}_1^{(t)T}, \dots, \mathbf{b}_N^{(t)T})^T$  is the normalized loss gradient at iteration  $t$ . This normalized loss gradient is defined recursively as

$$\mathbf{b}^{(m)} = A^{(m)} \left( \nabla L_y(\mathbf{q}^{(m)}) \right)^T \quad (5.14)$$

$$\mathbf{b}^{(t)} = A^{(t)} \hat{\Psi} \mathbf{b}^{(t+1)}, \quad t = 1, \dots, m-1, \quad (5.15)$$

where  $A^{(t)}$  is a block diagonal matrix with blocks  $A_i^{(t)} = \mathbf{q}_i^{(t)} \mathbf{q}_i^{(t)T} - \text{diag}(\mathbf{q}_i^{(t)})$ . We now derive the two partial derivatives in Eq. 5.13 for different weights and features.



**Unary weights and features.** The unary weight matrix  $\mathbf{W}_u$  only appears in the first term of Eq. 5.13. This term can be computed as

$$\frac{\partial \mathbf{b}^T \mathbf{u}}{\partial \mathbf{W}_u} = \sum_i \mathbf{b}_i \mathbf{f}_i^{net T}. \quad (5.16)$$

The gradient of the unary term w.r.t. the deep network features  $\mathbf{f}_i^{net}$  can be computed similarly as

$$\frac{\partial \mathbf{b}^T \mathbf{u}}{\partial \mathbf{f}_i^{net}} = \mathbf{W}_u^T \mathbf{b}_i. \quad (5.17)$$

Note that this feature gradient will be combined with the feature gradient of the pairwise term derived below.

**Pairwise weights and features.** For the pairwise term, we apply the chain-rule and first compute the gradient w.r.t. the pairwise features  $\mathbf{h}_i^p$ . This yields

$$\begin{aligned} \frac{\partial \mathbf{b}^T \hat{\Psi} \mathbf{q}}{\partial \mathbf{h}_i^p} &= \frac{\partial}{\partial \mathbf{h}_i^p} \left( \mathbf{b}^T \mathbf{K}^{(m)} \otimes \mu^{(m)} \mathbf{q} \right) \\ &= \sum_j (\mathbf{h}_j^p - \mathbf{h}_i^p) \mathbf{q}_j^T \mathbf{K}_{ij}^{(m)} \mu^{(m)} \mathbf{b}_i \\ &= \sum_j \mathbf{h}_j^p \mathbf{q}_j^T \mathbf{K}_{ij}^{(m)} \mu^{(m)} \mathbf{b}_i - \mathbf{h}_i^p \mathbf{b}_i^T \sum_j \mathbf{K}_{ij}^{(m)} \mu^{(m)} \mathbf{q}_j. \end{aligned} \quad (5.18)$$

This gradient can be computed efficiently using high-dimensional filtering on the permutohedral lattice [Krähenbühl and Koltun, 2011].

The gradient w.r.t. the pairwise weight matrix  $\mathbf{W}_p$  and the CNN features  $\mathbf{f}_i^{net}$  can be computed as

$$\frac{\partial \mathbf{b}^T \hat{\Psi} \mathbf{q}}{\partial \mathbf{W}_p} = \sum_i \frac{\partial \mathbf{b}^T \hat{\Psi} \mathbf{q}}{\partial \mathbf{h}_i^p} \frac{\partial \mathbf{h}_i^{p T}}{\partial \mathbf{W}_p} = \sum_i \frac{\partial \mathbf{b}^T \hat{\Psi} \mathbf{q}}{\partial \mathbf{h}_i^p} \mathbf{f}_i^{net T} \quad (5.19)$$

$$\frac{\partial \mathbf{b}^T \hat{\Psi} \mathbf{q}}{\partial \mathbf{f}_i^{net}} = \frac{\partial \mathbf{b}^T \hat{\Psi} \mathbf{q}}{\partial \mathbf{h}_i^p} \frac{\partial \mathbf{h}_i^{p T}}{\partial \mathbf{f}_i^{net}} = \mathbf{W}_p^T \frac{\partial \mathbf{b}^T \hat{\Psi} \mathbf{q}}{\partial \mathbf{h}_i^p}. \quad (5.20)$$

Altogether, we can compute all the gradients in a single forward and backward pass over our full deep structured network. The overall mean-field gradient computation framework is summarized in Algorithm 5.1. In practice, we use different step sizes for the unary, pairwise, regression and CNN weights, which helps the learning procedure focus on the pairwise and regression weights, while only fine-tuning the rest of the network.

**Algorithm 5.1** Mean-field Gradient for Classification Loss

**Input:** Features  $\{\mathbf{f}_i^{net}\}_{i=1}^N$ , Initial Weights  $\mathbf{W}_u, \mathbf{W}_p$ , Mean-field Iterations  $m$ , Loss Function  $L_y$   
**Output:** Feature and Weight Gradients:  $\mathbf{g}_w^u, \mathbf{g}_w^p, \mathbf{g}_{f,i}^{net}$

**Phase 1: Joint Inference**

```

1: procedure (Forward pass)
2:    $\mathbf{u}_i = \mathbf{q}_i^0 = -\mathbf{W}_u^T \mathbf{f}_i^{net}$ 
3:    $\mathbf{h}_i^p = \mathbf{W}_p^T \mathbf{f}_i^{net}$ 
4:   for  $t = 1 \dots m$  do
5:      $\mathbf{q}_i^{(t)} = \frac{1}{Z} \exp \left( \mathbf{u}_i - \sum_{j \neq i} k(\mathbf{h}_i^p, \mathbf{h}_j^p) \mu \mathbf{q}_j^{(t-1)} \right)$ 
6:   end for
7: end procedure

```

**Phase 2: Gradient Computation**

```

8: procedure (Backward pass)
9:    $\mathbf{g}_w^u \leftarrow 0$ 
10:   $\mathbf{g}_w^p \leftarrow 0$ 
11:   $\mathbf{g}_f^{net} \leftarrow 0$ 
12:   $\mathbf{A}_i^{(m)} = \mathbf{q}_i^{(m)} \mathbf{q}_i^{(m)T} - \text{diag}(\mathbf{q}_i^{(m)})$ 
13:   $\mathbf{b}^{(m-1)} = \mathbf{A}^{(m)} \left( \nabla L_y(\mathbf{q}^{(m)})^T \right)$ 
14:  for  $t = m - 1 \dots 1$  do
15:     $\mathbf{A}_i^{(t)} = \mathbf{q}_i^{(t)} \mathbf{q}_i^{(t)T} - \text{diag}(\mathbf{q}_i^{(t)})$ 
16:     $\mathbf{g}_w^u \leftarrow \mathbf{g}_w^u + \frac{\partial}{\partial \mathbf{W}_u} \left( \mathbf{b}^{(t)T} \mathbf{u} \right)$  ▷ Eq. 16
17:     $\mathbf{g}_{f,i}^{net} \leftarrow \mathbf{g}_{f,i}^{net} + \frac{\partial}{\partial \mathbf{f}_i^{net}} \left( \mathbf{b}^{(t)T} \mathbf{u} \right)$  ▷ Eq. 17
18:     $\mathbf{g}_w^p \leftarrow \mathbf{g}_w^p + \frac{\partial}{\partial \mathbf{W}_p} \left( \mathbf{b}^{(t)T} \hat{\Psi} \mathbf{q}^{(t)} \right)$  ▷ Eq. 19
19:     $\mathbf{g}_{f,i}^{net} \leftarrow \mathbf{g}_{f,i}^{net} + \frac{\partial}{\partial \mathbf{f}_i^{net}} \left( \mathbf{b}^{(t)T} \hat{\Psi} \mathbf{q}^{(t)} \right)$  ▷ Eq. 20
20:     $\mathbf{b}^{(t-1)} = \mathbf{A}^{(t)} \hat{\Psi} \mathbf{b}^{(t)}$ 
21:  end for
22: end procedure

```

## 5.4 Experiments

In this section, we demonstrate the effectiveness of our method on the problem of large-scale proposal generation for generic objects. To this end, we evaluate our approach on two challenging datasets with multiple object classes, *i.e.*, PASCAL VOC 2007 and Microsoft COCO, and compare our results with those of the state-of-the-art object proposal methods.

### 5.4.1 Datasets and Setup

The Pascal VOC 2007 dataset [Everingham et al., 2010] comprises 5011 training-validation (trainval) images and 4952 test images, and the Microsoft COCO 2014

---

validation dataset [Lin et al., 2014] contains 82783 training images and 40504 validation images. For Pascal VOC 2007, we used all the trainval data to learn our deep structured model, and evaluated it using all the test data. For Microsoft COCO, we also used all the training images to learn our model, but, following [Pinheiro et al., 2015], used only the first 5000 validation images for evaluation purpose.

In our experiments, we evaluated several techniques to generate the initial set of candidates. The choice of the particular initial proposal generation methods we use was motivated by the fact that [Hosang et al., 2016], whose protocol we follow, and other existing methods used them for these datasets. In particular, we used Bing [Cheng et al., 2014] for both datasets, as well as Selective Search [Uijlings et al., 2013] for Pascal and Edge Box [Zitnick and Dollár, 2014] for MS COCO, which represent the most commonly-used methods for each dataset, respectively. For training, we obtained the mini-batches by randomly sampling 2 images from the training set and taking 512 candidates per image. At test time, given the initial candidates of all the test images, we extracted the unary features, bounding box location offsets and pairwise features for each candidate using the deep CNN, and then performed inference in the fully-connected CRF using all the candidates. This allows us to truly exploit the similarities across all the test data, and, thanks to the efficient inference procedure, remains fast (e.g., 1.4 sec. for roughly 10k Bing candidates per image).

The standard error measures to evaluate object proposal quality rely on Average Recall (AR). It has been shown that, for a fixed number of proposals per image, AR correlates well with the mean Average Precision of the object detector applied to the proposals [Hosang et al., 2016]. We therefore report our results using the average recall metrics defined by Hosang et al. [Hosang et al., 2016] and the COCO-style metrics [Lin et al., 2014] used in [Pinheiro et al., 2015].

### 5.4.2 Results on VOC 2007

In Table 5.1, we provide a quantitative comparison of our approach with state-of-the-art objectness methods according to the criteria of [Lin et al., 2014] on the Pascal VOC 2007 dataset. The results of these baselines were directly reproduced from their respective papers. Note that our approach yields state-of-the-art results; for 10 and 100 proposals, when using Bing candidates, and for 1000 proposals, when using Selective Search candidates. Table 5.1 also shows that, when selecting 100 proposals, our co-generation approach yields consistent improvement across different sizes of objects. In terms of runtime, our method remains highly competitive. Altogether, we believe that these results clearly evidence the benefits of co-generating the object proposals.

The results of our approach and of the baselines according to the criteria defined

PASCAL VOC07	AR@10	AR@100	AR@1000	AR@Small	AR@Medium	AR@Large	Time (sec)
Bing	0.141	0.262	0.344	0.000	0.083	0.369	0.003
EdgeBoxes	0.203	0.407	0.601	0.035	0.159	0.559	0.25
Geodesic	0.121	0.364	0.596	-	-	-	1.7
Selective Search	0.085	0.347	0.618	0.017	0.134	0.364	10
MCG	0.232	0.462	0.634	0.073	0.228	0.618	30
Deep-Mask	0.337	0.561	0.690	-	-	-	1.2
Ours Co-Obj (Sel. Search)	0.325	0.509	<b>0.745</b>	0.114	0.321	0.629	1.1
Ours Co-Obj (Bing)	<b>0.430</b>	<b>0.602</b>	0.675	<b>0.453</b>	<b>0.517</b>	<b>0.654</b>	1.4

Table 5.1: **AR analysis on the PASCAL VOC 2007 test set:** We compare our method with state-of-the-art object proposal baselines according to the criteria of [Lin et al., 2014]. The results of our approach are provided in Rows 7-8 when using Bing and Selective Search to generate the initial candidates, respectively. The AR for small, medium and large objects were computed for 100 proposals. Note that our co-generation approach outperforms the state-of-the-art baseline in all metrics. The difference in speed between two versions of our approach is due to the fact that Bing yields a larger candidate pool than Selective Search.

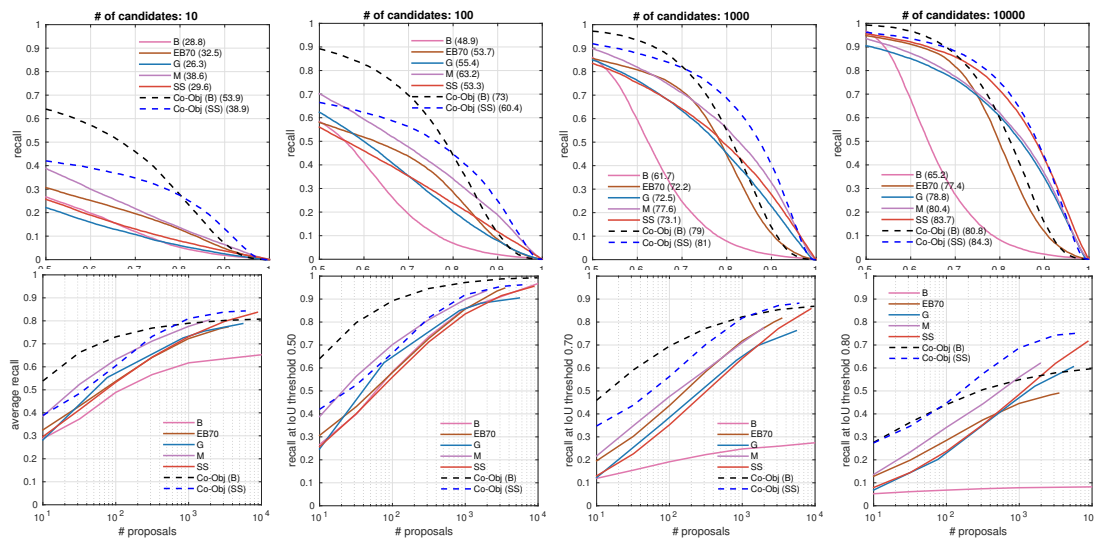


Figure 5.3: **Pascal VOC 2007 test:** We compare our method with state-of-the-art object proposal baselines according to the criteria of [Hosang et al., 2016]. **Left: Recall v.s. IoU threshold.** These recall curves were generated using the highest-scoring 10, 100, 1000 and 10000 object proposals, respectively. **Right: Recall v.s. Number of Proposals.** The first plot shows the AR, and the remaining recall curves were generated using IoU thresholds of 0.5, 0.7 and 0.8, respectively. In all the plots, the dashed lines correspond to our co-generation results, in blue when using Selective Search candidates (**Co-Obj (SS)**) and in black when using Bing candidates (**Co-Obj (B)**). The baselines correspond to Bing (**B**), EdgeBoxes (**EB70**), Geodesic (**G**), MCG (**M**) and SelectiveSearch (**SS**). These results clearly evidence the benefits of our co-generation approach.

by Hosang et al. [Hosang et al., 2016] are shown in Fig. 5.3<sup>2</sup>. In particular, the top row of Fig. 5.3 depicts the recall as a function of the Intersection over Union (IoU) threshold when using the 10, 100, 1000 & 10000 highest-scoring bounding boxes per image, respectively. The bottom row of Fig. 5.3 shows the AR and the recall as a function of the number of proposals for IoU thresholds of 0.5, 0.7 and 0.8, respectively. Again, these curves clearly show that co-generating the proposals yields to much better object bounding boxes. In particular, it is interesting to note that, even though the AR of the initial Bing bounding boxes is quite low, it is boosted to state-of-the-art results after our deep structured co-generation process. Note also that, while initially better, the Selective Search candidates still benefit from our approach. Interestingly, at high IoU, our results with Bing candidates tend to outperform our results with Selective Search candidates.

To evidence that our approach is not simply learning the behavior of a particular proposal method, we evaluated it with different proposal generation techniques during training and test time. As shown in Table 5.2, our method still outperforms the initial proposal generation methods, thus evidencing that it truly learns the relevant context for the object candidates themselves. We acknowledge, however, that the best results are obtained when using the same method at training and test time.

PASCAL VOC07	Train	Test	AR@10	AR@100	AR@1000
Bing	-	-	0.141	0.262	0.344
MCG	-	-	0.232	0.462	0.634
Ours Co-Obj	MCG	MCG	0.365	0.573	<b>0.709</b>
Ours Co-Obj	MCG	Bing	0.350	0.481	0.558
Ours Co-Obj	Bing	MCG	0.392	0.517	0.641
Ours Co-Obj	Bing	Bing	<b>0.430</b>	<b>0.602</b>	0.675

Table 5.2: **Using different initial proposal methods:** Rows 1-2 show the baseline object proposal methods. Rows 3-6 show our results using various candidate generation options at training and test time.

### 5.4.3 Results on Microsoft COCO

The results on Microsoft COCO, using the same metrics as before, are provided in Table 5.3 and Fig. 5.4, respectively. The same conclusions as before can be drawn from this analysis: Co-generating object proposals clearly is beneficial over generating the proposals independently. Our approach with EdgeBox initial candidates yields state-of-the-art results, with our Bing-based approach still outperforming the baselines for

<sup>2</sup>Note that we do not have access to the code, or the bounding boxes, of [Pineiro et al., 2015], and were thus unable to compute these curves for their approach.

Microsoft COCO 2014	AR@10	AR@100	AR@1000	AR@Small	AR@Medium	AR@Large
Bing	0.042	0.100	0.189	0.001	0.063	0.319
EdgeBoxes	0.074	0.178	0.338	0.015	0.134	0.502
Geodesic	0.040	0.180	0.359	-	-	-
Selective Search	0.052	0.163	0.357	0.012	0.132	0.466
MCG	0.101	0.246	0.398	0.008	0.119	0.530
DeepMask	0.153	0.313	0.446	-	-	-
Ours Co-Obj (Bing)	0.183	0.340	0.423	<b>0.111</b>	0.438	0.590
Ours Co-Obj (Edge Boxes 70)	<b>0.189</b>	<b>0.366</b>	<b>0.492</b>	0.107	<b>0.449</b>	<b>0.686</b>

Table 5.3: **AR analysis on the MS COCO validation set:** We compare our method with state-of-the-art object proposal baselines according to the criteria of [Lin et al., 2014]. The results of our approach are provided in Rows 7-8 when using Bing and EdgeBox to generate the initial candidates, respectively. The AR for small, medium and large objects were computed for 100 proposals. Note that our co-generation approach outperforms the state-of-the-art baseline in all metrics.

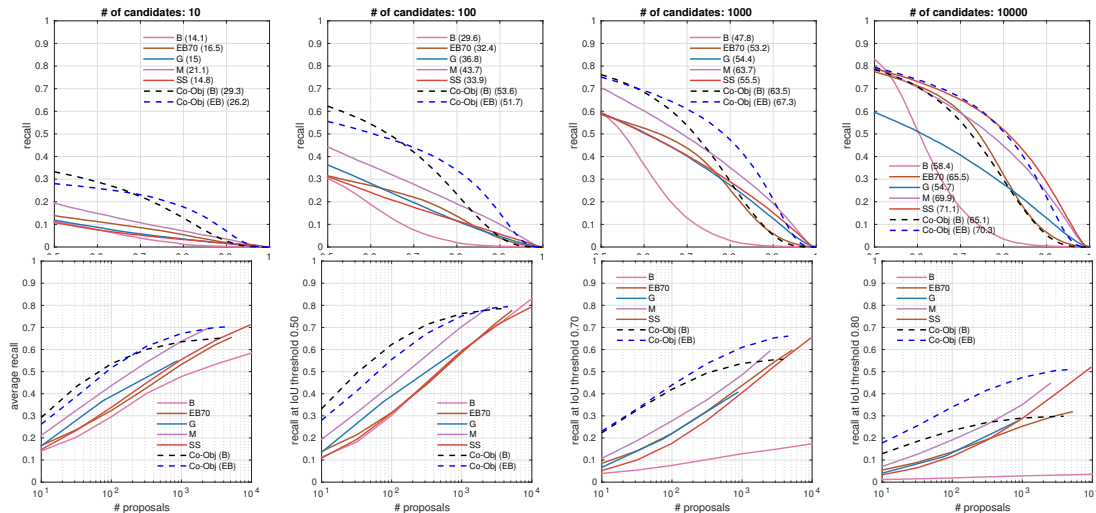


Figure 5.4: **MS COCO validation:** We compare our method with state-of-the-art object proposal baselines according to the criteria of [Hosang et al., 2016]. **Top: Recall v.s. IoU threshold.** These recall curves were generated using the highest-scoring 10, 100, 1000 and 10000 object proposals, respectively. **Bottom: Recall v.s. Number of Proposals.** The first plot shows the AR, and the remaining recall curves were generated using IoU thresholds of 0.5, 0.7 and 0.8, respectively. In all the plots, the dashed lines correspond to our co-generation results, in blue when using EdgeBox candidates (**Co-Obj (EB)**) and in black when using Bing candidates (**Co-Obj (B)**). The baselines correspond to Bing (**B**), EdgeBoxes (**EB70**), Geodesic (**G**), MCG (**M**) and SelectiveSearch (**SS**). These results again evidence the benefits of our co-generation approach.

all metrics, with the exception of AR@1000. The improvement due to our approach is again consistent across all object sizes. The runtimes of our method on the MS COCO dataset are 1 and 0.8 sec per image when using Bing and EdgeBox, respectively.

## 5.5 Deep Co-Objectness and Detection

To demonstrate the impact of having better object proposals on object detection, we employed our proposals within the Fast RCNN algorithm of [Girshick, 2015]. In Table 5.4, we compare the resulting Average Precisions with the of state-of-the-art object detectors on Pascal VOC 2007. We used selective search proposals by Hosang et al. [Hosang et al., 2016] as initial pool of bounding-boxes to co-generate the object proposals. Following [Hoiem et al., 2012], we also provide a detailed comparison of different performance criteria in Fig. 5.5. Note that we outperform the original Fast RCNN by 4.2% on average, with up to 12% for some classes, such as *bottle*. Note also that we outperform the state-of-the-art object co-detection approach of Chapter 4 and the region proposal network of [Ren et al., 2015]. The detection trends for all the object categories are detailed in Fig. 5.6.

VOC 2007 (test)	Fast RCNN SS	CoDet-G-DL SS	RPN	Fast RCNN SS (Q)	Co-Obj + Fast RCNN (Ours)
<b>aero</b>	74.5	<b>75.8</b>	70.0	74.9	73.4
<b>bike</b>	78.3	78.1	<b>80.6</b>	77.4	79.4
<b>bird</b>	69.2	69.3	70.1	69.9	<b>71.9</b>
<b>boat</b>	53.2	53.8	57.3	58.4	<b>61.6</b>
<b>bottle</b>	36.6	36.9	<b>49.9</b>	44.6	48.4
<b>bus</b>	77.3	77.5	78.2	<b>84.1</b>	<b>81.2</b>
<b>car</b>	78.2	79.0	80.4	81.2	<b>81.9</b>
<b>cat</b>	82.0	82.5	82.0	83.6	<b>84.8</b>
<b>chair</b>	40.7	40.1	<b>52.2</b>	47.4	47.5
<b>cow</b>	72.7	73.5	75.3	78.0	<b>78.2</b>
<b>table</b>	67.9	67.7	67.2	67.5	<b>69.6</b>
<b>dog</b>	79.6	81.4	80.3	80.7	<b>81.5</b>
<b>horse</b>	79.2	<b>82.2</b>	79.8	<b>84.3</b>	81.9
<b>mbike</b>	73.0	75.4	75.0	76.7	<b>77.8</b>
<b>person</b>	69.0	70.0	<b>76.3</b>	73.5	75.5
<b>plant</b>	30.1	33.4	<b>39.1</b>	37.4	38.1
<b>sheep</b>	65.4	65.4	68.3	70.9	<b>73.4</b>
<b>sofa</b>	<b>70.2</b>	70.0	67.3	69.1	68.9
<b>train</b>	75.8	74.3	<b>81.1</b>	77.5	78.0
<b>tv</b>	65.8	67.2	67.6	66.0	<b>68.0</b>
<b>mAP</b>	66.9	67.7	69.9	70.2	<b>71.1</b>

Table 5.4: **Detection Average Precision(%) on the PASCAL VOC 2007 test set:** Columns 1-3 show the multi-class object detection state-of-the-art results for Fast RCNN (SS) [Girshick, 2015], CoDet-G-DL (SS) of Chapter 4, RPN [Ren et al., 2015] and Fast RCNN (SS) respectively. The results of our approach are provided in Row 4.

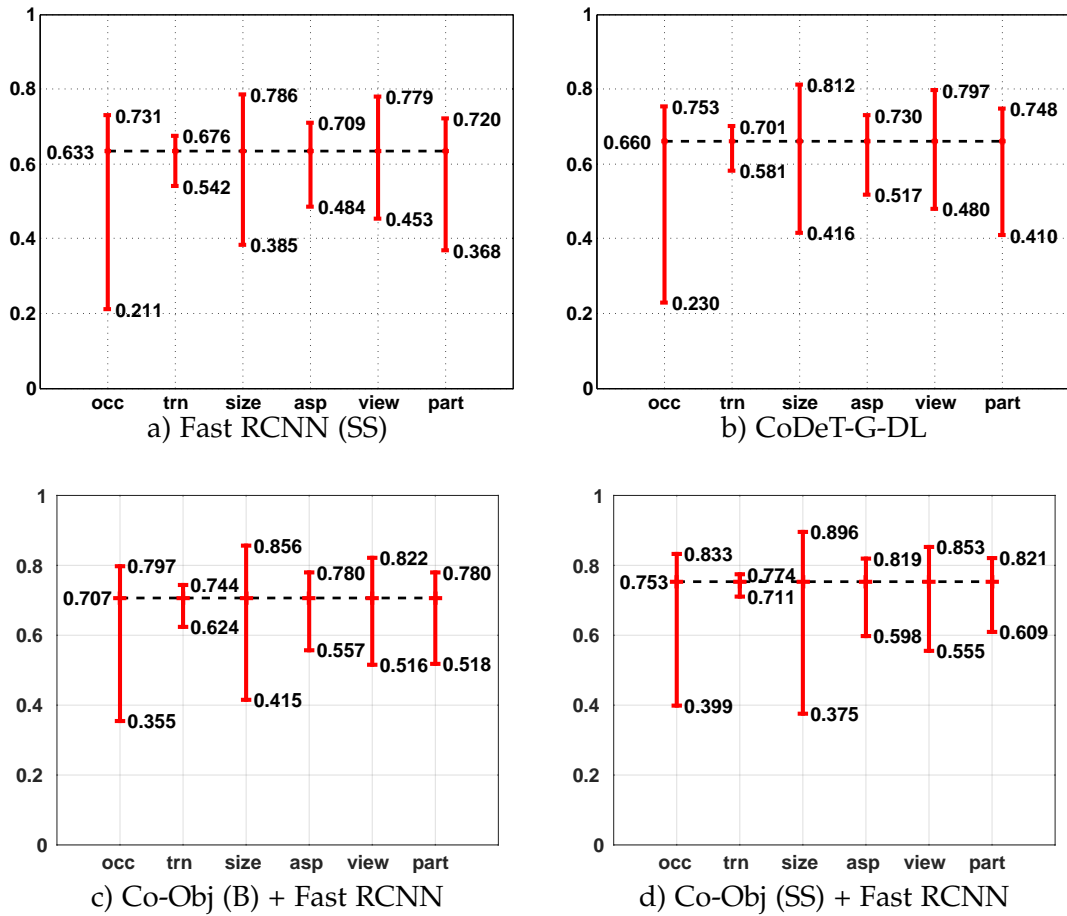


Figure 5.5: **Sensitivity and impact analysis:** Overall detailed performance comparison using different metrics (i.e., occlusion, truncated, size, aspect ratio, view point and part visibility). The black dashed line indicates the overall average normalized precision  $AP_N$ .

## 5.6 Conclusion

We have introduced a framework to jointly generate object proposals from multiple images, thus leveraging the collective power of multiple object candidates. Our method is based on a deep structured network that jointly predicts the objectness scores and the bounding box locations of multiple object candidates by extracting features that model the individual object candidates and the similarity between them. Our experiments have demonstrated the benefits of our approach over the state-of-the-art methods that generate object proposals individually. Note that the sub-networks for co-generating object proposals and object detection can be combined to train an end-to-end system. However, we observed that it requires more memory to jointly train these networks. In the next chapter, we extend object detection framework to obtain a detailed pixel-wise mapping of each object instance.



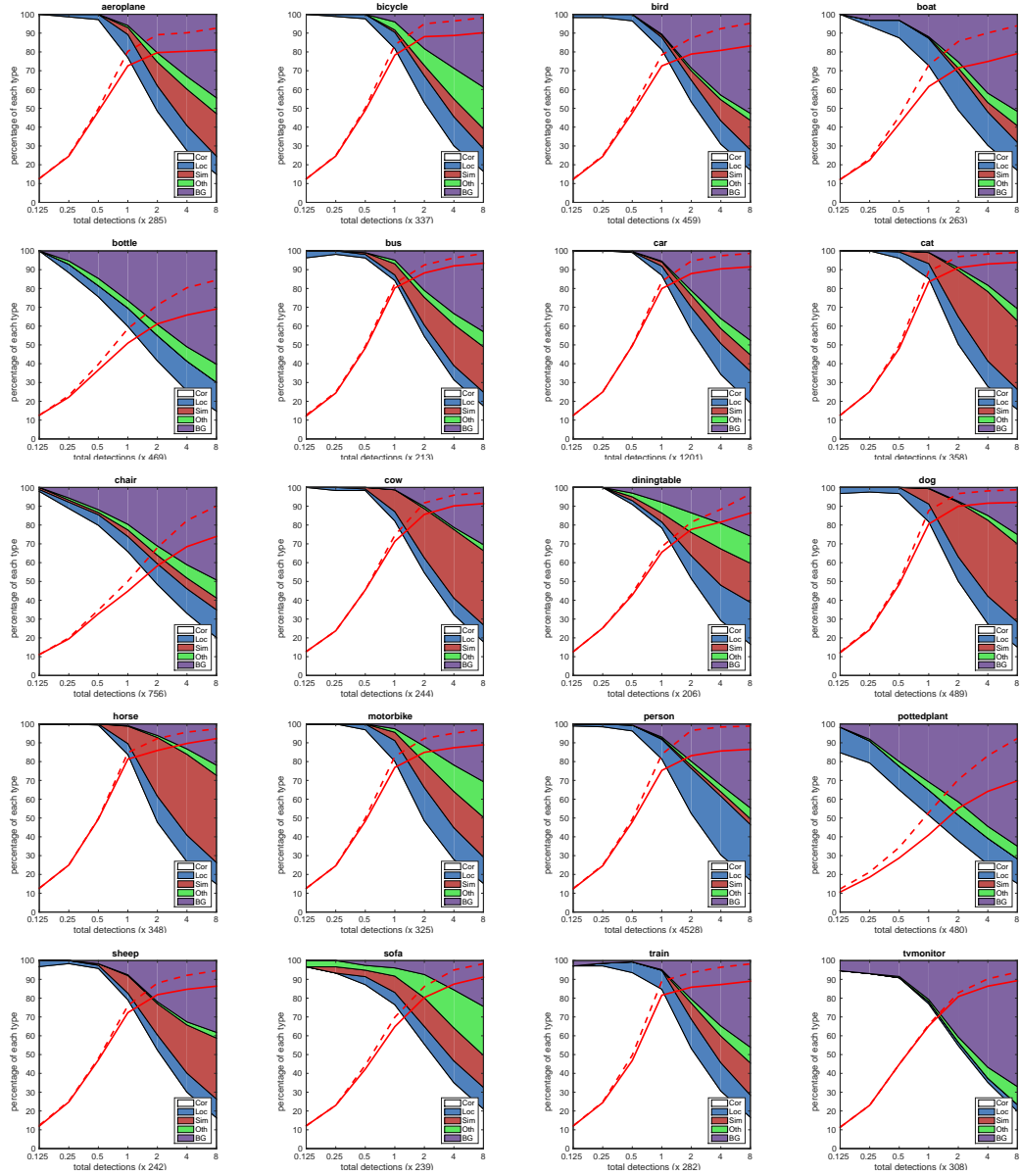


Figure 5.6: **Deep co-objectness and detection trends.** Following [Hoiem et al., 2012], we show the evolution of the type of detection as the number of detections increases; The white areas correspond to the correct detections; The blue areas represent the detections with localization error; The red areas correspond to confusion with a similar category; The green areas represent the confusion with a dissimilar category. Finally, the curves show the recall as a function of the number of objects (dashed=weak localization, solid=strong localization).



---

# Beyond Boxes: Boundary-Aware Instance Segmentation

---

Learning multiple related tasks jointly to improve generalization performance is an important aspect of multi-task learning. This is in contrast with methods proposed in Chapters 3, 4 and 5, where we exploit multiple images at test time. In this chapter, we address the problem of instance-level semantic segmentation, which aims at jointly detecting, segmenting and classifying every individual object in an image. In this context, existing methods typically propose candidate objects, usually as bounding boxes, and directly predict a binary mask within each such proposal. As a consequence, they cannot recover from errors in the object candidate generation process, such as too small or shifted boxes.

In this chapter, we introduce a novel object segment representation based on the distance transform of the object masks. We then design an object mask network (OMN) with a new residual-deconvolution architecture that infers such a representation and decodes it into the final binary object mask. This allows us to predict masks that go beyond the scope of the bounding boxes and are thus robust to inaccurate object candidates. We integrate our OMN into a Multitask Network Cascade framework, and learn the resulting boundary-aware instance segmentation (BAIS) network in an end-to-end manner. Our experiments on the PASCAL VOC 2012 and the CityScapes datasets demonstrate the benefits of our approach, which outperforms the state-of-the-art in both object proposal generation and instance segmentation.

## 6.1 From Detection to Instance Segmentation

Object detection methods provide an initial guess (bounding-box) where an object might exist in an image. However, segmenting the exact outline is a challenging task. Over the years, much progress has been made on the task of category-level semantic segmentation, particularly since the advent of Deep Convolutional Neural

Networks (CNNs) [Farabet et al., 2013; Long et al., 2015; Chen et al., 2015a]. Categorical labeling, however, fails to provide detailed annotations of individual objects, from which many applications could benefit. By contrast, instance-level semantic segmentation produces information about the identity, location, shape and class label of each individual object.

Instance-level semantic segmentation, which aims at jointly detecting, segmenting and classifying every individual object in an image, has recently become a core challenge in scene understanding [Cordts et al., 2016; Lin et al., 2014; Everingham et al.]. Unlike its category-level counterpart, instance segmentation provides detailed information about the location, shape and number of individual objects. As such, it has many applications in diverse areas, such as autonomous driving [Zhang et al., 2015a], personal robotics [Gupta et al., 2014] and plant analytics [Scharr et al., 2014].

## 6.2 Limitation of Existing Approaches

**Object detection based instance segmentation:** To simplify this challenging task, most existing methods first rely on detecting individual objects, for which a detailed segmentation is then produced. The early instances of this approach [Tighe et al., 2014; He and Gould, 2014] typically used pre-trained class-specific object detectors. More recently, however, many methods have proposed to exploit generic object proposals [Arbelaez et al., 2014; Ren et al., 2015], and postpone the classification problem to later stages. In this context, [Hariharan et al., 2014] makes use of Fast-RCNN boxes [Girshick, 2015] and builds a multi-stage pipeline to extract features, classify and segment the object. This framework was improved by the development of Hypercolumn features [Hariharan et al., 2015] and the use of a fully convolutional network (FCN) to encode category-specific shape priors [Li et al., 2016]. In [Dai et al., 2016b], the Region Proposal Network of [Ren et al., 2015] was integrated into a multi-task network cascade (MNC) for instance semantic segmentation. Ultimately, all these methods suffer from the fact that they predict a binary mask within the bounding box proposals, which are typically inaccurate. By contrast, here, we introduce a boundary-aware OMN that lets us predict instance segmentations that go beyond the box’s spatial extent. We show that integrating this OMN in the MNC framework outperforms the state-of-the-art instance-level semantic segmentation techniques.

**Semantic segmentation based instance segmentation:** Deep watershed transformation (DWT) [Bai and Urtasun, 2017] is an instance segmentation technique that relies on pretrained semantic segmentation model. The idea is to generate an energy map of the image by applying the classical watershed transform on deep learned

---

feature map. Object instances are represented as basins in the energy map. An important limitation of this technique is that objects bisected by occlusions cannot be properly segmented.

**Box proposals based instance segmentation:** Existing approaches to multiclass instance segmentation typically rely on generic object proposals in the form of bounding boxes. These proposals can be learned [Hariharan et al., 2014; Li et al., 2016; Dai et al., 2016b] or sampled by sliding windows [Pinheiro et al., 2015; Dai et al., 2016a], and greatly facilitate the task of identifying the different instances, may they be from the same category or different ones. Object segmentation is then achieved by predicting a binary mask within each box proposal, which can then be classified into a semantic category. This approach to segmentation, however, makes these methods sensitive to the quality of the bounding boxes; they cannot recover from errors in the object proposal generation process, such as too small or shifted boxes.

**Segmentation proposals based instance segmentation:** Beyond instance-level semantic segmentation, many methods have been proposed to generate class-agnostic region proposals [Arbelaez et al., 2014; Uijlings et al., 2013; Krähenbühl and Koltun, 2014]. The most recent such approaches rely on deep architectures [Pinheiro et al., 2015, 2016]. In particular, the method of [Dai et al., 2016a], in which an FCN computes a small set of instance-sensitive score maps that are assembled into object segmentation proposals, was shown to effectively improve instance-level semantic segmentation when incorporated in the MNC framework.

**Proposal free segmentation:** Other methods have nonetheless proposed to bypass the object proposal step for instance-level segmentation. For instance, the Proposal-free Network (PFN) of [Liang et al., 2015] predicts the number of instances and, at each pixel, a semantic label and the location of its enclosing bounding box. The results of this approach, however, strongly depend on the accuracy of the predicted number of instances. By contrast, [Zhang et al., 2015b] proposed to identify the individual instances based on their depth ordering. This was further extended in [Zhang et al., 2015a] via a deep densely connected Markov Random Field. It is unclear, however, how this approach handles the case where multiple instances are at roughly identical depths. To overcome this, the recent work of [Uhrig et al., 2016] uses an FCN to jointly predict depth, semantics and an instance-based direction encoding. This information is then used to generate instances via a template matching procedure. Unfortunately, this process involves a series of independent modules, which cannot be optimized jointly, thus yielding a potentially suboptimal solution. Finally,

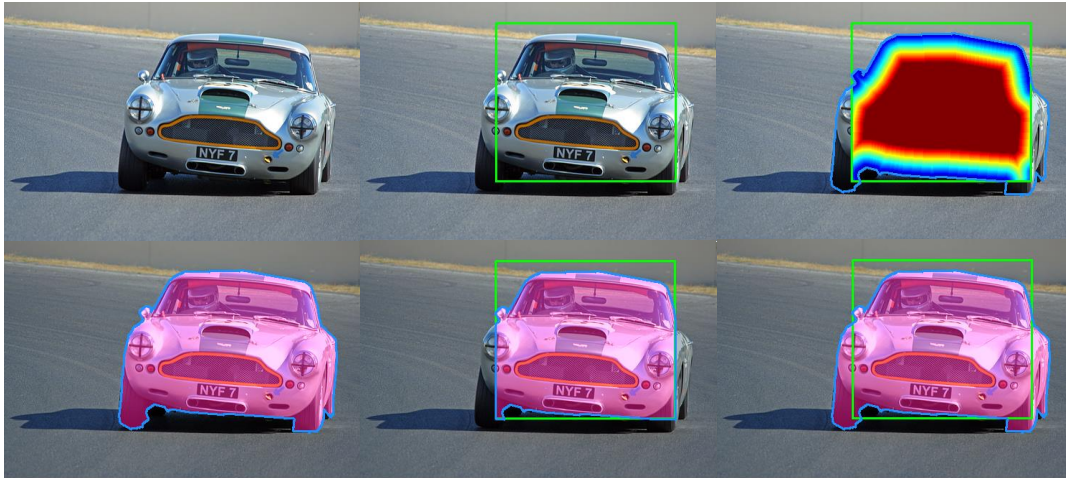


Figure 6.1: **Traditional instance segmentation vs our shape based representation.** **Left:** Original image and ground-truth segmentation. **Middle:** Given a bounding box, traditional methods directly predict a binary mask, whose extent is therefore limited to that of the box and thus suffers from box inaccuracies. **Right:** We represent the object segment with a multi-valued map encoding the truncated minimum distance to the object boundary. This can be converted into a mask that goes beyond the bounding box, which makes our approach robust to box errors.

in [Romera-Paredes and Torr, 2016], a recurrent neural network was proposed to segment an image instance-by-instance. This approach, however, essentially assumes that all the instances observed in the image belong to the same class.

### 6.2.1 Our Idea

In this chapter, we introduce a novel representation of object segments that is robust to errors in the bounding box proposals. To this end, we propose to model the shape of an object with a dense multi-valued map encoding, for every pixel in a box, its (truncated) minimum distance to the object boundary, or the fact that the pixel is outside the object. Object segmentation can then be achieved by converting this multi-valued map into a binary mask via the inverse distance transform [Borgefors, 1986; Kimmel et al., 1996]. In contrast to existing methods discussed above, and as illustrated in Fig. 6.1, the resulting mask is *not* restricted to lie inside the bounding box; even when the box covers only part of the object, the distances to the boundary in our representation may correspond to an object segment that goes beyond the box’s spatial extent.

To exploit our new object representation, we design an object mask network (OMN) that, for each box proposal, first predicts the corresponding pixel-wise multi-valued map, and then decodes it into the final binary mask, potentially going beyond

---

the box itself. In particular, we discretize the truncated distances and encode them using a binary vector. This translates the prediction of the multi-valued map to a pixel-wise labeling task, for which deep networks are highly effective, and facilitates decoding the map into a mask. The first module of our network then produces multiple probability maps, each of which indicates the activation of one particular bit in this vector. We then pass these probability maps into a new residual-deconvolution network module that generates the final binary mask. Thanks to the deconvolution layers, our output is not restricted to lie inside the box, and our OMN is fully differentiable.

To tackle instance-level semantic segmentation, we integrate our OMN into the Multitask Network Cascade framework of [Dai et al., 2016b], by replacing the original binary mask prediction module. As our OMN is fully differentiable, we can learn the resulting instance-level semantic segmentation network in an end-to-end manner. Altogether, this yields a *boundary-aware* instance segmentation (BAIS) network that is robust to noisy object proposals.

We demonstrate the effectiveness of our approach on PASCAL VOC 2012 [Everingham et al.] and the challenging CityScapes [Cordts et al., 2016] dataset. Our BAIS framework outperforms all the state-of-the-art methods on both datasets, by a considerable margin in the regime of high IOUs. Furthermore, an evaluation of our OMN on the task of object proposal generation on the PASCAL VOC 2012 dataset reveals that it achieves performance comparable to or even better than state-of-the-art methods, such as DeepMask [Pineiro et al., 2015] and SharpMask [Pineiro et al., 2016].

Our experiments demonstrate that our OMN produces segments of a quality comparable to or even higher than these state-of-the-art methods. Furthermore, by integrating it in a complete instance-level semantic segmentation network, we also outperform the state-of-the-art on this task.

## 6.3 Boundary-aware Segment Prediction

Our goal is to design an instance-level semantic segmentation method that is robust to the misalignment of initial bounding box proposals. To this end, we first introduce a novel object mask representation capable of capturing the overall shape or exact boundaries of an object. This representation, based on the distance transform, allows us to infer the complete shape of an object segment even when only partial information is available. We then construct a deep network that, given an input image, uses this representation to generate generic object segments that can go beyond the boundaries of initial bounding boxes.

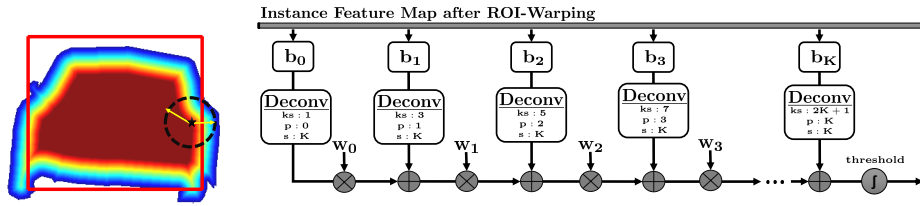


Figure 6.2: **Left:** Truncated distance transform. **Right:** Our deconvolution-based shape-decoding network. Each deconvolution has a specific kernel size ( $ks$ ), padding ( $p$ ) and stride ( $s$ ). Here,  $K$  represents the number of binary maps.

Below, we first describe our object mask representation and object mask network (OMN). In Section 6.4, we show how our network can be integrated in a Multistage Network Cascade [Dai et al., 2016b] to learn an instance-level semantic segmentation network in an end-to-end manner.

### 6.3.1 Boundary-aware Mask Representation

Given a window depicting a potentially partially-observed object, obtained from an image and a bounding box, we aim at producing a mask of the entire object. To this end, instead of directly inferring a binary mask, which would only represent the visible portion of the object, we propose to construct a pixel-wise, multi-valued map encoding the boundaries of the complete object by relying on the concept of distance transform [Borgefors, 1986]. In other words, the value at each pixel in our map represents either the distance to the nearest object boundary if the pixel is inside the object, or the fact that the pixel belongs to the background.

With varying window sizes and object shapes, the distance transform can produce a large range of different values, which would lead to a less invariant shape representation and complicate the training of our OMN in Section 6.3.2. Therefore, we normalize the windows to a common size and truncate the distance transform to obtain a restricted range of values. Specifically, let  $Q$  denote the set of pixels on the object boundary and outside the object. For every pixel  $p$  in the normalized window, we compute a truncated distance  $D(p)$  to  $Q$  as

$$D(p) = \min \left( \min_{\forall q \in Q} [d(p, q)], R \right), \quad (6.1)$$

where  $d(p, q)$  is the spatial, Euclidean distance between pixel  $p$  and  $q$ ,  $\lceil x \rceil$  returns the integer nearest to but larger than  $x$ , and  $R$  is the truncation threshold, i.e., the largest distance we want to represent. We then directly use  $D$  as our dense object representation. Fig. 6.2 (Left) illustrates such a dense map for one object.

As an object representation, the pixel-wise map described above as several advan-



tages over a binary mask that specifies the presence or absence of an object of interest at each pixel. First, the value at a pixel gives us information about the location of the object boundary, even if the pixel belongs to the interior of the object. As such, our representation is robust to partial occlusions arising from inaccurate bounding boxes. Second, since we have a distance value for every pixel, this representation is redundant, and thus robust to some degree of noise in the pixel-wise map. Importantly, predicting such a representation can be formulated as a pixel-wise labeling task, for which deep networks have proven highly effective.

To further facilitate this labeling task, we quantize the values in the pixel-wise map into  $K$  uniform bins. In other words, we encode the truncated distance for pixel  $p$  using a  $K$ -dimensional binary vector  $b(p)$  as

$$D(p) = \sum_{n=1}^K r_n \cdot b_n(p), \quad \sum_{n=1}^K b_n(p) = 1, \quad (6.2)$$

where  $r_n$  is the distance value corresponding to the  $n$ -th bin. By this one-hot encoding, we have now converted the multi-value pixel-wise map into a set of  $K$  binary pixel-wise maps. This allows us to translate the problem of predicting the dense map to a set of pixel-wise binary classification tasks, which are commonly, and often successfully, carried out by deep networks.

Given the dense pixel-wise map of an object segment (or truly  $K$  binary maps), we can recover the complete object mask approximately by applying an inverse distance transform. Specifically, we construct the object mask by associating each pixel with a binary disk of radius  $D(p)$ . We then compute the object mask  $M$  by taking the union of all the disks. Let  $T(p, r)$  denote the disk of radius  $r$  at pixel  $p$ . The object mask can then be expressed as

$$\begin{aligned} M &= \bigcup_p T(p, D(p)) = \bigcup_p T(p, \sum_{n=1}^K r_n \cdot b_n(p)) \\ &= \bigcup_{n=1}^K \bigcup_p T(p, r_n \cdot b_n(p)) = \bigcup_{n=1}^K T(\cdot, r_n) * B_n, \end{aligned} \quad (6.3)$$

where  $*$  denotes the convolution operator, and  $B_n$  is the binary pixel-wise map for the  $n$ -th bin. Note that we make use of the property of the one-hot encoding in the derivation. Interestingly, the resulting operation consists of a series of convolutions, which will again become convenient when working with deep networks.

The rightmost column in Fig. 6.1 illustrates the behavior of our representation. In the top image, the value at each pixel represents the truncated distance to the instance boundary inside the bounding box. Although it does not cover the entire

object, converting this dense map into a binary mask, yields the complete instance mask shown at the bottom.

### 6.3.2 Object Mask Network

We now turn to the problem of exploiting our boundary-aware representation to produce a mask for every object instance in an input image. To this end, we design a deep neural network that predicts  $K$  boundary-aware dense binary maps for every box in a set of bounding box proposals and decodes them into a full object mask via Eq. 6.3. In practice, we use the Region Proposal Network (RPN) [Ren et al., 2015] to generate the initial bounding box proposals. For each one of them, we perform a Region-of-Interest (RoI) warping of its features and pass the result to our network. This network consists of two modules described below.

Given the RoI warped features of one bounding box as input, the first module in our network predicts the  $K$  binary masks encoding our (approximate) truncated distance transform. Specifically, for the  $n$ -th binary mask, we use a fully connected layer with a sigmoid activation function to predict a pixel-wise probability map that approximates  $B_n$ .

Given the  $K$  probability maps, we design a new residual deconvolution network module to decode them into a binary object mask. Our network structure is based on the observation that the morphology operator in Eq. 6.3 can be implemented as a series of deconvolutions with fixed weights but different kernel and padding sizes, as illustrated in the Fig. 6.2 (Right). We then approximate the union operator with a series of weighted summation layers followed by a sigmoid activation function. The weights in the summation layers are learned during training. To accommodate for different sizes of the deconvolution filters, we upsample the output of the deconvolution corresponding to a smaller value of  $r_n$  in the network before each weighted summation. We use a fixed stride value of  $K$  for this purpose.

Our OMN is fully differentiable, and the output of the decoding module can be directly compared to the ground truth at a high resolution using a cross-entropy loss. This allows us to train our OMN in an end-to-end fashion, including the initial RPN, or, as discussed in Section 6.4, to integrate it with a classification module to perform instance-level semantic segmentation.

## 6.4 Learning Instance Segmentation

We now introduce our approach to tackling instance-level semantic segmentation with our OMN. To this end, we construct a Boundary-Aware Instance Segmentation (BAIS) network by integrating our object mask network into a Multistage Network

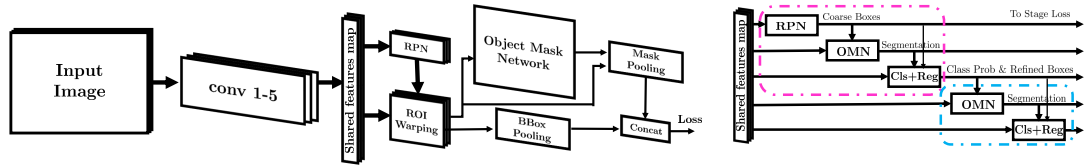


Figure 6.3: **Left:** Detailed architecture of our boundary-aware instance segmentation network. An input image first goes through a series of convolutional layers, followed by an RPN to generate bounding box proposals. After RoI warping, each proposal passes through our OMN to obtain a binary mask that can go beyond the box’s spatial extent. Mask features are then extracted and used in conjunction with bounding-box features for classification purpose. During training, our model makes use of a multi-task loss encoding, bounding box, segmentation and classification errors. **Right:** 5-stage BAIS network. The first three stages correspond to the model on the left. The five-stage model then concatenates an additional OMN and classification module to these three stages. The second OMN takes as input the classification score and refined box from the previous stage, and outputs a new segmentation with a new score obtained via the second classification module. The weights of the OMN and classification modules in both stages are shared.

Cascade (MNC) [Dai et al., 2016b]. Since our OMN module is differentiable, we can train the entire instance segmentation network in an end-to-end manner. Below, we first describe the overall network architecture, and then discuss our end-to-end training procedure and inference at test time.

#### 6.4.1 Boundary-aware Instance Segmentation Network

Our boundary-aware instance segmentation network follows a structure similar to that of the MNC. Specifically, our segmentation network consists of three sub-networks, corresponding to the tasks of bounding box proposal generation, object mask prediction and object classification. The first module consists of a deep CNN (in practice, the VGG16 [Simonyan and Zisserman, 2015] architecture) to extract a feature representation from an input image, followed by an RPN [Ren et al., 2015], which generates a set of bounding box proposals. After RoI warping, we pass each proposal through our OMN to produce a segment mask. Finally, as in the original MNC network, mask features are computed by using the predicted mask in a feature masking layer and concatenated with bounding box features. The resulting representation is then fed into the third sub-network, which consists of a single fully-connected layer for classification and bounding-box regression. The overall architecture of our BAIS network is illustrated in Fig. 6.3.

**Multi-stage Boundary-aware Segmentation Network.** Following the strategy of [Dai et al., 2016b], we extend the BAIS network described above, which can be thought of

---

as a 3-stage cascade, to a 5-stage cascade. The idea, here, is to refine the initial set of bounding box proposals, and thus the predicted segments, based on the output of our OMN. As illustrated in Fig. 6.3 (Right), the first three stages consist of the model described above, that is the VGG16 convolutional layers, RPN, OMN, classification module and bounding-box prediction. We then make use of the prediction offset generated by the bounding-box regression part of the third stage to refine the initial boxes. These new boxes act as input, via RoI warping, to the fourth-stage, which corresponds to a second OMN. Its output is then used in the last stage in conjunction with the refined boxes for classification purpose. In this 5-stage cascade, the weights of the two OMN and of the two classification modules are shared.

### 6.4.2 Network Learning and Inference

Our BAIS network is fully differentiable, and we therefore train it in an end-to-end manner. To this end, we use a multi-task loss function to account for bounding box, object mask and classification errors. Specifically, we use the softmax loss for the RPN and for classification, and the binary cross-entropy loss for the OMN. In our five-stage cascade, the bounding box and mask losses are computed after the third and fifth stages, and we use the smooth  $L_1$  loss for bounding-box regression.

We minimize the resulting multi-task, multi-stage loss over all parameters jointly using stochastic gradient descent (SGD). Following [Dai et al., 2016b,a; Girshick, 2015], we rely on min-batches of 8 images. As in [Dai et al., 2016b; Ren et al., 2015; Girshick, 2015], we resize the images such that the shorter side has 600 pixels. The VGG16 network in our first module was pre-trained on ImageNet. The other weights are initialized randomly from a zero-mean Gaussian distribution with std 0.01. We then train our model for 20k iterations with a learning rate of 0.001, and 5k iterations with a reduced learning rate of 0.0001.

The first module in our network first generates  $\sim 12k$  bounding boxes, which are pruned via non-maximum suppression (NMS). As in [Dai et al., 2016b], we use an NMS threshold of 0.7, and finally keep the top 300 bounding box proposals. In our OMN, we use  $K = 5$  probability maps to encode the (approximate) truncated distance transform. After decoding these maps via Eq. 6.3, we make use of a threshold of 0.4 to obtain a binary mask. This mask is then used to pool the features, and we finally obtain the semantic label via the classification module.

At test time, our BAIS network takes an input image and first computes the convolutional feature maps. The RPN module then generates 300 bounding box proposals and our OMN module predicts the corresponding object masks. These masks are categorized according to the class scores and a class-specific non-maximum suppression is applied with an IoU threshold of 0.5. Finally, we apply the in-mask voting

---

scheme of [Dai et al., 2016b] to each category independently to further refine the instance segmentations.

## 6.5 Experiments

In this section, we demonstrate the effectiveness of our method on instance-level semantic segmentation and segment proposal generation. We first discuss the former, which is the main focus of this work, and then turn to the latter. In both cases, we compare our approach to the state-of-the-art methods in each task.

**Datasets and setup.** To evaluate our approach, we make use of two challenging, standard datasets with multiple instances from a variety of object classes, *i.e.*, Pascal VOC 2012 and Cityscapes.

The Pascal VOC 2012 dataset [Everingham et al.] comprises 20 object classes with instance-level ground-truth annotations for 5623 training images and 5732 validation images. We used the instance segmentations of [Hariharan et al., 2011] for training and validation. We used all the training images to learn our model, but, following the protocols used in [Hariharan et al., 2014, 2015; Dai et al., 2015, 2016b,a], used only the validation dataset for evaluation. Following standard practice, we report the mean Average Precision (mAP) using IoU thresholds of 0.5 and 0.7 for instance semantic segmentation, and the Average Recall (AR) for different number and sizes of boxes for segment proposal generation.

The Cityscapes dataset [Cordts et al., 2016] consists of 9 object categories for instance-level semantic labeling. This dataset is very challenging since each image can contain a much larger number of instances of each class than in Pascal VOC, most of which are very small. It comprises 2975 training images from 18 cities, 500 validation images from 3 cities and 1525 test images from 6 cities. We only used the training dataset for training, and the test dataset to evaluate our method’s performance on the online test-server. Following the Cityscapes dataset guidelines, we computed the average precision (AP) for each class by averaging it across a range of overlap thresholds. We report the mean average precision (mAP) using an IoU threshold of 0.5, as well as mAP100m and mAP50m, where the evaluation is restricted to objects within 100 meters and 50 meters, respectively.

### 6.5.1 Instance-level Semantic Segmentation

We first present our results on the task of instance-level semantic segmentation, which is the main focus of this section. We report results on the two datasets discussed above. In both cases, we restricted the number of proposals to 300. For

our 5-stage models, this means 300 after the first RPN and 300 after bounding-box refinement.

### 6.5.1.1 Results on VOC 2012

Let us first compare the results of our Boundary-aware Instance Segmentation (BAIS) network with the state-of-the-art approaches on Pascal VOC 2012. These baselines include the SDS framework of [Hariharan et al., 2014], the Hypercolumn representation of [Hariharan et al., 2015], the InstanceFCN method of [Dai et al., 2016a] and the MNC framework of [Dai et al., 2016b]. In addition to this, we also report the results obtained by a Python re-implementation of the method in [Dai et al., 2016b], which we refer to as MNC-new. The results of this comparison are provided in Table 6.1. Note that our approach outperforms all the baselines, by a considerable margin in the case of a high IOU threshold. Note also that our approach is competitive in terms of runtime. Importantly, the comparison with BAIS-inside BBox, which restricts our masks to the spatial extent of the bounding boxes clearly evidences the importance of allowing the masks to go beyond the boxes’ extent.

Following the evaluation of MNC in [Dai et al., 2016b], we also study the influence of the number of stages in our model. We therefore learned different versions of our model using either our three-stage or five-stage cascade. At test time, because of parameter sharing across the stages, both versions are tested following a 5-stage procedure. The results of these different training strategies, for both MNC and our approach, are shown in Table 6.2. Note that, while our model trained with five-stages achieves the best results, our three-stage model still outperforms the two MNC baselines.

A detailed comparison with MNC [Dai et al., 2016b] including results for all the

VOC 2012 (val)	mAP (0.5)	mAP (0.7)	time/img (s)
SDS [Hariharan et al., 2014]	49.7	25.3	48
PFN [Liang et al., 2015]	58.7	42.5	~ 1
Hypercolumn [Hariharan et al., 2015]	60.0	40.4	>80
InstanceFCN [Dai et al., 2016a]	61.5	43.0	1.50
MNC [Dai et al., 2016b]	63.5	41.5	0.36
MNC-new	65.01	46.23	0.42
BAIS - insideBBox (ours)	64.97	44.58	0.75
BAIS - full (ours)	<b>65.69</b>	<b>48.30</b>	0.78

Table 6.1: **Instance-level semantic segmentation on Pascal VOC 2012.** Comparison of our method with state-of-the-art baselines. The results of [Hariharan et al., 2014, 2015] are reproduced from [Dai et al., 2016b].

VOC 2012 (val)	Training	Testing	mAP (0.5)	mAP (0.7)
MNC [Dai et al., 2016b]	3 stage	5 stage	62.6	-
MNC-new	5 stage	5 stage	65.01	46.23
BAIS - full (ours)	3 stage	5 stage	65.51	47.13
BAIS - full (ours)	5 stage	5 stage	<b>65.69</b>	<b>48.30</b>

Table 6.2: **Influence of the number of stages during training.** Whether trained using 3 stages or 5, our approach outperforms both MNC baselines.

classes is provided in the supplementary material.

### 6.5.1.2 Results on Cityscapes

We now turn to the Cityscapes dataset. In Table 6.3, we first report the results obtained from the online evaluation server on the test data, which is not publicly available. Note that our approach outperforms all the baselines (with an exception of improved DWT with PSPNet [Bai and Urtasun, 2017]) significantly on all the metrics. In Table 6.4, we provide a detailed comparison of our approach and the best performing baseline (DWT) in terms of AP(50%). Note that we outperform this method on average precision at 50%.

Additionally, we also compare our approach with MNC-new on the validation data. In this case, both models were trained using the training data only. For MCN, we used the same image size, RPN batch size, learning rate and number of iterations as for our model. Both models were trained using 5 stages. Table 6.5 show that, again, our model outperforms this baseline, thus demonstrating the benefits of allowing the masks to go beyond the box proposals.

In Fig. 6.4, we provide some qualitative results of our approach on Cityscapes. Note that we obtain detailed and accurate segmentation, even in the presence of

	Cityscapes (test)	AP	AP (50%)	AP (100m)	AP (50m)
Instance-level Segmentation of Vehicles by Deep Contours [Jan van den Brand, 2016]		2.3	3.7	3.9	4.9
R-CNN + MCG convex hull [Cordts et al., 2016]		4.6	12.9	7.7	10.3
Pixel-level Encoding for Instance Segmentation [Uhrig et al., 2016]		8.9	21.1	15.3	16.7
RecAttend [Ren and Zemel, 2017]		9.5	18.9	16.8	20.9
InstanceCut [Kirillov et al., 2017]		13.0	27.9	22.1	26.1
DWT [Bai and Urtasun, 2017]		15.6	30.0	26.2	31.8
	BAIS - full (ours)	17.4	<b>36.7</b>	29.3	34.0
	DWT + PSPNet [Bai and Urtasun, 2017]	<b>19.4</b>	35.3	<b>31.4</b>	<b>36.8</b>

Table 6.3: **Instance-level semantic segmentation on Cityscapes.** We compare our method with the state-of-the-art baselines on the Cityscapes test set. These results were obtained from the online evaluation server.

Cityscapes (test)	person	rider	car	truck	bus	train	motorcycle	bicycle	AP (50%)
DWT with PSPNet	33.98	36.87	48.50	<b>31.28</b>	40.06	36.23	<b>32.92</b>	22.89	35.34
BAIS - full (ours)	<b>34.04</b>	<b>40.36</b>	<b>54.72</b>	27.23	<b>40.11</b>	<b>38.86</b>	32.24	<b>26.03</b>	<b>36.70</b>

Table 6.4: **Detailed comparison with DTW [Bai and Urtasun, 2017] at AP(50%).** Note that our approach outperforms this baseline on all the classes except truck and motorcycle for the Cityscapes test dataset.

Cityscapes (val)	IoU	person	rider	car	truck	bus	train	motorcycle	bicycle	mAP
MNC-new	0.5	23.25	25.19	43.26	31.65	50.99	42.51	14.00	17.53	31.05
BAIS - full (ours)	0.5	23.30	25.67	43.19	33.01	54.36	44.87	15.95	18.84	<b>32.40</b>
MNC-new	0.7	9.09	1.86	34.81	24.46	39.08	33.33	1.98	4.55	18.64
BAIS - full (ours)	0.7	9.09	2.53	35.05	25.75	39.35	33.04	2.73	5.30	<b>19.10</b>

Table 6.5: **Comparison with MNC-new on the Cityscapes validation data.** Note that our approach outperforms this baseline, thus showing the importance of allowing the masks to go beyond the box proposals.



Figure 6.4: **Qualitative results on Cityscapes** From left to right, we show the input image, our instance level segmentations and the segmentations projected onto the image with class labels. Note that our segmentations are accurate despite the presence of many instances.



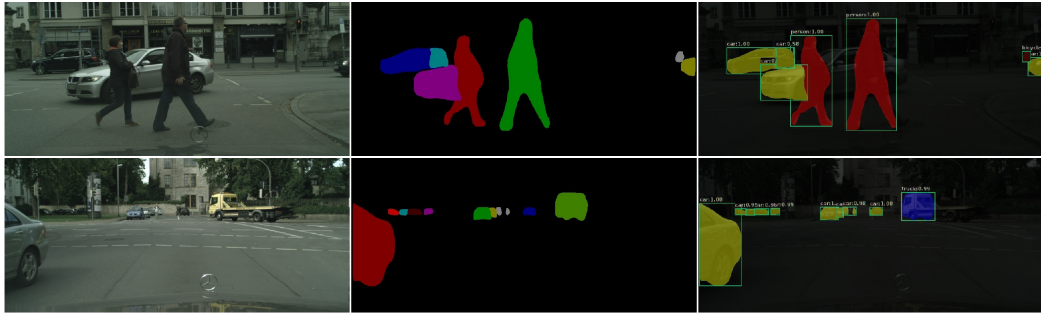


Figure 6.5: **Failure cases.** The typical failures of our approach correspond to cases where one instance is broken into multiple ones.

many instances in the same image. Some failure cases are shown in Fig. 6.5. These failures typically correspond to one instance being broken into several ones.

### 6.5.2 Segment Proposal Generation

As a second set of experiments, we evaluate the effectiveness of our object mask network (OMN) at generating high-quality segment proposals. To this end, we made use of the 5732 Pascal VOC 2012 validation images with ground-truth from [Hariharan et al., 2011], and compare our approach with the state-of-the-art segmentation proposal generation methods according to the criteria of [Hariharan et al., 2014; Lin et al., 2014]. In particular, we report the results of MCG [Arbelaez et al., 2014], DeepMask [Pinheiro et al., 2015] and Sharp-Mask [Pinheiro et al., 2016] using the publicly available pre-computed segmentation proposals. We also report the results of MNC by reproducing them from [Dai et al., 2016b], since these values were slightly better than those obtained from the publicly available segments. For our method, since the masks extend beyond the bounding box, the scores coming from the RPN, which correspond to the boxes, are ill-suited. We therefore learned a scoring function to re-rank our proposals. For the comparison to be fair, we also learned a similar scoring function for the MNC proposals. We refer to this baseline as MNC+score.

The results of our comparison are provided in Table 6.6. Our approach yields state-of-the-art results when considering 10 or 100 proposals. For 1000, SharpMask yields slightly better AR than us. Note, however, that, in practice, it is not always possible to handle 1000 proposal in later processing stages, and many instance-level segmentation methods only consider 100 or 300, which is the regime where our approach performs best. In Fig. 6.6, we report recall vs IOU threshold for all methods. Interestingly, even for 1000 segmentation proposals, our results outperform most of the baselines at high IOU thresholds. We refer the reader to the supplementary

	PASCAL VOC 2012	AR@10	AR@100	AR@1000
Selective Search [Uijlings et al., 2013]		7.0	23.5	43.3
MCG [Arbelaez et al., 2014]		18.9	36.8	49.5
Deep-Mask [Pinheiro et al., 2015]		30.3	45.0	52.6
Sharp-Mask [Pinheiro et al., 2016]		33.3	48.8	<b>56.5</b>
MNC [Dai et al., 2016b]		33.4	48.5	53.8
InstanceFCN [Dai et al., 2016a]		38.9	49.7	52.6
MNC+score		45.7	49.1	52.5
OMN (ours)		<b>47.8</b>	<b>51.8</b>	54.7

Table 6.6: **Evaluation of our OMN on the PASCAL VOC 2012 validation set.** We compare our method with state-of-the-art segmentation proposal baselines according to the criteria of [Hariharan et al., 2014; Lin et al., 2014]. Note that our approach outperforms the state-of-the-art methods for the top 10 and 100 segmentation proposals, which correspond to the most common scenarios when later processing is involved, e.g., instance level segmentation.

material for a comparison of the methods across different object sizes.

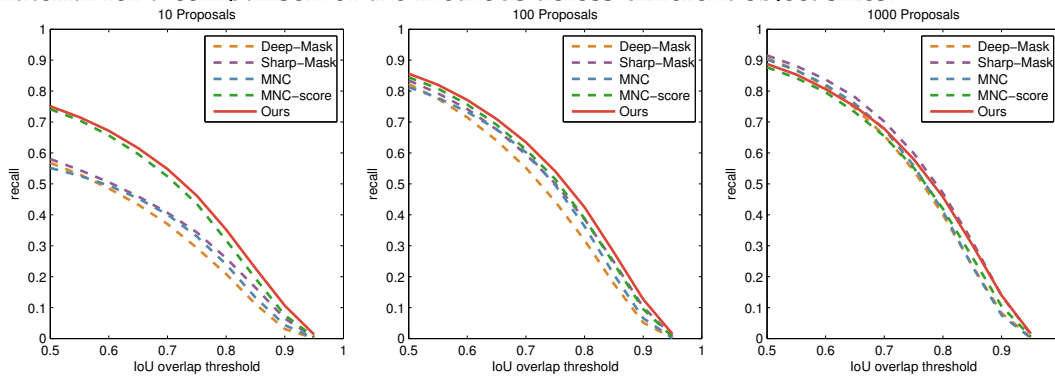


Figure 6.6: **Recall vs. IoU threshold on Pascal VOC 2012.** The curves were generated using the highest-scoring 10, 100 and 1000 segmentation proposals, respectively. In each plot, the solid line corresponds to our OMN results. Note that we outperforms the baselines when using the top 10 and 100 proposals. For 1000, our approach still yields state-of-the-art results at high IoU thresholds.

VOC 2012	IoU	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	mAP
MNC-new	0.5	80.1	67.6	71.7	49.6	47.4	80.1	70.2	86.6	37.1	62.9	37.0	83.1	75.2	75.6	76.1	40.3	70.1	47.8	76.4	65.6	65.0
SAIS (ours)	0.5	80.1	68.3	72.2	47.0	49.2	79.8	70.8	86.0	39.0	65.8	39.6	84.8	73.7	75.9	76.9	41.1	70.9	47.7	78.6	66.2	<b>65.7</b>
MNC-new	0.7	58.1	33.9	49.5	32.9	34.6	71.6	53.7	75.4	17.9	48.2	18.5	66.9	43.4	48.9	48.1	19.7	49.6	34.1	64.6	55.5	46.2
SAIS (ours)	0.7	60.2	39.7	52.1	31.4	36.0	71.8	56.8	75.7	20.4	51.5	21.8	69.3	47.7	52.8	51.0	21.3	51.6	35.3	63.9	55.7	<b>48.3</b>

Table 6.7: **Detailed Evaluation on the Pascal VOC 2012 validation set.** We report results with mask level IoU thresholds of 0.5 and 0.7. Note that our method outperforms MNC [Dai et al., 2016b] for most classes and achieves an improvement of 2.1% over MNC at a 0.7% IoU threshold

## 6.6 Ablation Studies

### 6.6.1 Detailed Comparisons with MNC on Pascal VOC 2012

We now focus on the detailed comparison with MNC [Dai et al., 2016b], which is the method most closely related to ours and achieves the best performance after us. In Table 6.7, we provide the detailed evaluation over all the classes of the Pascal VOC 2012 dataset [Everingham et al.] using IoU thresholds of 0.5 and 0.7, respectively. Note that our method outperforms this baseline for most classes, with a particularly large margin for an IoU of 0.7.

### 6.6.2 Object Mask Network: Comparison with MNC and DeepMask for Different Object Sizes

We further analyze the quality of the segmentation proposals for different sizes of objects. Following the criteria of [Lin et al., 2014], all object instances are categorized into small, medium, or large. The average recall is computed for the top 100 proposals using the mask level intersection over union. In Table 6.8, we provide the results of this evaluation on the Pascal VOC 2012 validation set and a comparison with the MNC [Dai et al., 2016b], DeepMask [Pineiro et al., 2015] and SharpMask [Pineiro et al., 2016] proposals. While SharpMask achieves the best performance on small objects, our approach yields competitive results on medium objects and clearly outperforms all these baselines on large ones.

### 6.6.3 Qualitative Comparison with MNC

In Table 6.9, we compare our shape-aware instance segmentation results with those obtained by MNC. These results again evidence the better quality of our instance-level semantic segmentations.

PASCAL VOC 2012	AR@Small	AR@Medium	AR@Large
DeepMask [Pineiro et al., 2015]	24.6	43.9	50.5
SharpMask [Pineiro et al., 2016]	<b>25.8</b>	<b>47.6</b>	55.3
MNC+score	19.4	44.0	60.7
OMN (ours)	20.3	45.9	<b>63.4</b>

Table 6.8: **Object size analysis on the Pascal VOC 2012 validation set.** We compare our method with state-of-the-art segmentation proposal baselines according to the criteria of [Lin et al., 2014]. The AR for small, medium and large objects were computed for 100 proposals. Note that our object mask network outperforms the state-of-the-art baselines for large objects by a significant margin.

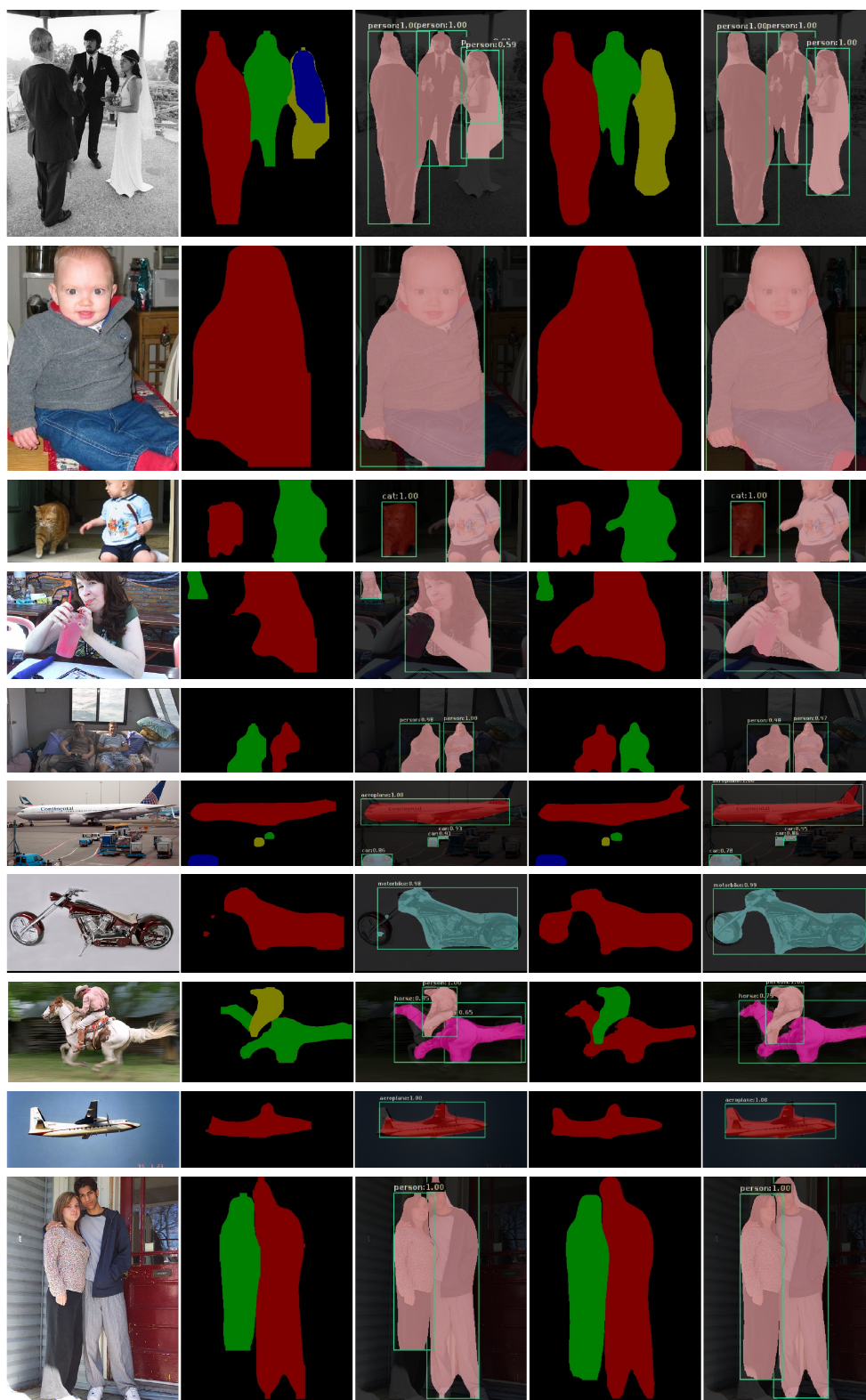


Table 6.9: **Qualitative results on Pascal VOC 2012.** From left to right, we show: the original image, the segmentations of MNC, these results projected on the image, our segmentations, our results projected on the image. Note the better quality of our results.

---

## 6.7 Conclusion

In this chapter, we introduced a distance transform-based mask representation that allows us to predict instance segmentations beyond the limits of initial bounding boxes. We also shown how to infer and decode this representation with a fully-differential Object Mask Network (OMN) relying on a residual-deconvolutional architecture. We employed this OMN to develop a Boundary-Aware Instance Segmentation (BAIS) network. Our experiments on Pascal VOC 2012 and Cityscapes demonstrated that our BAIS network outperforms the state-of-the-art instance-level semantic segmentation methods. Note that our instance-level object segmentation results can be further refined by using semantic segmentation pipelines. However, a region-of-interest alignment layer is required to have a pixel-level one-to-one correspondences as described in [He et al., 2017].



---

# Conclusions and Future Work

---

In this thesis, we have addressed structured learning for challenging large-scale multi-image multi-task datasets. In this context, we have presented efficient and effective deep structured models for context-aware object detection, co-localization and instance-level semantic segmentation. In this concluding chapter, we summarize the research contributions of this dissertation and discuss the important directions of future work.

## 7.1 Contributions

The contributions of this thesis are four-fold: first, we have presented a novel algorithm for class-specific similarity to jointly process large-scale datasets for efficient object co-detection (Chapter 3); second, we have extended our class-specific method to multi-class similarity learning and have further leveraged deep features and geometric scene layout (Chapter 4); third, we have introduced a class-agnostic formulation of object similarity for overcoming the poor localization performance of object detectors and have provided a solution that allowed us to jointly learn context within a deep structured network (Chapter 5); fourth, we have developed a boundary-aware instance segmentation approach that is not limited to predicting segmentation masks inside a bounding-box (Chapter 6). Below, we summarize the contributions in each chapter.

**Chapter 3.** In this chapter, we have presented a principled formulation of object co-detection. More specifically, we have introduced a fully-connected conditional random field (CRF) whose vertices represent object candidates and edges encode the object similarity via simple, yet effective pairwise potentials. In this context, we have designed a weighted mixture of Gaussian kernels for class-specific object similarity, and formulated kernel weights estimation as a least-squares regression problem. We have demonstrated that its solution can therefore be obtained in closed-form. It also allowed us to efficiently perform inference in large graphs using an approximate mean-field method with high-dimensional Gaussian filtering.

As a result, our approach effectively leverages the information in multiple images to improve detection accuracy, thus has outperformed existing detection and co-detection techniques on benchmark datasets. Furthermore, we have shown that the learned similarity is robust to noise because the variance of our algorithm is less than 1% in both the random pair and stereo pair experiments. We therefore believe that our class-specific similarity learning approach is the method of choice for scalable single-class object co-detection.

**Chapter 4.** In this chapter, we have introduced an extension of our class-specific method to handle multiple object classes simultaneously in large-scale image datasets. More specifically, we have presented two algorithms for structural boosting that are based on high-dimensional, rich representations learned using a deep convolutional neural network. In particular, our max-margin and direct-loss structural boosting algorithms have enabled us to learn the most suitable features that best encode pairwise similarity relationships within our CRF framework.

Furthermore, the resulting unified framework has allowed us to learn the contextual relationships within a CRF framework, thus leading to superior object (co-)detection results. It has also guaranteed  $O(n * t)$  time and space complexity which is linear in number of estimated object candidates. Moreover, our experiments have demonstrated the importance of learning rich similarity measures to account for the contextual relations across object classes and instances. However, our empirical error analysis has shown a need to learn better quality proposals for improving object localization performance.

**Chapter 5.** In this chapter, we have addressed the localization issue of object (co-)detectors. In this context, we have presented a class-agnostic model to jointly learn instance similarity to improve the average recall of object proposal generation methods. Specifically, we have presented an efficient object proposal co-generation technique which leverages the collective power of multiple images. We have designed a deep neural network layer that takes unary and pairwise features as input, builds a fully-connected CRF and produces mean-field marginals as output. This layer has simplified end-to-end learning and enabled us to backpropagate the gradient through entire network by unrolling the mean-field iterations of CRF inference.

Furthermore, empirical experiments have demonstrated the benefit of our approach over the state-of-the-art methods that generate object proposals individually. We therefore believe that our method provides high-quality object proposals and learns better contextual features for further detection.



---

**Chapter 6.** In this chapter, we have presented a multi-task strategy to jointly learn object detection, localization and instance-level semantic segmentation in a single network. In particular, the goal was, analogously to object co-detection techniques, to improve generalization performance. More specifically, we have tackled the instance segmentation problem and have introduced a novel representation based on the distance transform of the object masks. To this end, we have designed a new residual-deconvolution architecture that infers such a representation and decodes it into the final binary object mask.

Furthermore, our quantitative and qualitative experiments have demonstrated that the predicted masks can go beyond the scope of the bounding boxes and are thus robust to inaccurate object candidates. Finally, it has shown the superiority of the proposed approach over the state-of-the-art instance-level semantic segmentation methods, and that multiple tasks can benefit from each other.

All in all, we have addressed structured learning for challenging large-scale multi-image multi-task datasets and presented robust deep structured models for object co-detection, co-localization and boundary-aware instance segmentation. In general, the co-detection method presented in Chapter 4 can be used in conjunction with the method of Chapter 5 to get reliable end-to-end object co-detection with proposal co-generation, and the methods of Chapter 4, 5 and 6 can also be combined to leverage multiple images and multiple tasks jointly. However, it will require more GPU memory to train these models.

## 7.2 Future Directions

We conclude this dissertation with some suggestions for future research.

**Object co-detection in videos:** All the methods described in this thesis rely on large-scale image datasets and can be further used to exploit context in image sequences or videos. Recently, the Imagenet dataset [Russakovsky et al., 2015] has introduced an object detection challenge from videos with 30 object classes. This dataset consists of 3862, 555 and 937 snippets for training, validation and testing set, respectively. The Youtube-BB dataset [Real et al., 2017] also provides 380k densely labeled video snippets of 19 seconds. We believe that these datasets will be very useful to exploit spatio-temporal consistencies and to improve object detection and co-detection performance. The method discussed in Chapter 4 can be further extended to videos. Recently, [Kundu et al., 2016] has proposed a technique to improve semantic video segmentation by incorporating spatio-temporal regularization. We

believe that establishing temporal correspondences in our method will be very useful for robust object co-detection in videos.

**Video-based weakly supervised (co-)detection:** Another interesting field of further research is weakly supervised learning. Videos provide a natural source of structured data, but a major limitation is that it is not densely labeled. In the Cityscapes dataset [Cordts et al., 2016], annotations are only provided for every 20<sup>th</sup> frame in the video. This makes it very useful to learn weakly supervised models and transfer labels from the nearby densely labeled frames. The methods proposed in Chapter 4 and 5 can be further extended to handle weakly labeled image sequences. We believe that the optical flow can be used to propagate true labels to nearby unlabeled video frames. Furthermore, the method proposed by [Jampani et al., 2017] can be used in conjunction with our technique to forward propagate structured information in an adaptive manner.

**Unsupervised object discovery:** It is very costly and not feasible to label each and every object in large amounts of data. Therefore, automatically learning from videos is an important task. However, it becomes very challenging in the absence of labeled data. Everyday, millions of videos are uploaded to the Internet without any label information (with an exception of some useful tags). These videos are very useful for automatically discovering and learning the most frequently occurring objects. We believe that our multi-image multi-class similarity method discussed in Chapter 4 and our class-agnostic similarity in Chapter 5 are a step in this direction. However, extending this to completely the unsupervised setting will require an enormous amount of effort. Indeed, a robust clustering technique is required and we refer the interested researchers to [Gholami and Pavlovic, 2017] for further details.

**3D scene layout:** As discussed in Chapter 6, multiple tasks when trained together can benefit from each other. In this direction, we have developed an instance-level semantic segmentation method that utilizes only 2D object bounding boxes. However, another task which has recently gained much popularity is to predict 3D bounding boxes. One interesting research direction along these lines is to extend instance-level semantic segmentation to predict 3D scene layout using stereo cameras. We believe that the method of [Zhai et al., 2017] is very useful as it learns an adaptive transformation to map image-level features into the 3D perspective. However, this can be extended to the object-level using our boundary-aware instance segmentation technique. This research may also prove very useful for autonomous driving.

---

# Bibliography

---

- ABADI, M.; AGARWAL, A.; BARHAM, P.; BREVDO, E.; CHEN, Z.; CITRO, C.; CORRADO, G. S.; DAVIS, A.; DEAN, J.; DEVIN, M.; GHEMAWAT, S.; GOODFELLOW, I.; HARP, A.; IRVING, G.; ISARD, M.; JIA, Y.; JOZEFOWICZ, R.; KAISER, L.; KUDLUR, M.; LEVENBERG, J.; MANÉ, D.; MONGA, R.; MOORE, S.; MURRAY, D.; OLAH, C.; SCHUSTER, M.; SHLENS, J.; STEINER, B.; SUTSKEVER, I.; TALWAR, K.; TUCKER, P.; VANHOUCHE, V.; VASUDEVAN, V.; VIÉGAS, F.; VINYALS, O.; WARDEN, P.; WATTENBERG, M.; WICKE, M.; YU, Y.; AND ZHENG, X., TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. <http://tensorflow.org/>. Software available from [tensorflow.org](http://tensorflow.org/). (cited on page 34)
- ADAMS, A.; BAEK, J.; AND DAVIS, M. A., Fast High-Dimensional Filtering Using the Permutohedral Lattice. *Computer Graphics Forum*, (2010). (cited on pages 23 and 24)
- ALEXE, B.; DESELAERS, T.; AND FERRARI, V., What is an object? In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 4, 76, and 96)
- ARBELAEZ, P.; PONT-TUSET, J.; BARRON, J. T.; MARQUES, F.; AND MALIK, J., Multiscale Combinatorial Grouping. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 116, 117, 129, and 130)
- BAEK, J.; ADAMS, A.; AND DOLSON, J., Lattice-Based High-Dimensional Gaussian Filtering and the Permutohedral Lattice. *Journal of Mathematical Imaging and Vision (JMIV)*, (2013). (cited on pages 58 and 74)
- BAI, M. AND URTASUN, R., Deep Watershed Transform for Instance Segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. <http://arxiv.org/abs/1611.08303>. (cited on pages xxv, 116, 127, and 128)
- BAO, S.; XIANG, Y.; AND SAVARESE, S., Object co-detection. In *European Conference on Computer Vision (ECCV)*. (cited on pages xvi, 3, 52, 53, 54, 55, 56, 57, 60, 61, 63, 64, 65, 71, 72, 96, and 97)
- BARINOVA, O.; LEMPITSKY, V.; AND KOHLI, P., On detection of multiple object instances using hough transforms. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, (2012). (cited on page 53)

- BELLMAN, R., 1957. *Dynamic Programming*. Princeton University Press. (cited on page 14)
- BENGIO, S., Large Scale Visual Semantic Extraction. In *Frontiers of Engineering*. (cited on pages 2 and 25)
- BENGIO, Y., Learning Deep Architectures for AI. *Foundations and Trends in Machine Learning*, (jan 2009). (cited on pages 25 and 28)
- BENGIO, Y., Deep Learning of Representations for Unsupervised and Transfer Learning. In *International Conference on Unsupervised and Transfer Learning workshop (UTLW)*. (cited on pages 25 and 28)
- BENGIO, Y.; COURVILLE, A.; AND VINCENT, P., Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 35, 8 (aug 2013). (cited on pages 25 and 28)
- BERGSTRA, J. AND BENGIO, Y., Random Search for Hyper-parameter Optimization. *Journal of Machine Learning Research (JMLR)*, 13, 1 (Feb. 2012). (cited on page 25)
- BLASCHKO, M. B. AND LAMPERT, C. H., Guest Editorial: Special Issue on Structured Prediction and Inference. *International Journal on Computer Vision (IJCV)*, 99, 3 (2012). (cited on page 2)
- BORGEFORS, G., Distance transformations in digital images. *Computer Vision, Graphics, and Image Processing (CVGIP)*, (1986). (cited on pages 118 and 120)
- BOYKOV, Y.; VEKSLER, O.; AND ZABIH, R., Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 23, 11 (2001). (cited on pages 58 and 68)
- CANDÈS, E. J.; LI, X.; MA, Y.; AND WRIGHT, J., Robust Principal Component Analysis. *Journal of the ACM*, 58 (jun 2011). (cited on pages 14 and 25)
- CARUANA, R., Learning Many Related Tasks at the Same Time with Backpropagation. In *Advances in Neural Information Processing Systems (NIPS)*. (cited on page 25)
- CHEN, L.-C.; PAPANDREOU, G.; KOKKINOS, I.; MURPHY, K.; AND YUILLE, A. L., Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs. In *International Conference on Learning Representations (ICLR)*. (cited on page 116)
- CHEN, L.-C.; SCHWING, A.; YUILLE, A.; AND URTASUN, R., Learning Deep Structured Models. In *International Conference on Machine Learning (ICML)*. JMLR Workshop and Conference Proceedings. (cited on page 97)

- 
- CHENG, M.; ZHANG, Z.; LIN, W.; AND TORR, P. H. S., BING: Binarized Normed Gradients for Objectness Estimation at 300fps. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 96 and 107)
- CHOI, M. J.; TORRALBA, A.; AND WILLSKY, A. S., A tree-based context model for object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, (2012). (cited on page 73)
- COLLOBERT, R.; BENGIO, S.; AND MARITHOZ, J., Torch: A Modular Machine Learning Software Library. (cited on page 34)
- CORDTS, M.; OMRAN, M.; RAMOS, S.; REHFELD, T.; ENZWEILER, M.; BENENSON, R.; FRANKE, U.; ROTH, S.; AND SCHIELE, B., The Cityscapes Dataset for Semantic Urban Scene Understanding. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages xvi, 1, 46, 116, 119, 125, 127, and 138)
- DAI, J.; HE, K.; LI, Y.; REN, S.; AND SUN, J., Instance-sensitive Fully Convolutional Networks. In *European Conference on Computer Vision (ECCV)*. (cited on pages 117, 124, 125, 126, and 130)
- DAI, J.; HE, K.; AND SUN, J., Convolutional feature masking for joint object and stuff segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 125)
- DAI, J.; HE, K.; AND SUN, J., Instance-aware Semantic Segmentation via Multi-task Network Cascades. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages xxv, 42, 43, 116, 117, 119, 120, 123, 124, 125, 126, 127, 129, 130, and 131)
- DALAL, N. AND TRIGGS, B., Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages xv, 4, 25, 26, 27, and 37)
- DESAI, C.; RAMANAN, D.; AND FOWLKES, C., Discriminative models for multi-class object layout. In *IEEE International Conference on Computer Vision (ICCV)*. (cited on pages xxiii, 52, 53, 72, 73, 75, 77, 78, 84, 85, 86, and 90)
- DICKINSON, S. J.; LEONARDIS, A.; SCHIELE, B.; AND TARR, M. J., 2009. *Object Categorization: Computer and Human Vision Perspectives*. Cambridge University Press. (cited on page 3)
- DUNTEMAN, G., 1989. *Principal Components Analysis*. Quantitative Applications in the Social Sciences. SAGE Publications. (cited on pages 14 and 25)

- ENDRES, I. AND HOIEM, D., Category independent object proposals. In *European Conference on Computer Vision (ECCV)*. (cited on pages 4 and 76)
- ESS, A.; LEIBE, B.; AND GOOL, L. V., Depth and Appearance for Mobile Scene Analysis. In *IEEE International Conference on Computer Vision (ICCV)*. (cited on pages xvi, 52, 56, and 60)
- The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. <http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html>. (cited on pages xv, 1, 5, 6, 75, 84, 116, 119, 125, and 131)
- EVERINGHAM, M.; VAN GOOL, L.; WILLIAMS, C. K. I.; WINN, J.; AND ZISSERMAN, A., The Pascal Visual Object Classes (VOC) Challenge. *International Journal on Computer Vision (IJCV)*, (jun 2010). (cited on pages xvi, 1, 3, 44, 46, 51, 75, 83, 84, 98, and 106)
- FARABET, C.; COUPRIE, C.; NAJMAN, L.; AND LECUN, Y., Learning hierarchical features for scene labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 35, 8 (2013). (cited on page 116)
- Discriminatively Trained Deformable Part Models, Release 4. <http://people.cs.uchicago.edu/~pff/latent-release4/>. (cited on pages 37, 60, 61, 64, and 65)
- FELZENSZWALB, P. F.; GIRSHICK, R. B.; MCALLESTER, D.; AND RAMANAN, D., Object Detection with Discriminatively Trained Part Based Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, (2010). (cited on pages 1, 3, 4, 37, 38, 51, 52, 55, 57, and 60)
- FISCHLER, M. A. AND ELSCHLAGER, R. A., The Representation and Matching of Pictorial Structures. *IEEE Transactions on Computers*, (1973). (cited on page 37)
- FRIEDMAN, N.; GEIGER, D.; AND GOLDSZMIDT, M., Bayesian Network Classifiers. *Machine Learning*, 29 (nov 1997). (cited on pages xv and 17)
- GALLEGUILLOS, C. AND BELONGIE, S., Context based object categorization: A critical survey. *Computer Vision and Image Understanding (CVIU)*, (2010). (cited on pages 52 and 73)
- GHOLAMI, B. AND PAVLOVIC, V., Probabilistic Temporal Subspace Clustering. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 138)
- GIRSHICK, R., Fast R-CNN. In *IEEE International Conference on Computer Vision (ICCV)*. (cited on pages xvi, xxiv, 1, 3, 40, 41, 42, 72, 87, 95, 96, 103, 111, 116, and 124)

- 
- GIRSHICK, R.; DONAHUE, J.; DARRELL, T.; AND MALIK, J., Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages xvi, xvii, xxiii, xxiv, 3, 39, 40, 41, 42, 43, 72, 74, 75, 76, 84, 85, 87, 89, 90, 91, 95, and 96)
- GÖNEN, M. AND ALPAYDIN, E., Multiple kernel learning algorithms. *Journal of Machine Learning Research (JMLR)*, 12 (2011). (cited on pages 54 and 74)
- GOODFELLOW, I. J.; LE, Q. V.; SAXE, A. M.; LEE, H.; AND NG, A. Y., Measuring Invariances in Deep Networks. In *Advances in Neural Information Processing Systems (NIPS)*. (cited on page 25)
- GUO, X.; LIU, D.; JOU, B.; ZHU, M.; CAI, A.; AND CHANG, S.-F., Robust Object Co-detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages xxiii, 3, 52, 53, 54, 60, 61, 64, 65, 71, 72, 85, 96, and 97)
- GUPTA, S.; GIRSHICK, R.; ARBELÁEZ, P.; AND MALIK, J., Learning rich features from RGB-D images for object detection and segmentation. In *European Conference on Computer Vision (ECCV)*. (cited on page 116)
- HARIHARAN, B.; ARBELAEZ, P.; BOURDEV, L.; MAJI, S.; AND MALIK, J., Semantic Contours from Inverse Detectors. In *IEEE International Conference on Computer Vision (ICCV)*. (cited on pages 125 and 129)
- HARIHARAN, B.; ARBELÁEZ, P.; GIRSHICK, R.; AND MALIK, J., Simultaneous Detection and Segmentation. In *European Conference on Computer Vision (ECCV)*. (cited on pages xxv, 95, 96, 116, 117, 125, 126, 129, and 130)
- HARIHARAN, B.; ARBELÁEZ, P. A.; GIRSHICK, R. B.; AND MALIK, J., Hypercolumns for object segmentation and fine-grained localization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages xxv, 116, 125, and 126)
- HAYDER, Z.; HE, X.; AND SALZMANN, M., Structural Kernel Learning for Large Scale Multiclass Object Co-Detection. In *IEEE International Conference on Computer Vision (ICCV)*. (cited on pages 8, 51, 72, and 95)
- HAYDER, Z.; HE, X.; AND SALZMANN, M., Learning to Co-Generate Object Proposals with a Deep Structured Network. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 9)
- HAYDER, Z.; HE, X.; AND SALZMANN, M., Boundary-aware Instance Segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 9)

- HAYDER, Z.; SALZMANN, M.; AND HE, X., Object co-detection via efficient inference in a fully-connected CRF. In *European Conference on Computer Vision (ECCV)*. (cited on pages 8, 51, 72, and 85)
- HE, K.; GKIOXARI, G.; DOLLÁR, P.; AND GIRSHICK, R., Mask R-CNN. *CoRR*, abs/1703.06870 (2017). (cited on page 133)
- HE, K.; ZHANG, X.; REN, S.; AND SUN, J., Deep Residual Learning for Image Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages xvi, 1, 32, and 37)
- HE, X. AND GOULD, S., An exemplar-based CRF for multi-instance object segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 116)
- HINTON, G. E., Training Products of Experts by Minimizing Contrastive Divergence. *Neural Computation*, (aug 2002). (cited on page 25)
- HINTON, G. E.; OSINDERO, S.; AND TEH, Y.-W., A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*, (jul 2006). (cited on page 25)
- HOIEM, D.; CHODPATHUMWAN, Y.; AND DAI, Q., Diagnosing error in object detectors. In *European Conference on Computer Vision (ECCV)*. (cited on pages xviii, xx, 87, 91, 111, and 113)
- HOIEM, D.; EFROS, A. A.; AND HEBERT, M., Putting Objects in Perspective. *International Journal on Computer Vision (IJCV)*, 80 (2008). (cited on pages 52, 53, and 73)
- HOSANG, J.; BENENSON, R.; DOLLAR, P.; AND SCHIELE, B., What Makes for Effective Detection Proposals? *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 38, 4 (apr 2016). (cited on pages xix, 96, 107, 108, 109, 110, and 111)
- HYVÄRINEN, A.; KARHUNEN, J.; AND OJA, E., 2004. *Independent component analysis*. John Wiley & Sons. (cited on pages 14 and 25)
- JAMPANI, V.; GADDE, R.; AND GEHLER, P. V., Video Propagation Networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 138)
- JAN VAN DEN BRAND, R. M., MATTHIAS OCHS, Instance-level Segmentation of Vehicles using Deep Contours. In *Workshop on Computer Vision Technologies for Smart Vehicle, in Asian Conference on Computer Vision (ACCV)*. (cited on page 127)
- JIA, Y.; SHELHAMER, E.; DONAHUE, J.; KARAYEV, S.; LONG, J.; GIRSHICK, R.; GUADARRAMA, S.; AND DARRELL, T., Caffe: Convolutional Architecture for Fast Feature



- 
- Embedding. In *ACM International Conference on Multimedia (MM)*. (cited on pages 29 and 34)
- JIMENEZ, L. O. AND LANDGREBE, D. A., Supervised classification in high-dimensional space: geometrical, statistical, and asymptotical properties of multivariate data. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 28 (1998). (cited on page 14)
- JORDAN, M. I.; GHAHRAMANI, Z.; JAAKKOLA, T. S.; AND SAUL, L. K., An Introduction to Variational Methods for Graphical Models. *Machine Learning*, (1999). (cited on page 22)
- KEDDEM, D.; TYREE, S.; WEINBERGER, K.; SHA, F.; AND LANCKRIET, G., Non-linear Metric Learning. In *Advances in Neural Information Processing Systems (NIPS)* (Eds. P. BARTLETT; F. PEREIRA; C. BURGESS; L. BOTTOU; AND K. WEINBERGER). (cited on page 25)
- KIMMEL, R.; KIRYATI, N.; AND BRUCKSTEIN, A. M., Sub-pixel distance maps and weighted distance transforms. *Journal of Mathematical Imaging and Vision (JMIV)*, (1996). (cited on page 118)
- KIRILLOV, A.; LEVINKOV, E.; ANDRES, B.; SAVCHYNSKYI, B.; AND ROTHER, C., Instance-Cut: from Edges to Instances with Multicut. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 127)
- KOLLER, D. AND FRIEDMAN, N., 2009. *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press. (cited on pages 16, 17, 18, 19, and 22)
- KOLMOGOROV, V. AND ZABIN, R., What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 26, 2 (2004). (cited on page 58)
- KRÄHENBÜHL, P. AND KOLTUN, V., Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials. In *Advances in Neural Information Processing Systems (NIPS)*. (cited on pages xxvii, 23, 24, 52, 54, 56, 57, 58, 74, 78, 79, 97, 98, 101, 102, and 105)
- KRÄHENBÜHL, P. AND KOLTUN, V., Parameter learning and convergent inference for dense random fields. In *International Conference on Machine Learning (ICML)*. (cited on pages 74, 82, 101, and 104)
- KRÄHENBÜHL, P. AND KOLTUN, V., Geodesic Object Proposals. In *European Conference on Computer Vision (ECCV)*. (cited on pages 96 and 117)

- KRIZHEVSKY, A.; SUTSKEVER, I.; AND HINTON, G. E., Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*. (cited on pages xv, xvi, 1, 35, 36, 40, 45, 51, 74, 84, and 86)
- KUNDU, A.; VINEET, V.; AND KOLTUN, V., Feature Space Optimization for Semantic Video Segmentation. In *CVPR*. (cited on page 137)
- LECUN, Y.; BENGIO, Y.; AND HINTON, G., Deep learning. *Nature*, (2015). (cited on page 28)
- LECUN, Y.; BOSER, B.; DENKER, J. S.; HENDERSON, D.; HOWARD, R. E.; HUBBARD, W.; AND JACKEL, L. D., Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation*, (dec 1989). (cited on pages 25 and 35)
- LECUN, Y.; BOTTOU, L.; BENGIO, Y.; AND HAFFNER, P., Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, (1998). (cited on pages 25, 26, and 97)
- LI, K.; HARIHARAN, B.; AND MALIK, J., Iterative Instance Segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 116 and 117)
- LIANG, X.; WEI, Y.; SHEN, X.; YANG, J.; LIN, L.; AND YAN, S., Proposal-free Network for Instance-level Object Segmentation. *CoRR*, abs/1509.02636 (2015). (cited on pages 117 and 126)
- LIN, T.; MAIRE, M.; BELONGIE, S.; HAYS, J.; PERONA, P.; RAMANAN, D.; DOLLÁR, P.; AND ZITNICK, C. L., Microsoft COCO: Common Objects in Context. In *European Conference on Computer Vision (ECCV)*. (cited on pages xxiv, xxv, 1, 45, 98, 107, 108, 110, 116, 129, 130, and 131)
- LONG, J.; SHELHAMER, E.; AND DARRELL, T., Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 116)
- LOWE, D. G., Object Recognition from Local Scale-Invariant Features. In *IEEE International Conference on Computer Vision (ICCV)*. (cited on pages 25, 26, and 51)
- MUNOZ, D.; BAGNELL, J. A.; VANDAPEL, N.; AND HEBERT, M., Contextual classification with functional max-margin markov networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 74 and 79)

- 
- NOWOZIN, S. AND LAMPERT, C. H., Structured Learning and Prediction in Computer Vision. *Foundations and Trends in Computer Graphics and Vision*, (2011). (cited on pages xv, 2, 16, 19, 20, 21, and 23)
- NVIDIA, GPU-Accelerated Deep Learning using NVIDIA cuDNN. <https://developer.nvidia.com/cudnn>. (cited on page 2)
- OJALA, T.; PIETIKAINEN, M.; AND MAENPAA, T., Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 24, 7 (2002). (cited on pages 25 and 60)
- OUYANG, W.; ZENG, X.; WANG, X.; QIU, S.; LUO, P.; TIAN, Y.; LI, H.; YANG, S.; WANG, Z.; LI, H.; WANG, K.; YAN, J.; LOY, C.-C.; TANG, X.; UNDEFINED; UNDEFINED; UNDEFINED; AND UNDEFINED, DeepID-Net: Object Detection with Deformable Part Based Convolutional Neural Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, (2017). (cited on page 74)
- PARAMESWARAN, S. AND WEINBERGER, K., Large Margin Multi-Task Metric Learning. In *Advances in Neural Information Processing Systems (NIPS)* (Eds. J. LAFFERTY; C. K. I. WILLIAMS; J. SHAWE-TAYLOR; R. ZEMEL; AND A. CULOTTA). (cited on page 25)
- PINHEIRO, P. O.; COLLOBERT, R.; AND DOLLÁAR, P., Learning to Segment Object Candidates. In *Advances in Neural Information Processing Systems (NIPS)*. (cited on pages 96, 107, 109, 117, 119, 129, 130, and 131)
- PINHEIRO, P. O.; LIN, T.-Y.; COLLOBERT, R.; AND DOLLÁAR, P., Learning to Refine Object Segments. In *European Conference on Computer Vision (ECCV)*. (cited on pages xv, 5, 117, 119, 129, 130, and 131)
- RATLIFE, N. D.; SILVER, D.; AND BAGNELL, J. A., Learning to search: Functional gradient techniques for imitation learning. *Autonomous Robots*, (2009). (cited on page 74)
- REAL, E.; SHLENS, J.; MAZZOCCHI, S.; PAN, X.; AND VANHOUCHE, V., YouTube-BoundingBoxes: A Large High-Precision Human-Annotated Data Set for Object Detection in Video. *CoRR*, abs/1702.00824 (2017). (cited on page 137)
- REN, M. AND ZEMEL, R. S., End-to-End Instance Segmentation and Counting with Recurrent Attention. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 127)

- REN, S.; HE, K.; GIRSHICK, R.; AND SUN, J., Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*. (cited on pages xvi, xxiv, 1, 41, 42, 43, 111, 116, 122, 123, and 124)
- RIFFENBURGH, R. H. AND CLUNIES-ROSS, C. W., 1960. *Linear discriminant analysis*. University of Hawai'i Press. (cited on pages 14 and 25)
- ROMERA-PAREDES, B. AND TORR, P. H. S., Recurrent Instance Segmentation. In *European Conference on Computer Vision (ECCV)*. (cited on page 118)
- RUSSAKOVSKY, O.; DENG, J.; SU, H.; KRAUSE, J.; SATHEESH, S.; MA, S.; HUANG, Z.; KARPATHY, A.; KHOSLA, A.; BERNSTEIN, M.; BERG, A. C.; AND FEI-FEI, L., ImageNet Large Scale Visual Recognition Challenge. *International Journal on Computer Vision (IJCV)*, 115, 3 (2015). (cited on pages xvi, 35, 36, 45, and 137)
- SCHARR, H.; MINERVINI, M.; FISCHBACH, A.; AND TSAFTARIS, S. A., Annotated image datasets of rosette plants. In *European Conference on Computer Vision (ECCV)*. (cited on page 116)
- SCHWING, A. G. AND URTASUN, R., Fully Connected Deep Structured Networks. *CoRR*, abs/1503.02351 (2015). (cited on page 97)
- SHEN, Z. AND XUE, X., Do More Dropouts in Pool5 Feature Maps for Better Object Detection. *CoRR*, abs/1409.6911 (2014). (cited on pages 74 and 87)
- SHI, J.; LIAO, R.; AND JIA, J., CoDeL: An Efficient Human Co-detection and Labeling Framework. In *IEEE International Conference on Computer Vision (ICCV)*. (cited on pages xvi, 3, 52, 53, 54, 56, 60, 61, 62, 64, 65, 66, 71, 72, 74, 96, and 97)
- SIMONYAN, K. AND ZISSERMAN, A., Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*. (cited on pages xvi, 36, 40, 74, 86, 87, 103, and 123)
- SRIVASTAVA, N.; HINTON, G.; KRIZHEVSKY, A.; SUTSKEVER, I.; AND SALAKHUTDINOV, R., Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research (JMLR)*, 15, 1 (Jan. 2014). (cited on page 32)
- SZEGEDY, C.; LIU, W.; JIA, Y.; SERMANET, P.; REED, S.; ANGUELOV, D.; ERHAN, D.; VANHOUCHE, V.; AND RABINOVICH, A., Going Deeper with Convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 74)

- 
- TIGHE, J.; NIETHAMMER, M.; AND LAZEBNIK, S., Scene Parsing with Object Instances and Occlusion Ordering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 116)
- UHRIG, J.; CORDTS, M.; FRANKE, U.; AND BROX, T., Pixel-Level Encoding and Depth Layering for Instance-Level Semantic Labeling. In *German Conference on Pattern Recognition (GCPR)*. (cited on pages 117 and 127)
- UIJLINGS, J.; VAN DE SANDE, K.; GEVERS, T.; AND SMEULDERS, A., Selective Search for Object Recognition. *International Journal on Computer Vision (IJCV)*, (2013). (cited on pages xv, 4, 5, 40, 75, 76, 96, 107, 117, and 130)
- VEDALDI, A.; GULSHAN, V.; VARMA, M.; AND ZISSERMAN, A., Multiple kernels for object detection. In *IEEE International Conference on Computer Vision (ICCV)*. (cited on pages 54 and 74)
- VINEET, V.; WARRELL, J.; AND TORR, P. H., Filter-based mean-field inference for random fields with higher-order terms and product label-spaces. In *European Conference on Computer Vision (ECCV)*. (cited on pages 54 and 74)
- VIOLA, P. AND JONES, M. J., Robust Real-Time Face Detection. *International Journal on Computer Vision (IJCV)*, (2004). (cited on pages 3 and 4)
- WAINWRIGHT, M. J. AND JORDAN, M. I., Graphical Models, Exponential Families, and Variational Inference. *Foundations and Trends in Machine Learning*, 1 (jan 2008). (cited on page 22)
- WANG, L. AND HE, D.-C., Texture Classification Using Texture Spectrum. *Pattern Recognition (PR)*, (Aug. 1990). (cited on pages xv, 25, and 26)
- WEINBERGER, K.; BLITZER, J.; AND SAUL, L., Distance Metric Learning for Large Margin Nearest Neighbor Classification. In *Advances in Neural Information Processing Systems (NIPS)*. (cited on pages 25 and 54)
- WEINBERGER, K. AND CHAPELLE, O., Large Margin Taxonomy Embedding for Document Categorization. In *Advances in Neural Information Processing Systems (NIPS)* (Eds. D. KOLLER; D. SCHUURMANS; Y. BENGIO; AND L. BOTTOU). (cited on page 25)
- WEINBERGER, K. AND SAUL, L., Fast solvers and efficient implementations for distance metric learning. In *International Conference on Machine Learning (ICML)*. (cited on page 25)

- WEINBERGER, K. AND SAUL, L., Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research (JMLR)*, 10 (2009). (cited on page 25)
- ZHAI, M.; BESSINGER, Z.; WORKMAN, S.; AND JACOBS, N., Predicting Ground-Level Scene Layout from Aerial Imagery. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 138)
- ZHANG, Y. AND CHEN, T., Efficient inference for fully-connected CRFs with stationarity. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on pages 54 and 74)
- ZHANG, Z.; FIDLER, S.; AND URTASUN, R., Instance-Level Segmentation with Deep Densely Connected MRFs. *CoRR*, abs/1512.06735 (2015). (cited on pages 116 and 117)
- ZHANG, Z.; SCHWING, A.; FIDLER, S.; AND URTASUN, R., Monocular Object Instance Segmentation and Depth Ordering with CNNs. In *IEEE International Conference on Computer Vision (ICCV)*. (cited on page 117)
- ZHENG, S.; JAYASUMANA, S.; ROMERA-PAREDES, B.; VINEET, V.; SU, Z.; DU, D.; HUANG, C.; AND TORR, P., Conditional Random Fields as Recurrent Neural Networks. In *IEEE International Conference on Computer Vision (ICCV)*. (cited on page 97)
- ZHENG, S.; PRISACARIU, V. A.; AVERKIOU, M.; CHENG, M.-M.; MITRA, N. J.; SHOTTON, J.; TORR, P. H.; AND ROTHER, C., Object Proposal Estimation in Depth Images using Compact 3D Shape Manifolds. *German Conference on Pattern Recognition (GCPR)*, (2015). (cited on page 96)
- ZHU, Y.; URTASUN, R.; SALAKHUTDINOV, R.; AND FIDLER, S., segDeepM: Exploiting segmentation and context in deep neural networks for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. (cited on page 74)
- ZITNICK, C. L. AND DOLLÁR, P., Edge Boxes: Locating Object Proposals from Edges. In *European Conference on Computer Vision (ECCV)*. (cited on pages 96 and 107)