Mid-level representations for modeling objects

Stavros Tsogkas

NNT : 2016SACLC012

# PhD Thesis
from
Paris-Saclay University
prepared at
CentraleSupélec

Ecole Doctorale n° 580
STIC | Sciences et technologies de l'information et de la communication

Mathématiques et Informatique

by

Mr. Stavros Tsogkas

Mid-level representations for modeling objects

Thesis presented and defended at Châtenay-Malabry : 15/01/2016

Members of jury:

Titre : Représentations de niveau intermédiaire pour la modélisation d'objets

Mots clés : Axes médians, parties d'objets, multiple instance learning, réseaux de neurones convolutionnels, segmentation sémantique, modèles de parties déformables

Résumé : Dans cette thèse, nous proposons l'utilisation de représentations de niveau intermédiaire, et en particulier I) d'axes médians, ii) de parties d'objets, et iii) des caractéristiques convolutionnels, pour modéliser des objets.

La première partie de la thèse traite de détecter les axes médians dans des images naturelles em couleur. Nous adoptons une approche d'apprentissage, en utilisant la couleur, la texture et les caractéristiques de regroupement spectral pour construire un classificateur qui produit une carte de probabilité dense pour la symétrie. Multiple Instance Learning (MIL) nous permet de traiter l'échelle et l'orientation comme des variables latentes pendant l'entraînement, tandis qu'une variante fondée sur les forêts aléatoires offre des gains significatifs en termes de temps de calcul.

Dans la deuxième partie de la thèse, nous traitons de la modélisation des objets, utilisant des modèles de parties déformables (DPM). Nous développons une approche «coarse-to-fine» hiérarchique, qui utilise des bornes probabilistes pour diminuer le coût de calcul dans les modèles à grand nombre de composants basés sur HOGs.

Ces bornes probabilistes, calculés de manière efficace, nous permettent d'écarter rapidement de grandes parties de l'image, et d'évaluer précisément les filtres convolutionnels seulement à des endroits prometteurs. Notre approche permet d'obtenir une accélération de 4-5 fois sur l'approche naïve, avec une perte minimale en performance. Nous employons aussi des réseaux de neurones convolutionnels (CNN) pour améliorer la détection d'objets. Nous utilisons une architecture CNN communément utilisée pour extraire les réponses de la dernière couche de convolution. Nous intégrons ces réponses dans l'architecture DPM classique, remplaçant les descripteurs HOG fabriqués à la main, et nous observons une augmentation significative de la performance de détection (14.5% de mAP).

Dans la dernière partie de la thèse nous expérimentons avec des réseaux de neurones entièrement convolutionnels pous la segmentation de parties d'objets. Nous réadaptons un CNN utilisé à l'état de l'art pour effectuer une segmentation sémantique fine de parties d'objets et nous utilisons un CRF entièrement connecté comme étape de post-traitement pour obtenir des bords fins. Nous introduirons aussi un à priori sur les formes à l'aide d'une Restricted Boltzmann Machine (RBM), à partir des segmentations de vérité terrain. Enfin, nous concevons une nouvelle architecture entièrement convolutionnel, et l'entraînons sur des données d'image à résonance magnétique du cerveau, afin de segmenter les différentes parties du cerveau humain. Notre approche permet d'atteindre des résultats à l'état de l'art sur les deux types de données.

Title : Mid-level representations for modeling objects

Keywords : Medial axes, object parts, multiple instance learning, convolutional neural networks, semantic segmentation, deformable part models

Abstract : In this thesis we propose the use of mid-level representations, and in particular I) medial axes, ii) object parts, and iii) convolutional features, for modeling objects. The first part of the thesis deals with detecting medial axes in natural RGB images. We adopt a learning approach, utilizing colour, texture and spectral clustering features to build a classifier that produces a dense probability map for medial axes. Multiple Instance Learning (MIL) allows us to treat scale and orientation as latent variables during training, while a variation based on random forests offers significant gains in terms of running time.
In the second part of the thesis we focus on object part modeling using both hand-crafted and learned feature representations. We develop a coarse-to-fine, hierarchical approach that uses probabilistic bounds for part scores to decrease the computational cost of mixture models with a large number of HOG-based templates. These efficiently computed probabilistic bounds allow us to quickly discard large parts of the image, and evaluate the exact convolution scores only at promising locations. Our approach achieves a $4\times-5\times$ speedup over the naive approach with minimal loss in performance. We also employ convolutional features to improve object detection. We use a popular CNN architecture to extract responses from an intermediate convolutional layer. We integrate these responses in the classic DPM pipeline, replacing hand-crafted HOG features, and observe a significant boost in detection performance (14.5% increase in mAP).
In the last part of the thesis we experiment with fully convolutional neural networks for the segmentation of object parts. We re-purpose a state-of-the-art CNN to perform fine-grained semantic segmentation of object parts and use a fully-connected CRF as a post-processing step to obtain sharp boundaries. We also inject prior shape information in our model through a Restricted Boltzmann Machine, trained on ground truth segmentations. Finally, we train a new fully-convolutional architecture from a random initialization, to segment different parts of the human brain in magnetic resonance image data. Our methods achieve state-of-the-art results on both types of data.

Acknowledgements

In the four years I spent doing my PhD, I have been fortunate enough to interact with several people, both on a personal and a professional level. I believe that each one of these people has contributed in his own way towards the completion of this thesis, either directly or in an indirect, subtle way. I am also sure that a few lines (or even pages) are not enough to acknowledge their contribution. Nonetheless, I will still make an attempt, hoping to include as many people as memory and time allows.

First and foremost I would like to express my gratitude to my thesis supervisor, Iasonas Kokkinos. Iasonas has been an excellent advisor, suggesting research directions, but at the same time giving me the freedom to experiment with my own ideas and take part in projects that were not directly related to his own interests. His experience on a variety of research topics, and his hands on attitude were incredibly valuable, especially during the first years of my PhD. His responsiveness, meticulousness, and constant encouragement when things were not going as planned, motivated me to keep pushing myself and putting more effort –effort that paid off in the end. Importantly, Iasonas has also been friendly and supportive, generously providing advice and help with professional matters when needed. I feel very lucky that I had the opportunity to have him as a mentor –an opportunity I owe to Prof. Petros Maragos, who initially brought us into contact while I was still an undergrad student in NTUA.

Besides my advisor, I was fortunate enough to be surrounded by excellent colleagues and personnel at the Center for Visual Computing. I thank our lab director, Prof. Nikos Paragios who supported my participation in scientific summer schools and conferences, and encouraged my involvement in interesting projects. I thank Haithem for sharing the same office with me for almost three years and Eduard Trulls for collaborating with me during his visit in CVC. I would also like to thank the rest of my labmates for all the moments we shared, either within the walls of the lab, working for some deadline, or somewhere in Paris enjoying a break from research : Aris, Olivier, Loic, Chaohui, Katerina, Sarah, Fabrice, Stavros (aka "the tall one"), Bo, Enzo, Eugene, Evgenios, Vivien, Puneet, Maxim, Stefan. I particularly appreciate Siddhartha and Adam's feedback and useful discussions during the last months of my PhD. Last but not least, I am grateful to our secretary, Natalia, for being so helpful and effective with administrative matters, especially in the last, stressing months before my defence.

I am grateful to the members of my committee, Prof. Josev Sivic, Prof. Sven Dickinson, Prof. Frederic Jurie, Prof. Andrea Vedaldi, Prof. Alan Yuille and Dr. Florent Peronnin, for the honor of reviewing my work and being present at my defence. I particularly thank Prof. Andrea Vedaldi for accepting my visit in the Visual Geometry Group, as well as Karel, Chai, Mircea, Elliot, Duncan, Aravindh, Karen, Max, Mohsan, Relja and Martha, for making my stay in Oxford even more enjoyable. I also appreciate Dr. George Papandreou for his useful input in mutual research projects.

On a personal level, there are so many people to which I want to express my most sincere gratefulness and affection, starting with my colleagues and long-time friends from NTUA, Vaggelis, Apostolis, Gavriil, Panagiota, and Michalis. Also, my friends Valia, Dimitris, Ektoras, Thanos, who helped me feel at home during my first months in Paris, and Christos, Natalia, Margarita, Alexandra, Fragkiskos, Kostas, Christina for the endless, colorful sessions of Munchkin and Settlers of Catan. I will not forget my old friends Stratis, Yannis, George, Panayotis, Nick, Kotsos, Stathis, Ilias, for simply staying in touch all this time, and Anthi for putting up with the distance for two years, and for giving me a really good reason to travel frequently to Athens. Special credit goes to Yannis Gkioulekas, for being a loyal friend, a mentor and a role model ; I have always considered his advice, help, and remote company invaluable for my academic and personal growth.

Finally, it goes without saying that I would not be here today without the unconditional love from my family, and especially my mother, who has supported me in every decision I have made so far. This thesis is dedicated to everyone mentioned in these acknowledgements, and to all the rest that I have left out. Realizing there are so many wonderful people around me is as rewarding as reaching the end of this unique journey.

Contents

List of Figures

Chapter 1

Introduction

Contents

1.1 Context and Motivation

An important problem in computer vision is to analyze a scene and recognize all of the constituent objects (*recognition*), and/or their positions (*detection*). During the last two decades, advances in available datasets [Everingham 2010, Deng 2009, Lin 2014], hardware processing power [Kirk 2007, Lindholm 2008, Nickolls 2010], and algorithms [Rumelhart 1988, LeCun 1998, Krizhevsky 2012, Girshick 2014a] have reduced, or even eliminated the gap between machine and human performance [He 2015, Yu 2015]. However, there are many computer vision tasks on which computers cannot match human performance.

Object recognition is challenging due to a number of reasons. First of all, there are tens of thousands of different object classes found in real images. These objects also vary from rigid (*e.g.* furniture, vehicles) to highly deformable (*e.g.* animals and people). Second, even if we restrict our interest on a single object class, we have to deal with an enormous *intra*-class variability : instances can have different shapes and appearances, assume complex non-rigid articulations, and appear under uncommon illumination or viewpoint conditions.

Given the huge number of possible appearance, pose, viewpoint, and context combinations, exhaustively comparing every novel input to a large library of exemplars is not a viable strategy. The standard approach in almost every state-of-the-art recognition algorithm involves the extraction of *features* related to the target task, a possible intermediate *encoding* step, and a *learning* algorithm that maps this representation to a class label and/or a structured output such as the coordinates of an object's bounding box.

Starting from the extraction of features from an image, many techniques have traditionally relied on low-level image processing to extract fundamental primitives, such as edges or corners. Features of this type are relatively simple to compute and local in nature ; for instance, edges measure abrupt changes in grayscale image brightness, between adjacent pixels [Canny 1986]. More complex features can also be devised, *e.g.* histograms of color and texture in a small neighborhood around each pixel [Martin 2004], differences of Laplacians of Gaussians in the scale space [Lowe 2004], differences of image brightenss in adjacent regions [Viola 2001], or histograms of gradients (HOG) over a canonical grid [Dalal 2005]. These features have proved to be particularly effective for boundary detection [Martin 2004, Arbelaez 2011], image retrieval [Sivic 2003, Philbin 2007, Jegou 2008, Jégou 2010, Arandjelović 2012], and object detection [Viola 2001, Felzenszwalb 2008, Felzenszwalb 2010b] and can be further processed to construct high-level interpretations of the input image.

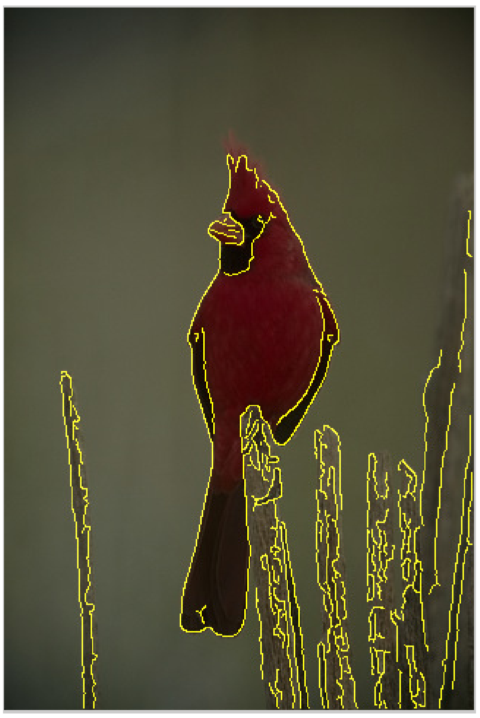
Feature *encodings* or *embeddings* can be extracted around points of interest as in [Sivic 2003] or densely over the image domain as in [Jurie 2005, Moosmann 2007] for image recognition and object categorization. Lazebnik *et al.* build a global image descriptor on top of densely computed local features, and use it to recognize scene categories [Lazebnik 2006]. Oliva *et al.* [Oliva 2001, Oliva 2006] employ global features to design a holistic description for the same task. Li *et al.* [Li 2010] employ object detectors responses [Felzenszwalb 2008] as high-level cues, on top of which a spatial pyramid is constructed to form a global scene representation. Perronnin *et al.* exploit the Fisher Kernel [Jaakkola 1999] to design visual vocabularies, for image categorization [Perronnin 2007, Perronnin 2010]. The Fisher representation is a feature encoding combining the strengths of discriminative and generative models and have demonstrated state-of-the-art results at a low computational cost. Vedaldi and Zisserman build on the work of Maji and Berg [Maji 2009] and analyze a large family of kernels called homogeneous, which include commonly employed kernels in computer vision such as Hellinger's, the intersection and χ^2 kernel. They introduce feature maps that approximate this family of kernels and allow efficient learning methods for linear classifiers to be exploited. This leads to a significant reduction in training and testing time, maintaining a virtually identical performance to the exact kernels.

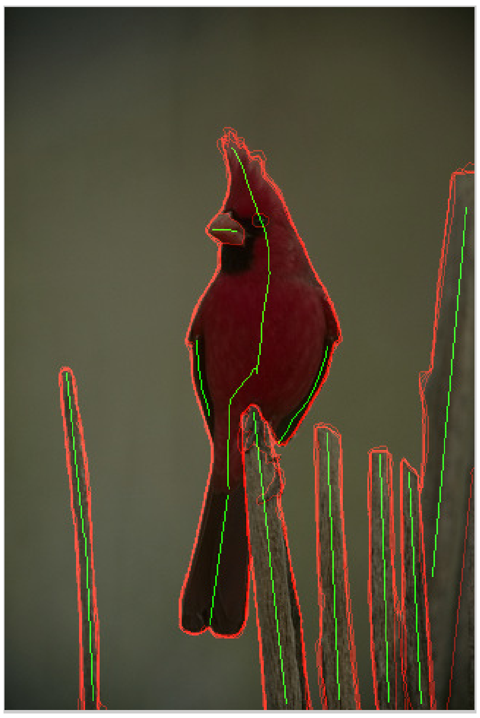
In this thesis we advocate the use of *mid-level representations* for modelling objects for object recognition and other high-level vision tasks. We argue that mid-level structures have several appealing characteristics that can help us tackle some of the challenges mentioned earlier. First, they capture information within a neighborhood or region, rather than a single pixel. As a result, they are more robust and reliable features for a recognition task (see Figure 1.2). Consider, for example, a long contour versus a very localized edge or junction, or a region of uniform color versus a single pixel in an RGB image [Felzenszwalb 2004, Gu 2009, Achanta 2012, Carreira 2012, Singh 2012]. More importantly, mid-level structures are not tied to a particular object class ; contour fragments and texture patches can appear as elements of different types of shapes and objects. This implies that models for mid-level elements can be *shared* among different objects [Stark 2009, Singh 2012, Kokkinos 2013].

Knowledge sharing or knowledge transfer has become increasingly important in recent years, as datasets keep growing in size, or extended with annotations for new object classes. Training separate models for each individual object type can quickly become computationally unattractive or even intractable.

The elements we consider in our work are *i)medial axes*, *ii)object parts*, and

(a) Input image (pixel values)


(b) Canny edges


(c) Boundaries and medial axes

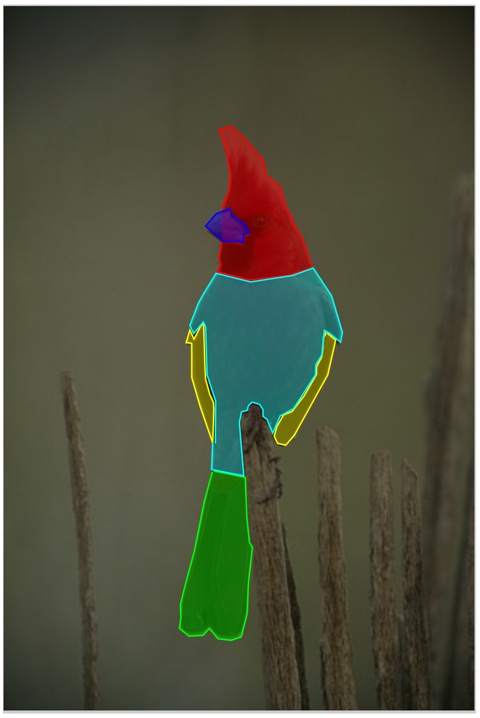

(d) Part segmentations

FIGURE 1.1 – Going from low-level to richer representations.

iii)convolutional features. Medial axes are contours carrying local symmetry information, and encode object shape in a compact way. Parts are image regions with distinct semantic connotations, and can be viewed as building blocks for objects. Convolutional features are produced as the response of a Convolutional Neural Network (CNN) layer [LeCun 1998] to a 2D input image. In contrast with hand-crafted features, such as histograms of gradients or SIFT, CNN features are *learned* directly from data.

At a first glance, this is a diverse set of cues, but, in fact, they also share many common characteristics. *Contours* of boundaries and and local symmetry axes have been previously used to transfer knowledge from existing models to new object classes, from a few training instances [Stark 2009]. An important observation regarding medial axes is that they lie in the approximate center of elongated, possibly deformable structures, which usually correspond to objects or their parts (see Figure 1.1). As such, they can be useful cues for proposing object and object part locations, or imposing constraints among pixels for foreground-background segmentation [Teo 2015], making accurate medial axis detection in cluttered images an important task.

Parts are also shared between different types of objects. For example, wheels are common in cars, aeroplanes, bicycles and motorbikes. Windows are shared by cars and buses. Horses, cows, sheep, cats and dogs, all have legs ; and although horse legs may be somewhat different than sheep legs in terms of appearance, they share a common structure, namely they have similar elongated shape and hooves. Exploiting such commonalities has leads to fewer and more expressive models, as we avoid dataset fragmentation. Sharing parts from different classes also reduces computational complexity during training and inference, as the number of object classes grows [Zhu 2010b, Fidler 2010, Wang 2015b].

CNNs became popular as digit recognition systems [LeCun 1998] and they were recently revitalized after demonstrating state-of-the-art performance for image classification. However, they are also used as feature extraction machinery, steadily replacing traditional hand-crafted features. A common practice is to pre-train a CNN on a large dataset to obtain a good initialization for the parameters of the network, and then re-purpose a subset of the network's layers for a specific task via *fine-tuning*. Notably, the same pre-trained network can be used to initialize CNNs for very diverse tasks ; a reasonable explanation is that the features produced from intermediate convolutional layers function as generic, mid-level representations, encoding increasingly complex structures in the input image in a hierarchical fashion. There have been works that investigate what parts of the input image trigger activations in different layers of the network, and they support this assumption. For example, Zeiler *et al.* [Zeiler 2010, Zeiler 2014] showed that early layers in a CNN architecture behave like Gabor filters and are activated by elements that resemble edges, whereas layers deeper in the network are activated by more complex configurations, such as object parts. CNN features have also been used recently to propose regions corresponding to full objects of arbitrary classes [Ghodrati 2015, Ren 2015].

In this thesis we focus on improving detection of the mid-level structures listed

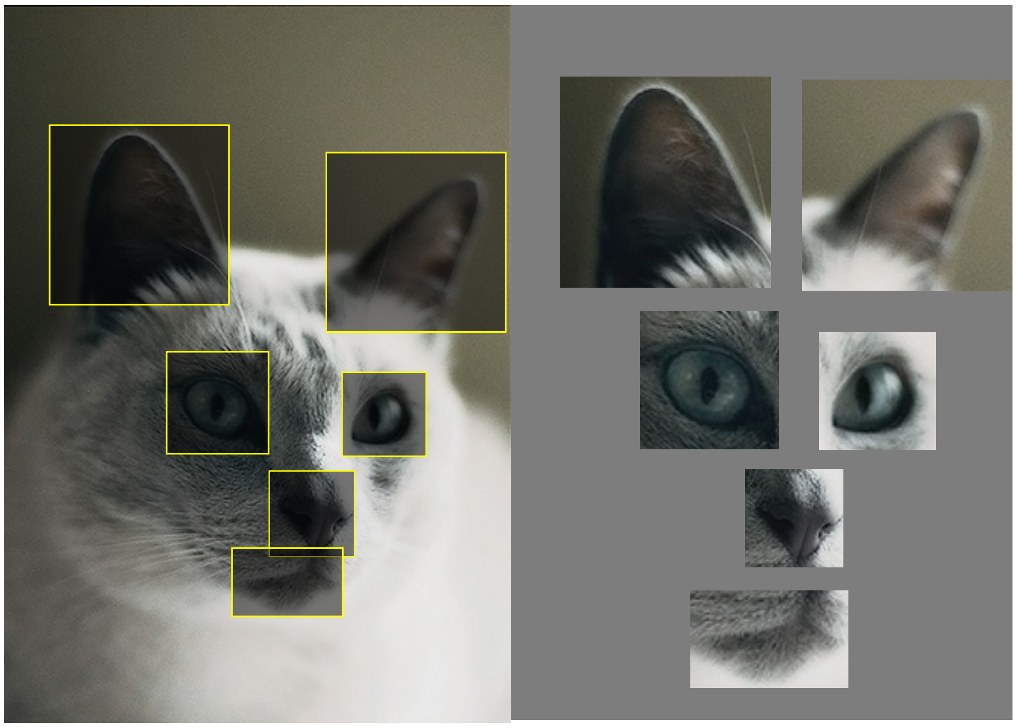


FIGURE 1.2 – An object as a composition of its parts. Extracting an appropriately chosen set of parts often provides enough information for determining the object's base class, or more fine-grained details.

above and using them to model objects, motivated by their importance for high-level vision tasks. Specifically, the tasks we target are medial axis detection in natural images, and efficient object part detection. We also use CNN features to improve DPM-based object detection and address dense semantic segmentation of object parts.

1.2 Thesis Layout

We describe below the layout of the thesis. We give a brief description of the content in each chapter, as well as a concise list of our contributions. We note that given the technical diversity of the mid-level modeling techniques utilized in this thesis, we have preferred to present previous works separately per chapter, focusing on the works that most closely pertain to our own contributions.

Chapter 2 : Medial Axes

In Chapter 2 we introduce a novel approach for detecting medial axes in natural, RGB images. Our approach uses machine learning and treats medial axis detection as a binary classification problem in which a "symmetry" or "no-symmetry" label is assigned to each image pixel. To that end, we construct two novel datasets, with medial axis annotations : one where axes are semi-automatically extracted from manually selected symmetric segments, and one with annotations directly created by a human user. These datasets are used as the training data for our learning algorithm and provide a benchmark for formal evaluation and comparison with competing methods.

Our first technical contributions is using multiple instance learning (MIL) to account for the unknown scale and orientation during training, while exploiting diverse features (color, texture, spectral clustering), extracted at multiple scales and orientations. We also build a fast variant of our algorithm, based on random forests, that makes local symmetry detection practical, locating medial axes in less than 1 second on a CPU. Finally, we explore the effect of grouping individual pixel responses into meaningful contours for suppressing false positives. Our approach surpasses all other published works in detection accuracy.

Chapter 3 : Object Parts

In Chapter 3 we turn our attention to part-based representations for objects. Our first contribution is an algorithm for efficient detection of object parts using HOG templates [Dalal 2005]. We take advantage of the recently developed Objects In Detail (OID) dataset, that contains high-quality part annotations for different types of aeroplanes, and train models for individual aeroplane parts that accommodate a large number of components. During an offline stage our algorithm constructs a tree that orders HOG templates depending on their pairwise similarity. At test time, we use this tree-cascade to construct probabilistic estimations for the parts scores and avoid evaluation at image regions with low probability of containing a part. Our results show that we can achieve almost the same performance as brute-force evaluation, with a $4\times - 7\times$ speedup.

Our second contributions consists in combining convolutional neural networks and deformable part models for object detection. We use a CNN [Krizhevsky 2012] as a feature extraction mechanism and construct a pyramid of responses from the network's last convolutional layer, extracted at multiple scales. We replace the standard HOG feature pyramid in DPMs with our CNN pyramid and show that convolutional features significantly outperform hand-crafted features for detection, even when the rest of the pipeline remains identical. From a technical standpoint, we address the increased complexity of the hybrid model following a patchwork approach that allows us to convolve response maps from multiple scales with a single elementwise product in the frequency domain. We have catered for increased computational efficiency by adapting to our task the techniques of [Dubout 2012, Girshick 2013] that have been used in conjunction with DPMs to accelerate testing and training time respectively.

Chapter 4 : Convolutional Neural Networks

In Chapter 4 we build on recent advances in deep learning and demonstrate that fully convolutional neural networks can be used for semantic segmentation of object parts. We show results in a variety of challenging and diverse datasets and object classes. We also inject prior information about part layout in our model, through a Restricted Boltzmann Machine that is trained jointly with the CNN, so as to correct erroneous segmentations and provide plausible part configurations. We also propose a method that allows us to perform semantic part segmentation "in the wild", taking advantage of a fast and accurate object detector to generate bounding box proposals and dynamically combine features from multiple scales in a CNN feature pyramid. In the second part of the chapter we design a fully convolutional network to train a system for the semantic segmentation of sub-cortical areas in the human brain. We treat the label probabilities computed by the CNN as potentials of a Markov Random Field (MRF) to impose spatial volumetric homogeneity. Our approach achieves superior performance compared to state-of-the-art segmentation methods or a similar approach using potentials based on Random Forests, on two different brain MRI datasets.

Chapter 5 : Conclusions

In Chapter 5 we conclude the thesis, summarizing our contributions and discussing ideas for future research.

Chapter 2

Medial Axis Detection in Natural Images

Contents

In this chapter we make contributions in the detection of medial axes in natural images. In Section 2.3 we introduce two novel datasets, SYMMAX300 (Section 2.3.1) and SYMMAX500 (Section 2.3.2), with medial axis annotations on images from the BSDS300 and BSDS500 benchmarks respectively. In Section 2.4 we use SYMMAX300 to train a binary classifier that *learns* to combine diverse features to output symmetry probability at each pixel. Our algorithm demonstrates state-of-the-art performance for the task of medial axis detection but feature extraction is relatively slow, amounting to approximately 20*sec* for a single 321×421 image. In Section 2.5 we develop a method that uses random forests and significantly reduces running time to under 1 second. Systematic evaluation on SYMMAX500 show that our forest-based medial axis detector outperforms both previous methods and our MIL-based variant, pushing further the state-of-the-art. In Section 2.7 we include details on the construction of SYMMAX500 and the tools we used to construct manual annotations for medial axes. The work presented in this chapter has resulted in a conference publication [Tsogkas 2012] and additional material that is part of a journal submission. It has also been used in recent papers for text line detection [Zhang 2015] and constrained foreground-background segmentation [Teo 2015].

2.1 Introduction

Symmetry is omnipresent in both natural and man-made environments. Humans can effortlessly recognise a broad variety of symmetric patterns, which are useful in analyzing and understanding complex scenes [Kootstra 2008, Locher 1989], while symmetry allows for the encoding of shape forms and relationships, positively influencing recall and discrimination [Barlow 1979, Wagemans 1998, Royer 1981]. Moreover Gestalt principles suggest that we prefer to perceive objects as symmetrical, and connect otherwise disjoint elements to form coherent objects [Soegaard 2010], while people tend to fixate in the middle of symmetric structures [Kootstra 2008].

Despite the well-known role of symmetry in human vision, it use is not as widespread in computer vision applications that deal with object recognition/detection, segmentation, and scene understanding. Many high-performing systems for object recognition rely on edges, or edge-based features, such as SIFT, HOGs and their variants, and overlook symmetry altogether. We believe that this is mainly due the lack of an efficient and accurate symmetry detector that can be directly applied on RGB images. One main contribution in this chapter lies in showing that symmetry can be readily extracted from natural images with high accuracy and speed.

In the traditional mathematical sense symmetry is a congruence property of shapes under a motion that can be a rotation, a translation or a linear motion [Weyl 1989]. In contrast, we are interested in the modified notion of symmetry, introduced by Blum in his seminal works [Blum 1967, Blum 1973]. There symmetry is viewed as a property of points in space, rather than a property of shapes themselves. This modified definition of symmetry is commonly referred to as the *medial axis*, or *skeleton* of a shape; in the rest of the text we use the terms *medial axis, symmetry axis* and *skeleton* interchangeably. We elaborate more on the medial axis representation in Section 2.2. For a detailed analysis on the matter, as well as other types of symmetry and their applications in vision we defer to [Siddiqi 2008, Liu 2009].

In this chapter we describe a learning-based approach to detect medial axes of elongated structures in natural images. These contours locally amount to approximate reflective symmetry and automatically extracting them can prove useful in numerous contexts. An obvious application is in image segmentation, where skeletons can serve as seeds for watershed segmentation, or to enforce “rigid” pairwise costs for MRFs. Symmetry axes can also be used to propose object part lo-

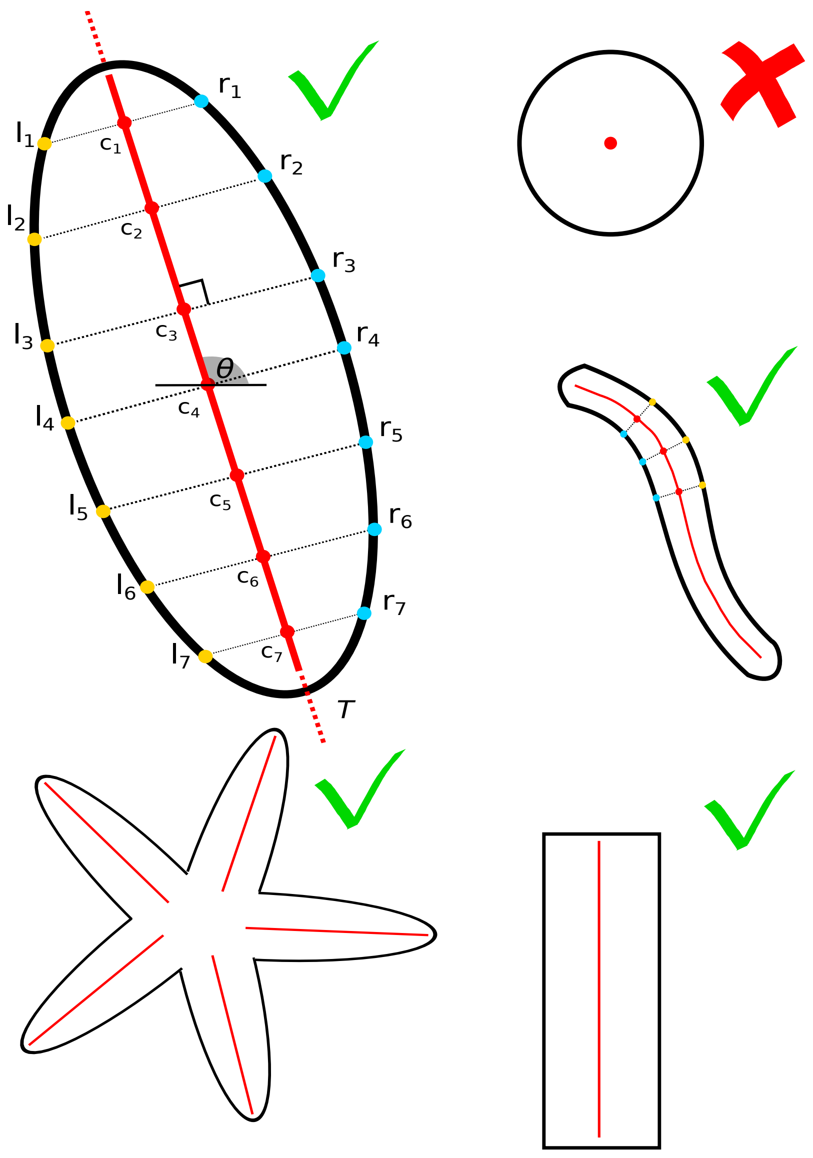


FIGURE 2.1 – Examples of local bilateral symmetries in binary shapes. Given an the image function f, T is a symmetry axis at angle θ for the ellipse, since $f(l_i) = f(r_i)$ and $|l_i - c_i| = |r_i - c_i|$. We are only interested in detecting symmetry axes of elongated structures, such as rectangles, ellipses, and "deformable snake" shapes.

cations [Kokkinos 2006], serving bottom-up object recognition, and to transfer this knowledge among object classes enabling efficient learning of new models [Stark 2009].

Our approach is distinct from the large body of work on silhouette-based symmetry detection in that instead of assuming that we are provided with pre-segmented shapes we aim at directly extracting symmetry axes from the image. This broadens the range of potential applications but also makes the problem more challenging. Several works have studied this problem over the previous decades [Lindeberg 1998, Pizer 1994, Siddiqi 2008, Lôpez 1999, Levinshtein 2009] under the names of grayscale symmetry or skeleton, or ridge, valley, crease, or ribbon detection ; it is with respect to such works that we compare our own. Employing machine learning methods to improve the extraction of other low- and mid-level features has been a particularly fruitful approach in recent years. Examples include works on boundary detection [Konishi 2003, Martin 2004], corner detection [Rosten 2006] and junction detection [Apostoloff 2005, Maire 2008].

Our first contribution consists in constructing and making publicly available a ground truth dataset for medial-axis detection. Over the course of our research we have constructed two datasets, that we denominate SYMMAX300 and SYMMAX500, using images in the Berkeley Segmentation DataSet (BSDS300/BSDS500) [Martin 2001, Arbelaez 2011]. SYMMAX300 is built using a semi-automatic skeletonization procedure on ground truth segments from BSDS300, whereas SYMMAX500 was developed at a later stage and is composed of manual annotations. Example annotations are included in Section 2.3 and details on the annotation protocol are left for Appendix 2.7. We use these datasets for both training and evaluation, and hope that it will prove useful for improving and benchmarking symmetry detection algorithms in the future.

Our second contribution lies in using multiple cues for symmetry detection. We exploit information from grayscale, color and texture cues by constructing histogram-based region discrepancy features in a manner similar to [Martin 2004] as well as a spectral clustering cue, as in [Arbelaez 2011], but we modify the extraction process so that it better fits the symmetry detection problem. In particular we extract features by considering interior-exterior region combinations that correspond to symmetry axes at multiple scale and orientations, and score each orientation and scale accordingly. As shown by our experiments, each of these cues yields a clear boost in performance. Our third contribution consists in using multiple instance learning (MIL) [Keeler 1990, Dietterich 1997] to train our symmetry detector. Our classifier's decision is based on scale- and orientation-sensitive features. MIL allows us to handle orientation and scale as latent variables during training while a noisy-or [Viola 2006] combination is used to deliver the symmetry probability map at test time.

One drawback of our method is the high complexity of the feature extraction step. The input image must be rotated and resized several times to extract features at multiple orientations and scales, increasing running time for our MATLAB/C++implementation to approximately 20-25 seconds. Our next contribution is thus focused on reducing the running time of the symmetry detector. We draw inspiration from [Lim 2013] on using *sketch tokens* for boundary detection, and show that a similar approach gives dramatic acceleration and performance improvements in symmetry detection as well. The merit of the sketch token method lies in (a) breaking up the complexity of the positive class into manageable subclasses, through clustering and (b) using fast features and classifiers to deal with the smaller subclasses. In particular, we use our manual annotations as a basis for clustering symmetry patterns into classes encoding axes of various orientations, shape and curvature ; we then train a random forest classifier that classifies each image patch into one of these classes, or a background class corresponding to the absence of symmetric patterns.

This largely outperforms our first approach, while being more than an order of magnitude faster. A single-core implementation processes a 321×481 image in approximately 1.5 seconds, while, on a dual-core PC, computation time goes down to less than a second. Being accurate *and* fast opens up the possibility of exploiting symmetry in a variety of settings. In Section 2.5.5 we couple our local symmetry detector with a grouping algorithm that delivers longer contiguous contours, that could serve as object part proposals, in a similar way edges were employed in [Zitnick 2014]
.

Finally, in Section 2.4.3 and Section 2.5.4, we use our ground truth to systematically evaluate every aspect of our algorithm, such as the contributions of each cue individually. We compare with previously reported methods and demonstrate a systematic improvement in performance.

2.2 Previous Work

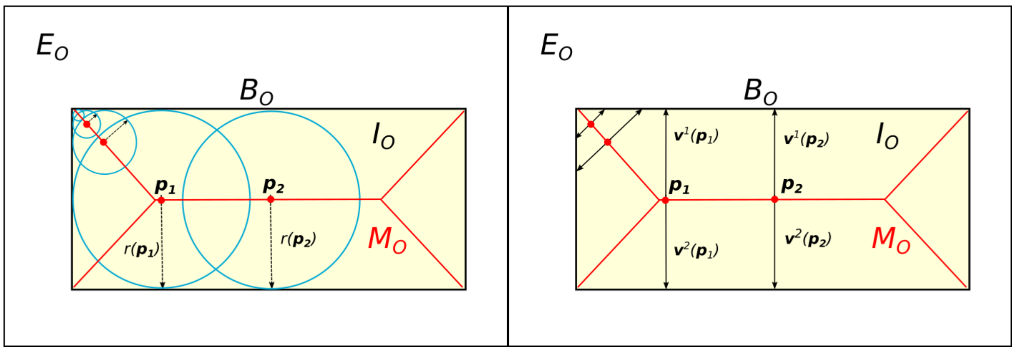
In this section we conduct a brief overview of previous works related to detection of medial axes in different types of images. We start with works on medial axis extraction for binary shapes in Section 2.2.1 and continue with works that detect ridges in grayscale images in Section 2.2.2. Works mostly related to our own are listed in Section 2.2.3 and aim to detect symmetry axes directly on cluttered RGB images.

2.2.1 Binary images

Among the multiple notions of symmetry [Liu 2009], one that has been broadly studied in vision is local reflective symmetry, and in particular its computation from binary shapes. This approach can be traced back to the *medial axis transform* (MAT) [Blum 1967, Blum 1973], defined as the locus of centers of maximal inscribed circles within a shape. We only discuss the medial axis representation for two-dimensional shapes, although its definition has been extended to 3D objects [Siddiqi 2008].

Consider an object O in the image plane, such as the rectangle in Figure 2.2a. We also consider the following notation : I_O is the interior of the object, E_O is the exterior of the object and B_O the points that separate I_O from E_O, the *object boundary*. The medial approach represents O as a set of points $\mathbf{p} \in M_O$ that lie midway between two sections of its boundary. The points $\mathbf{p}$ are the centers of *maximally inscribed* disks, bitangent to B_O in the interior of the object. The radius $r(\mathbf{p})$ of the disk is the distance between $\mathbf{p}$ and the points where the disk touches B_O, and is called the *medial radius*. Alternatively, we can associate each point $\mathbf{p}$ with a pair of vectors of equal length, connecting it to two points on the object boundary. The two representations are equivalent and are illustrated in Figure 2.2.

In both approaches the medial representation of O can be defined as a set of tuples $(\mathbf{p}, r(\mathbf{p})) \in \mathbb{R}^2 \times \mathbb{R}$. Given these pairs, we can fully reconstruct the object



(a) Disk representation (b) Vector representation

FIGURE 2.2 – Two equivalent medial axis representations.

either as a union of overlapping disks or non-overlapping spokes that sweep-out the object's interior ; similarly, given the object boundary, we can generate the medial locus $M = (\mathbf{p}, r(\mathbf{p}))$. Note that only the latter constitutes a function, as a medial point corresponds to more than one boundary points.

Usually an object and its boundary are described as a binary image, for example values of "zero" for the object's interior and exterior, and values of "one" for the object boundary. In this example we do not distinguish between interior and exterior but other value assignment schemes could be considered. The algorithmic objective is to transform this binary image, or equivalently, the curve that forms the boundary, to its medial axis. We call this operation the *Medial Axis Transform* or *MAT*. Another term that is often used to refer to the medial axis, is *skeleton*, and the corresponding process is called skeletonization.

An analogous approach was taken in [Brady 1984], which introduced a two-dimensional shape representation termed *smoothed local symmetries* (SLS). Siddiqi *et al.* introduced the theory of *shock graphs* for the generic representation of 2D shapes [Siddiqi 1999]. In their work, the medial axis of a 2D shape is computed as part of a curve evolution process and is used for shape matching and recognition [Siddiqi 1999, Sebastian 2001]. In recent work [van Eede 2006] skeleton construction is posed as an optimization problem that involves the conflicting goals of contour simplicity and good reconstruction of the original shape. These and subsequent techniques developed along these lines [Telea 2002, Telea 2004, Siddiqi 2002, Demirci 2009], demand a smooth and closed boundary, such as the result of a prior figure-ground segmentation. Such segmentations are rarely available for RGB images, making this an unrealistic assumption. Moreover, these representations are not robust, as small perturbations in the boundaries can result to severe changes or spurious branches in the final skeleton.

2.2.2 Grayscale images

In the case of grayscale images, local symmetry axes are often referred to as *ridges*. The term is borrowed from topography : a ridge is defined as a separator between regions from which water flows in different directions. However, there hasn't been a precise mathematical formulation of this property, leading to confusion among researchers. A common approach in early computer vision was to define ridges as local extrema of some hand-crafted analytic function of the image brightness. Haralick, for instance, defined bright ridges as the points where the main principal curvature is maximized in the main principal curvature direction [Haralick 1983]. Pizer *et al.* further discusses the interplay between "boundariness", the degree to which an image point behaves like a boundary, and "medialness", the degree to which an image point behaves like the middle of an object, and suggest that medialness must be derived from boundariness at various locations and scales [Pizer 1994]. Eberly *et al.* generalize the ridge definition in higher dimensions [Eberly 1994], while in [Steger 1998] an algorithm is proposed that jointly determines the ridge position and corresponding bilateral distance from the boundary with sub-pixel accuracy.

In his seminal work, Lindeberg presents a systematic methodology of addressing the *scale* at which descriptors of image data are computed [Lindeberg 1998]. He introduces the novel concept of a *scale-space edge*, defined as a connected set of points in scale-space that are locally maximal in strength, both in the gradient direction, and over neighboring scales. As a consequence, the scale levels can vary along the edge. Similar ideas can be applied for detecting ridges, with just slight modifications. In the former case, scale is chosen depending on the "coarseness" of the edge, whereas in the latter case, the selected scale reflects the width of the ridge. The scale-space representation has been studied in various works [Witkin 1984, Koenderink 1984, Yuille 1986, Lindeberg 1990, Florack 1992]. Other multiscale schemes are introduced in [Pizer 1998], whereas Lopez *et al.* [Lôpez 1999] compare many of these criteria and discuss other alternatives. All these approaches however fail in cases where texture rises as the discriminating factor between image regions, and typically result in false positives around areas with strong shading.

2.2.3 Color images

In more recent works symmetry axes are defined as the medial axes of symmetric parts, composed of superpixels extracted at multiple scales. Levinshtein *et al.* [Levinshtein 2009] present a method for extracting skeletal branches from color images based on a region segmentation process. They use superpixels extracted at different scales to build an adjacency graph and learn the probability that two adjacent, superpixels represent medial point approximations of the same symmetric part. Despite its ability to parse an image into symmetric segments, this method produces axes formed by linear segments and therefore cannot handle structures with large curvature variations along the symmetry axis. Lee *et al.* [Sie Ho Lee 2013] overcome this limitation by grouping superpixels at different scales, allowing for the detection of curved and tapered symmetric parts.

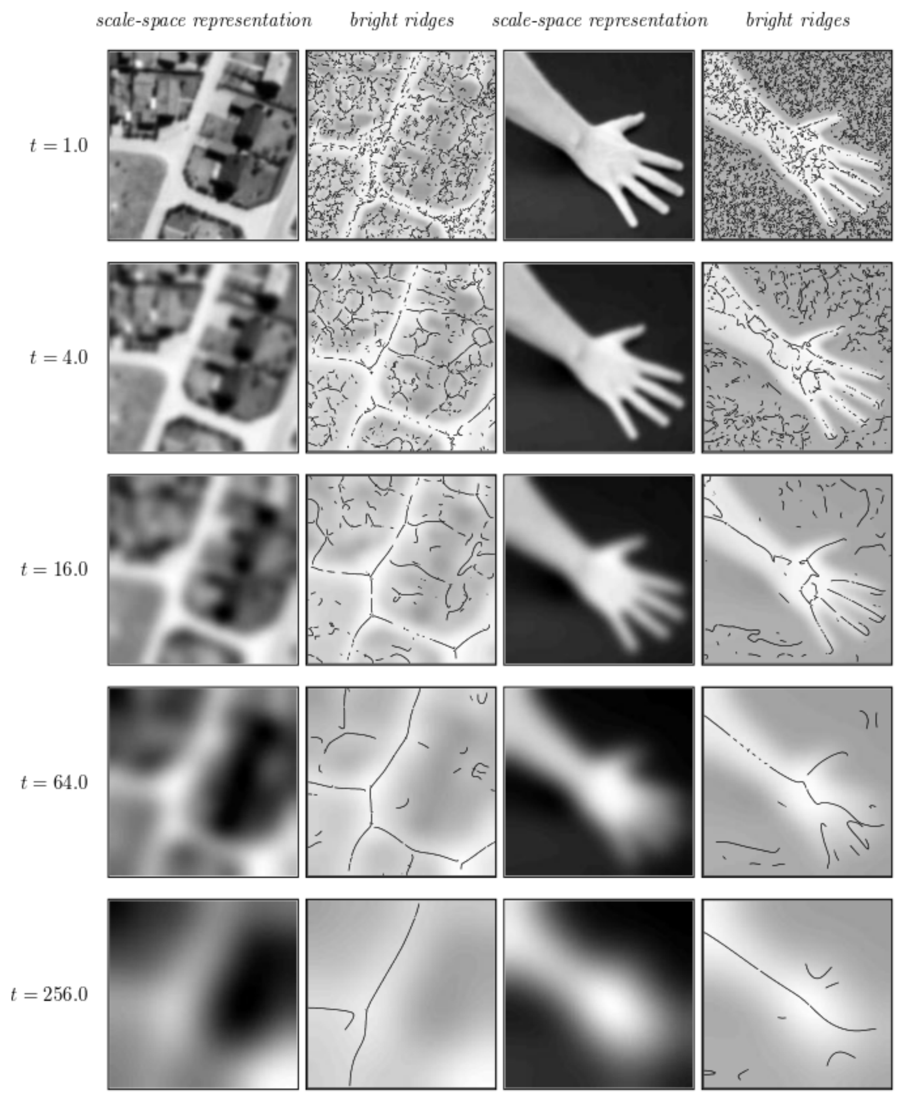


FIGURE 2.3 – Ridges computed at different scales t for an aerial image and an image of a hand (taken from [Lindeberg 1998]).

2.3 Constructing a dataset for medial axes

Please take a look at Figure 2.4 and consider that f is the image function : $f(\mathbf{x})$ is a triplet of values corresponding to the RGB channels of the image at point $\mathbf{x}$. Points $\mathbf{a}$ and $\mathbf{b}$ in the highlighted boxes are symmetric with respect to point $\mathbf{c}$ lying on the horse's medial axis ; however, $f(\mathbf{a}) \neq f(\mathbf{b})$. *Learning* to combine image cues to reason about symmetry locations seems to be a more promising approach.

Training a classifier for symmetry detection in natural images requires a ground truth dataset of annotations for symmetry axes. In this section we describe in detail the construction of the datasets used for training and evaluation in Section 2.4.3 and Section 2.5.4. We have constructed two such datasets ; the first one, SYMMAX300, was built on top of the BSDS300 dataset, using a semi-automatic approach, with

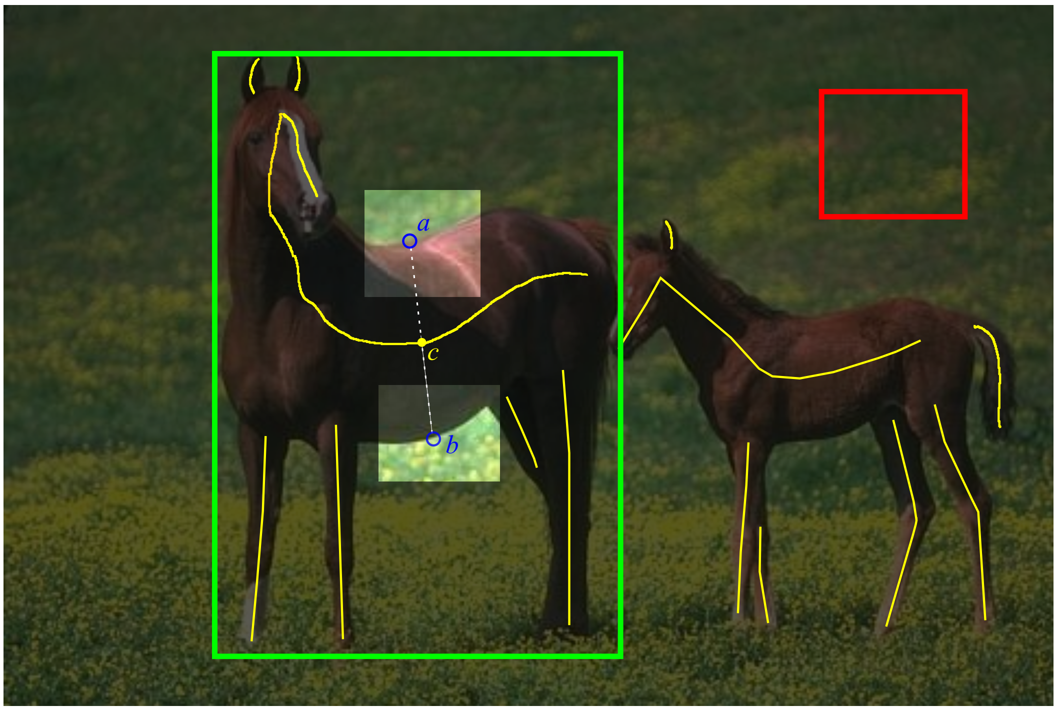


FIGURE 2.4 – Symmetry axes of elongated parts can provide a compact, robust image representation, and indicate possible object and object part locations. A bounding box corresponding to an object (green) contains numerous symmetry axes, whereas the symmetry scores inside a random, background bounding box (red) are low. We also see that for color images the analytic definition for bilateral symmetry is no longer adequate.

human supervision. The second dataset, SYMMAX500, is an extension of SYMMAX300, comprising annotations by human users, and is built on the extended BSDS500 segmentation benchmark.

2.3.1 SYMMAX300 : a Dataset for Medial Axes in Natural Images

The main technical hurdle when constructing a medial axis dataset has been the definition of a medial axis in natural images, since a number of factors, including shading, deformation, perspective effects and partial occlusion make exact reflective symmetry, defined as e.g. in [Weyl 1989], hard to occur. One option we considered was to use annotation tools such as Amazon's Mechanical Turk, but realized that this will open another "can of worms" : the annotations one gets are typically noisy, while describing symmetry to non-experts is a non trivial task. Instead, we opt for an operational definition of symmetry that exploits the segmentations available for the Berkeley Segmentation Dataset (BSDS300) [Martin 2001].

In particular, every image in the BSDS300 dataset is accompanied by 5-7 human segmentations. We combine the information provided by this dataset with a recent skeletonization algorithm [Telea 2002] to extract skeletons of image segments corres-

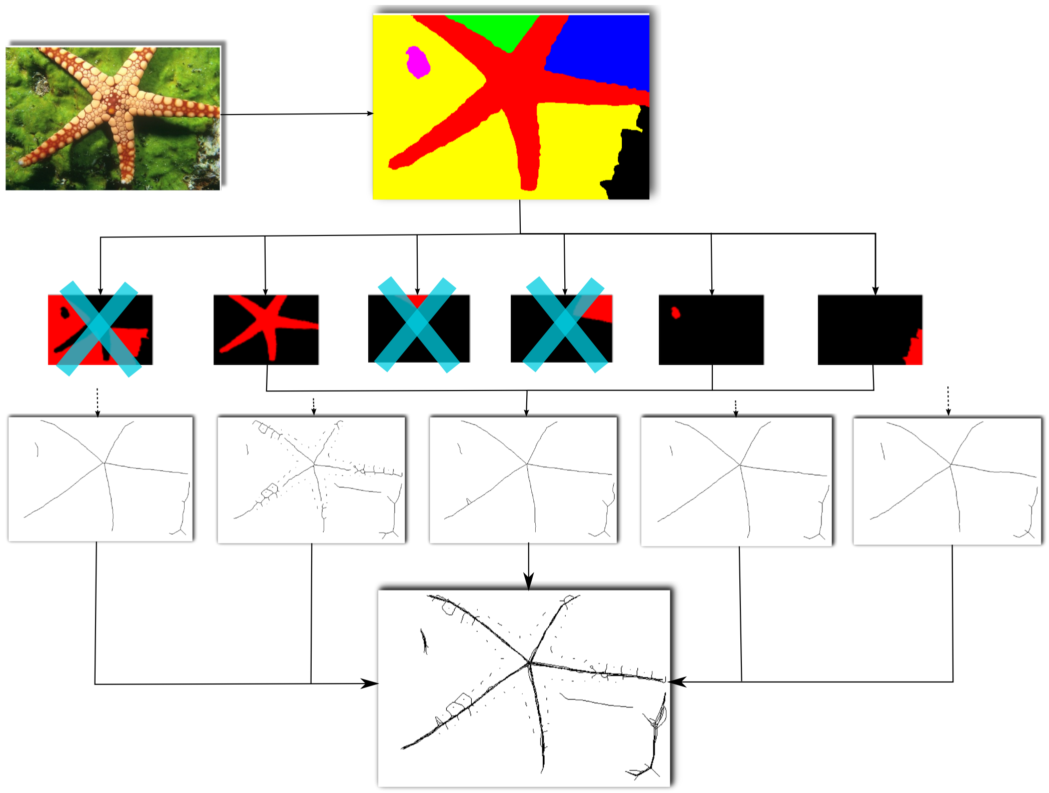


FIGURE 2.5 – Construction of an image skeleton by combining single-segment skeletons : the user examines one-by-one the segments provided in a human segmentation and rejects the ones that are deemed inappropriate for skeleton extraction. The process is repeated for every human-generated image segmentation and the union of the resulting skeletons forms the symmetry ground truth.

ponding to object parts. We employed a freely available MATLAB implementation for Telea's algorithm due to its ease of use and good qualitative results. In this way we obtain multiple binary images of segment skeletons which we combine into the final skeleton by taking their union.

Segments corresponding to the image background do not convey information useful for high-level tasks such as object recognition, so we prune them. Furthermore, we need to account for noisy branches arising from the sensitivity of the skeleton extraction algorithm to minor shape perturbations. We address these issues by creating an interface that allows a human user to supervise the construction of the ground truth : given a segmentation of the input image, the human user can examine every segment separately and decide whether it should be included in the overall skeleton map. By using skeletonization on top of human-generated segmentations we ensure that the symmetry axes will correspond to object boundaries. In Figure 2.5 we show the partial and final skeletons obtained from the segmentations of an example image. We note here that the above procedure is applied separately for each of the multiple human segmentations available for every image in the BSDS300. We aggregate the multiple skeleton maps obtained this way into a final one by taking their union.

2.3.2 SYMMAX500 : Manual Annotations for Medial Axes

The procedure described in Section 2.3.1 was our first attempt towards developing a ground truth dataset for medial axes. Although practical and effective, this approach has several drawbacks. Skeletonization often results in spurious branches, especially when the segment is not elongated enough. Also, relying on a finite set of existing segmentations immediately imposes limitations on the extracted annotations : many symmetric structures cannot be recovered, simply because they are segmented as a part of a larger image area. We decided that the only way to remedy these limitations was to take a step further and extend SYMMAX300 with annotations from human users.

As a first step to this end we used MATLAB to develop a graphical user interface (GUI) that facilitates the manual construction and editing of symmetry annotations. Details on the GUI are included in Appendix 2.7. Using this interface we constructed SYMMAX500, an extension to our previous SYMMAX300, that comprises manual annotations of medial axes for all images in the BSDS500 dataset. This new dataset corrects many of the annotation flaws of SYMMAX300, as highlighted in Figure 2.6. It is also the first set of publicly available, human-generated annotations for medial axes in natural images.

Following the list of rules formulated in the Appendix 2.7 we construct a binary ground truth map for each one of the 500 images in BSDS500. The dataset is divided in 200 images for training, 100 images for validation, and 200 images for testing. Note that at the current version of the dataset we do not explicitly annotate local orientation or scale for each ground truth pixel. We can however compute an estimation of the scale at each symmetry-positive point combining the available ground truth for boundaries and simple heuristics. For instance we can compute the distance transform for the binary annotations for symmetry and compute the minimum distance of each symmetry pixel from its nearest boundary. In Section 2.6 we discuss more ideas for improving SYMMAX500.

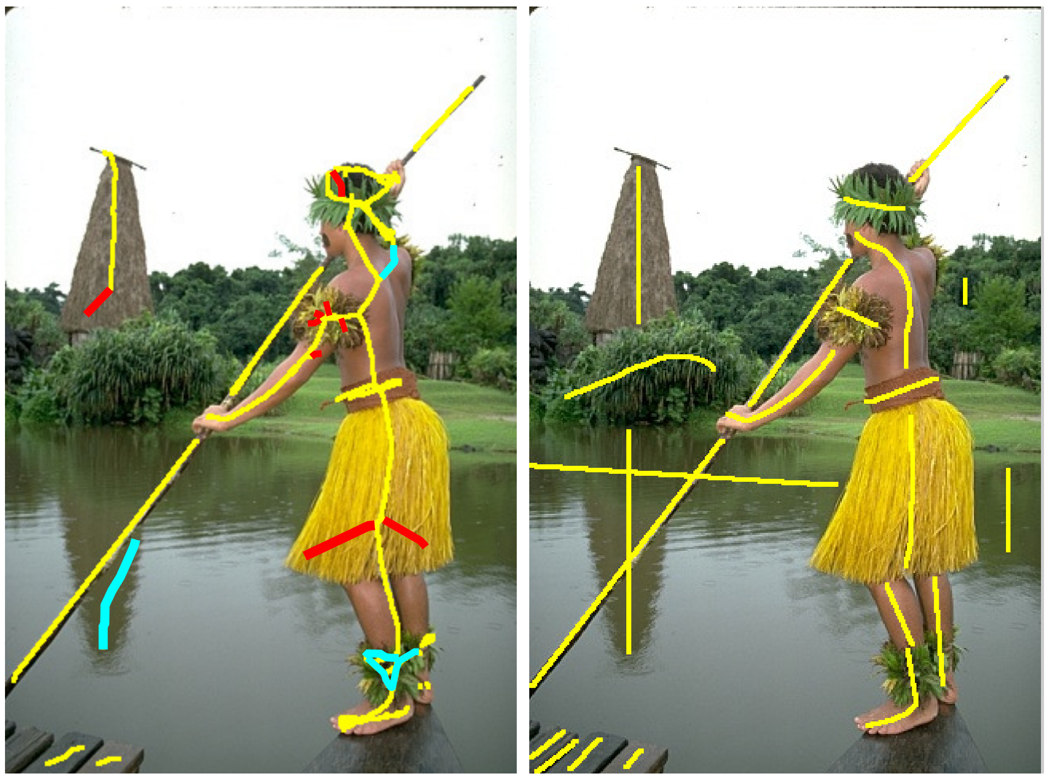


Figure 2.6 – Comparison betweem automatically created ground truth from segmentations in SYMMAX300 (left) and manually annotated data in SYMMAX500 (right). Symmetric structures are often broken into separate segments or parsed as part of a larger image area, resulting in noisy skeletons (cyan). The skeletons also contain spurious branches (red). Our new, manually annotated symmetry ground truth corrects these flaws.

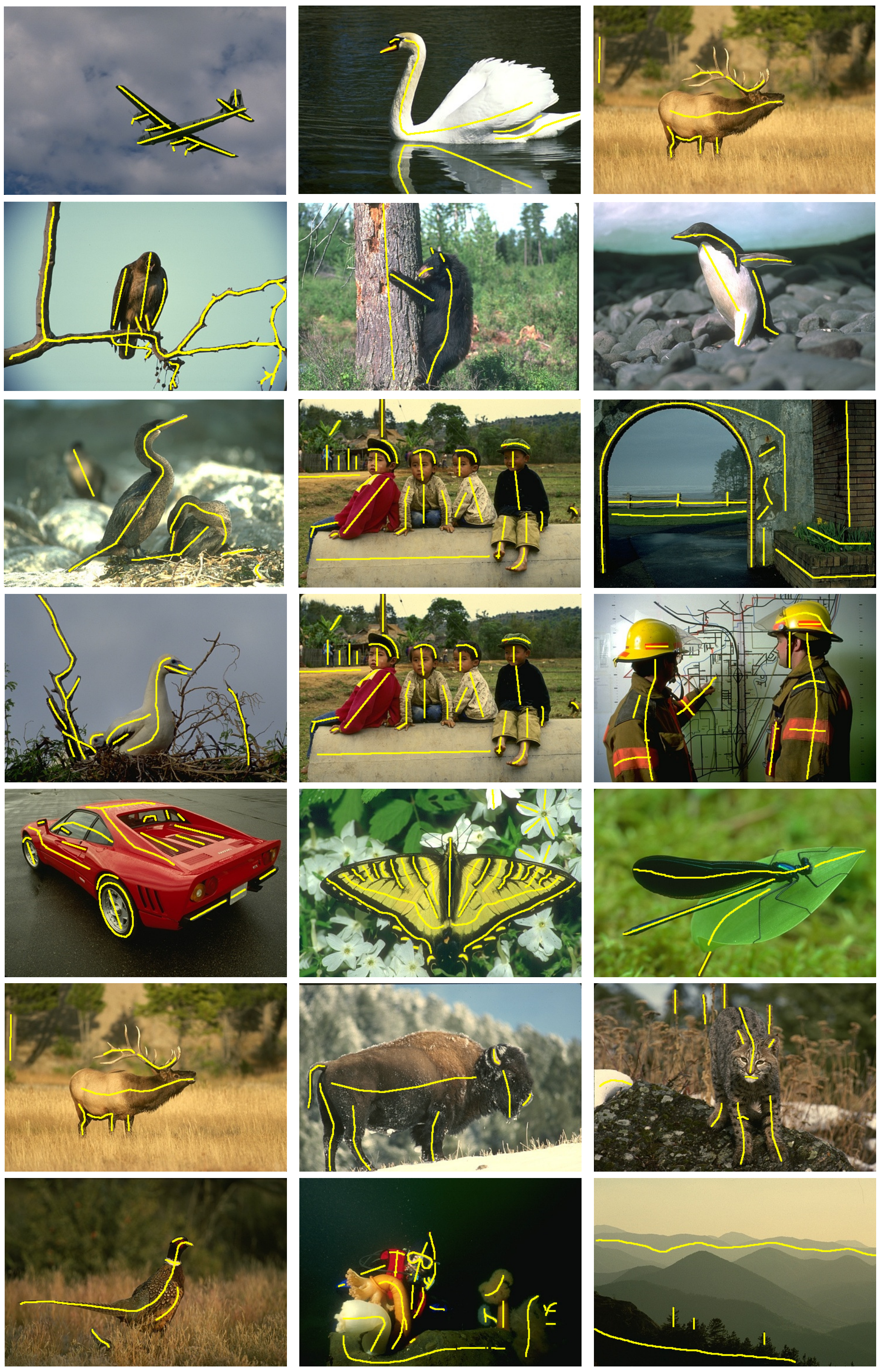


FIGURE 2.7 – Examples of binary ground truth maps from SYMMAX500.

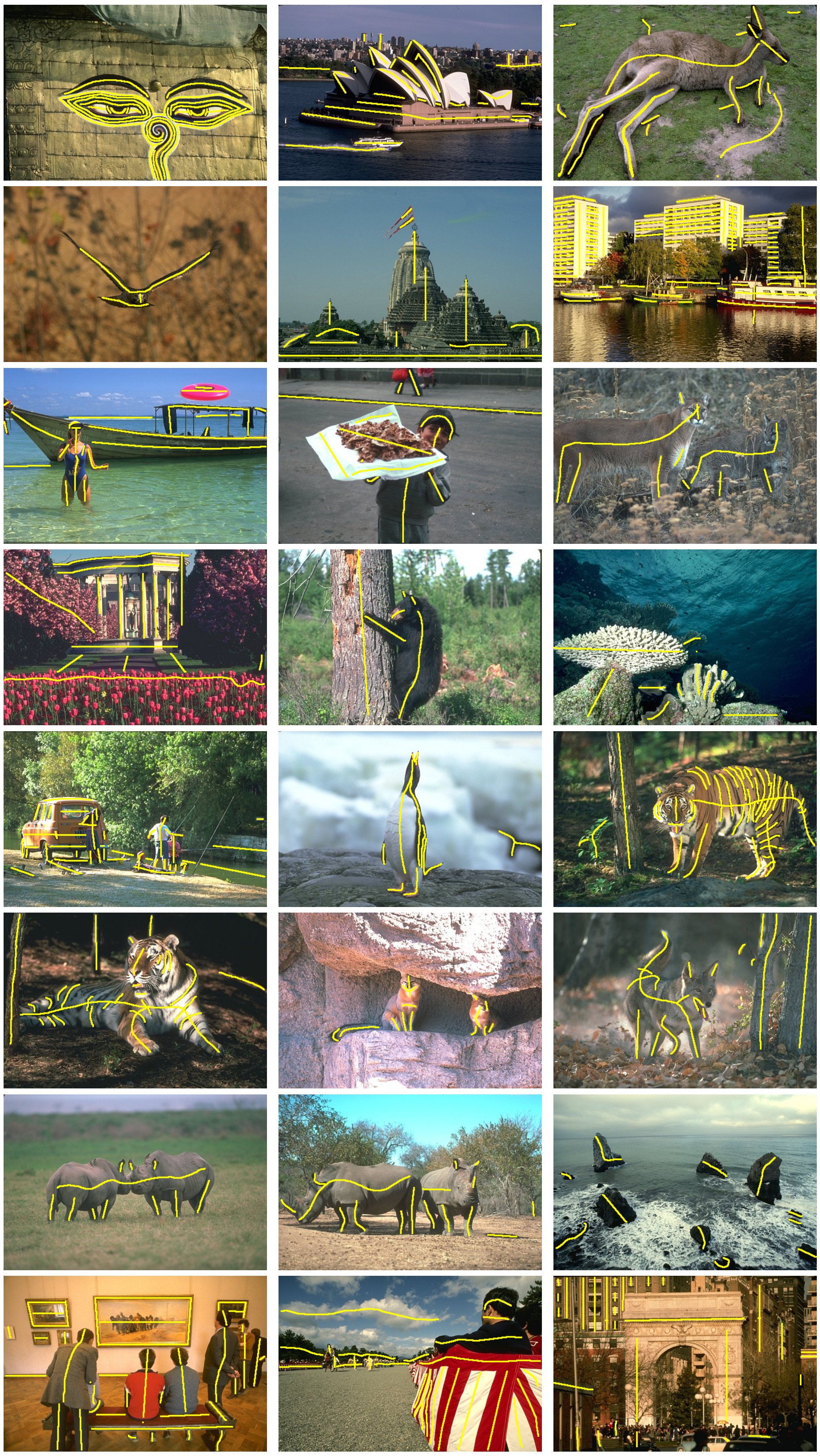


FIGURE 2.8 – Examples of binary ground truth maps from SYMMAX500.

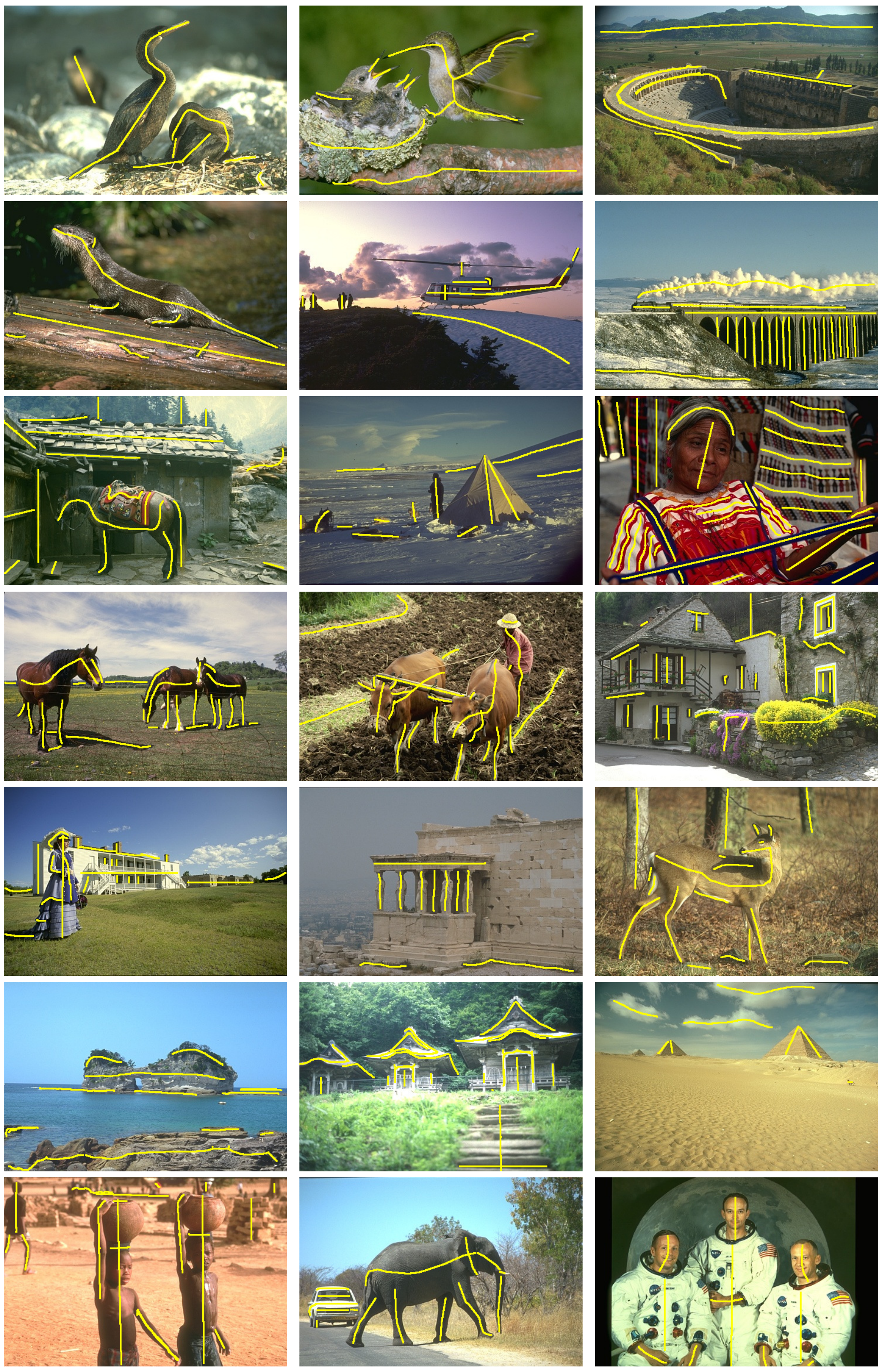


FIGURE 2.9 – Examples of binary ground truth maps from SYMMAX500.

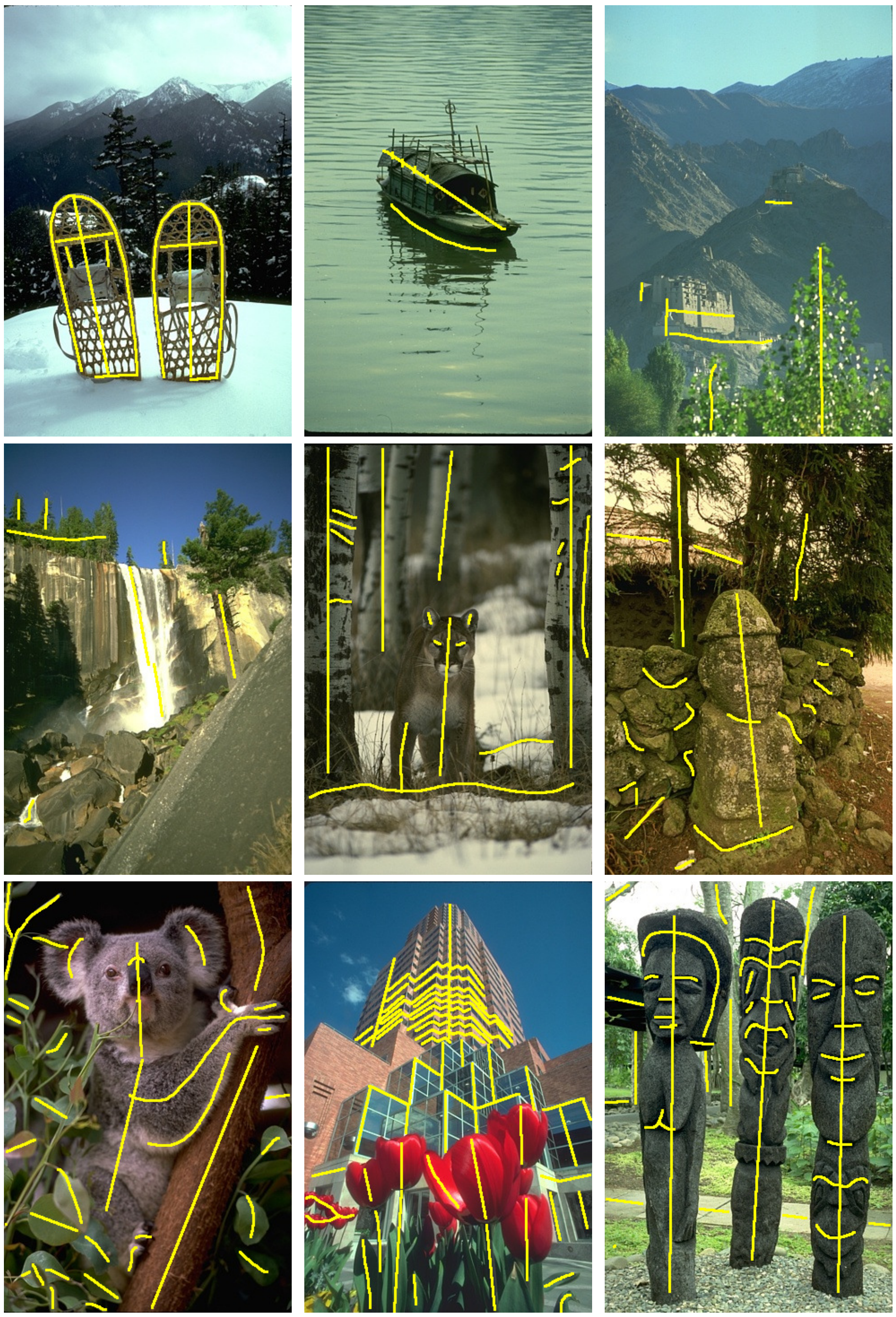


FIGURE 2.10 – Examples of binary ground truth maps from SYMMAX500.

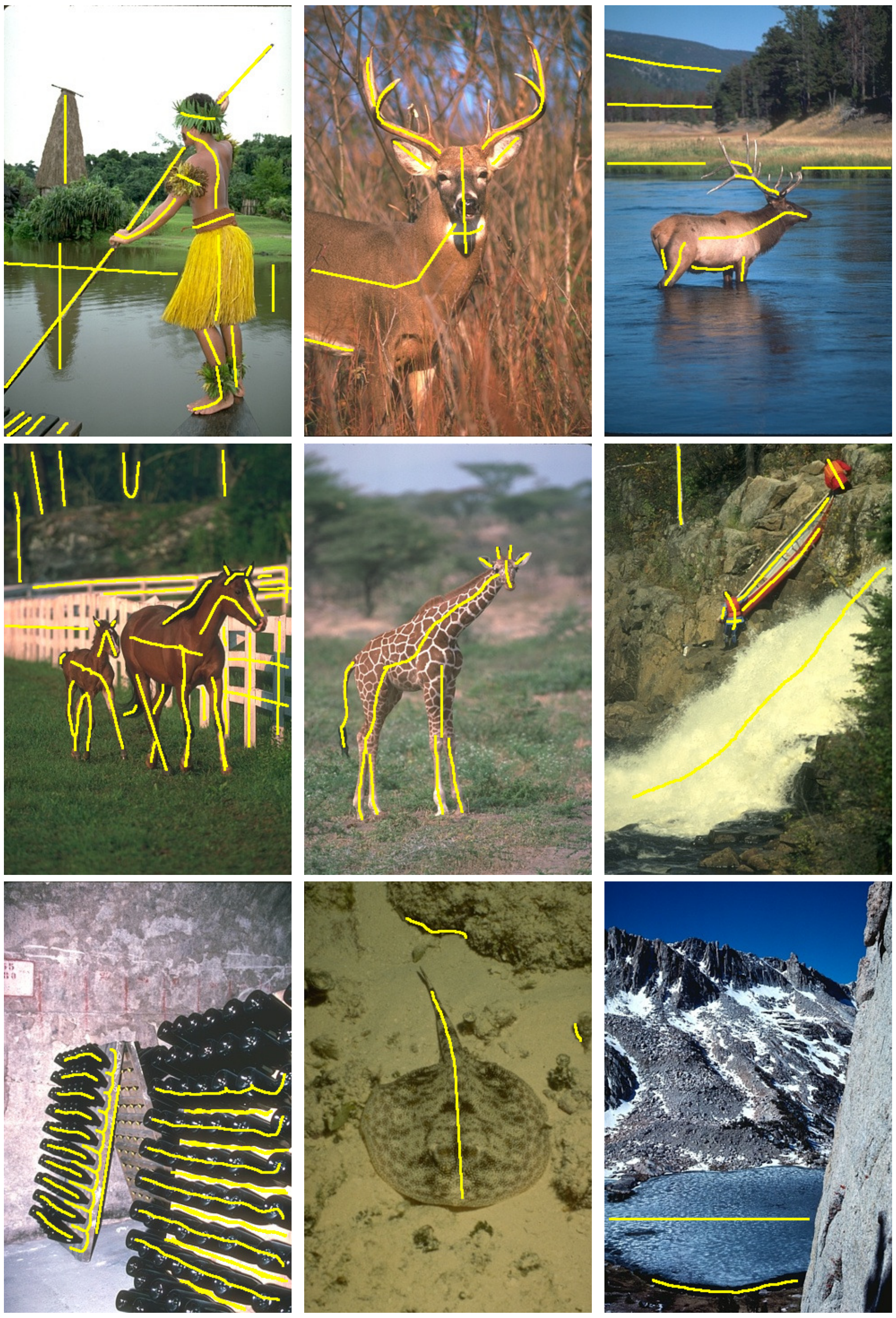


FIGURE 2.11 – Examples of binary ground truth maps from SYMMAX500.

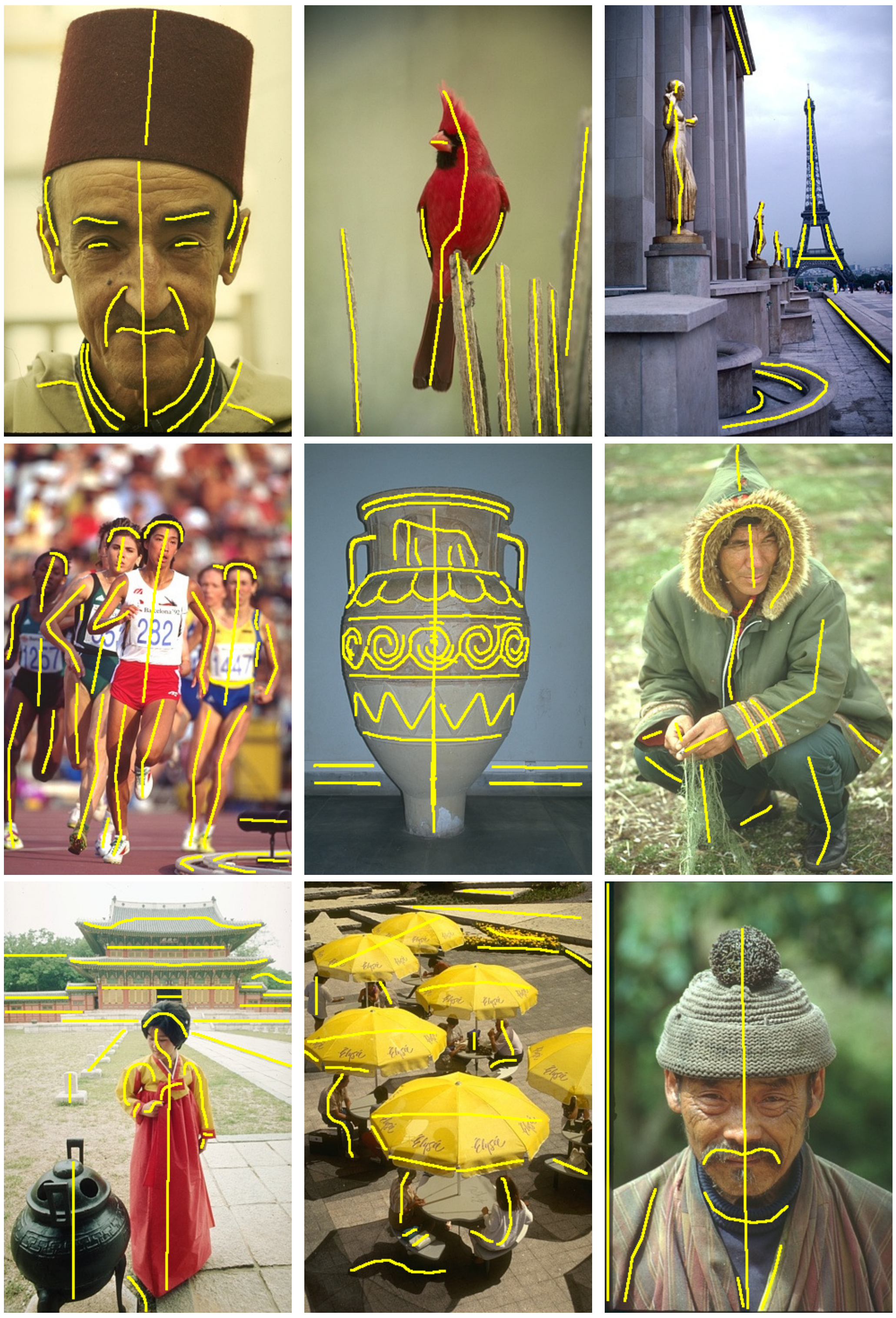


FIGURE 2.12 – Examples of binary ground truth maps from SYMMAX500.

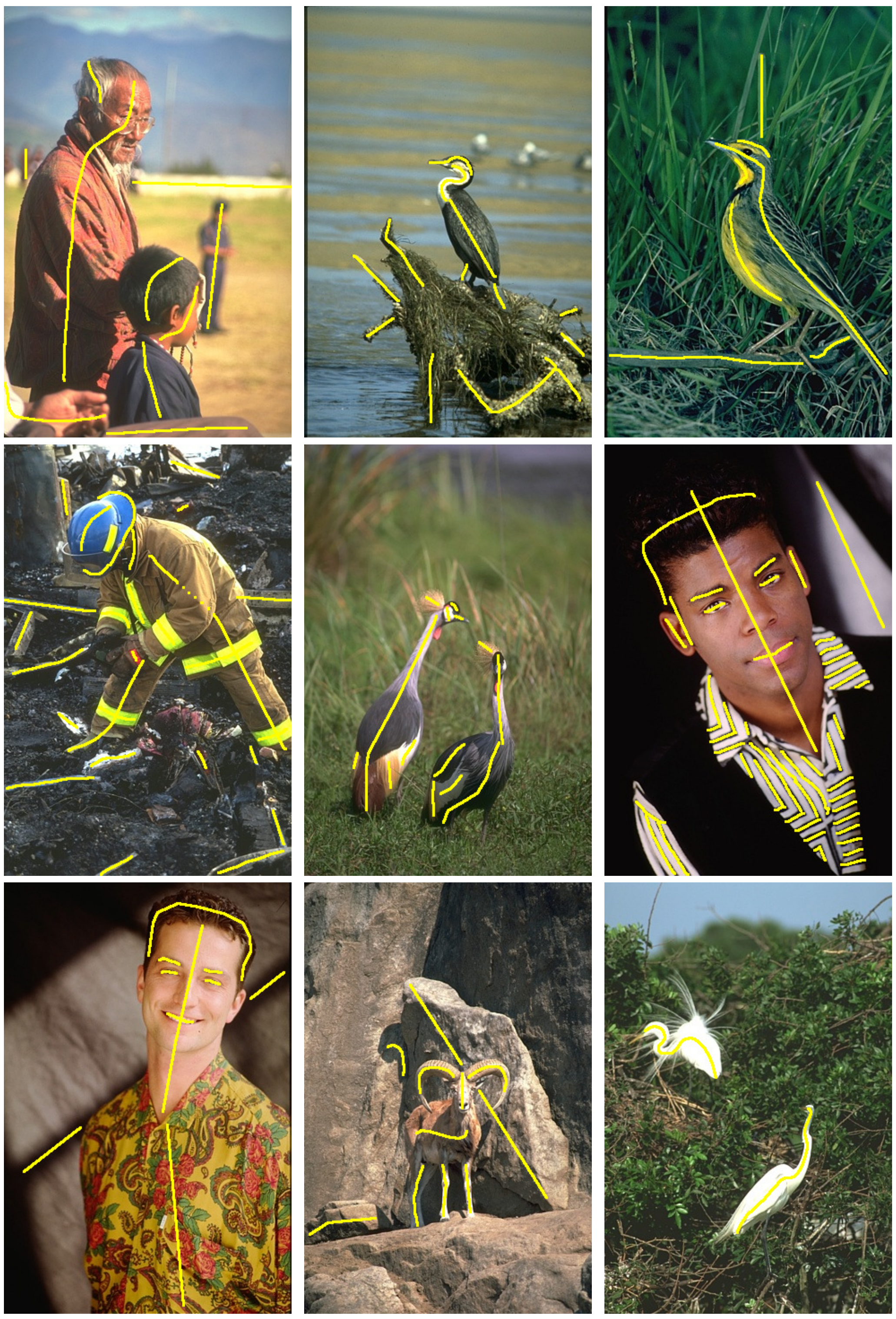


FIGURE 2.13 – Examples of binary ground truth maps from SYMMAX500.

2.4 Medial Axis Detection Using Multiple Instance Learning

In this section we describe a machine learning approach to medial axis detection, using multiple instance learning (MIL). We start by listing the different types of diverse features our classifier uses. We divide features into two categories : local features computed through histogram-based operators are defined in Section 2.4.1.1 and a global feature based on normalized cuts is formulated in Section 2.4.1.2. We then discuss how we use MIL to address the unknown scale and orientation during training in Section 2.4.2 and show qualitative and quantitative results that outperform competitive works in Section 2.4.3.

2.4.1 Feature Extraction

Our feature extraction method is inspired by [Martin 2004], where the authors use features extracted locally from image patches to determine the existence of a boundary at some orientation. Based on the success of the boundary model presented in that paper we examine whether similar local cues can be employed for symmetry detection. One significant difference is that in [Martin 2004] a circular area of fixed radius around the point of interest is used, whereas we need to consider more and also larger sizes for the pixel neighborhoods to account for symmetry at multiple scales. We do this by creating a Gaussian pyramid of the original image and using rectangle areas with sides chosen from a small set of fixed sizes. To alleviate the additional computational cost of this multi-scale analysis, we replace filtering operations with integral image-based computations [Viola 2001], and obtain features at various orientations by creating multiple integral images for rotated versions of the original image [Catanzaro 2009]. In the following, we describe in detail each of the individual features we use.

2.4.1.1 Histogram-Based Operators and Features Using Integral Images

We consider three adjacent rectangles of side lengths a and b, labeled as shown in Figure 2.14 ; the middle rectangle is centered at location (x, y) on the image plane. For any two such rectangles we define a dissimilarity function $H_{i,j}(x, y, \theta, s)$, where indices $i, j \in \{1, 2, 3\}$ indicate which two rectangles are being compared, θ denotes orientation, and $s = a/2$ denotes scale.

Following [Martin 2004], to compute H first we create a histogram representation of the empirical distribution of some feature value in the pixels included in each rectangle. Then we use the χ^2-distance function [Rubner 2001] to compare the two histograms, defined for two histograms g and h with K bins as

$$\chi^2(g, h) = \frac{1}{2}\sum_{k=1}^{K} \frac{(g(k) - h(k))^2}{g(k) + h(k)}. \tag{2.1}$$

Among the many possible distance functions between histograms we selected the

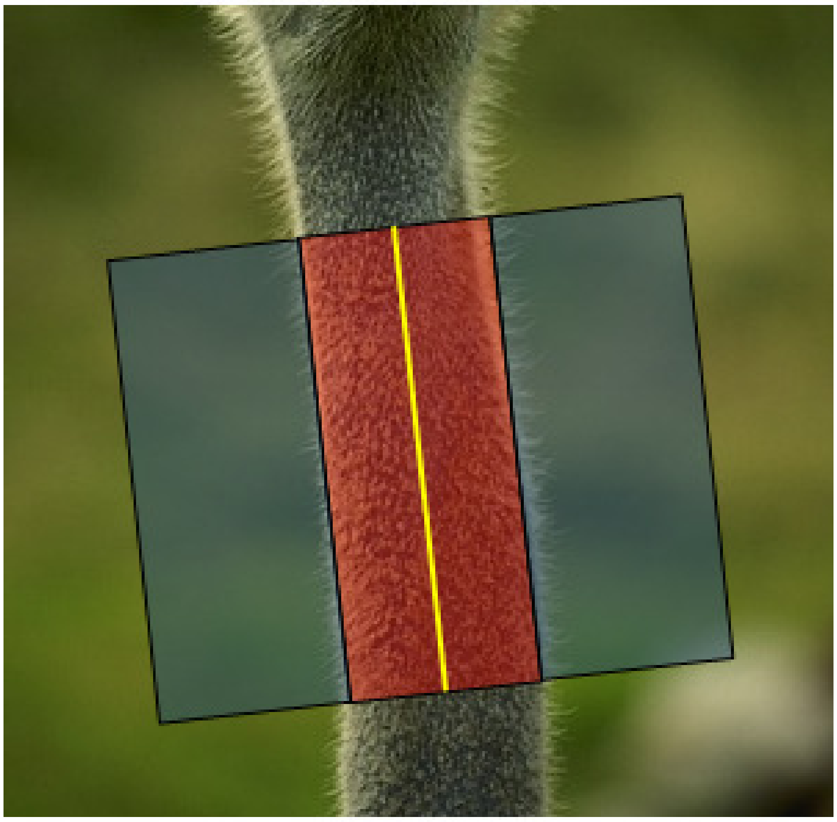


FIGURE 2.14 – Feature contents of the middle part show high dissimilarity to the contents of the left and right parts. Therefore the cue strength in the vertical diameter (in yellow) is high.

χ^2-distance for its relative computational simplicity. Using the above notation the function H becomes

$$H_{i,j}(x, y, \theta, s) = \frac{1}{2} \sum_k \frac{(R_i(k) - R_j(k))^2}{R_i(k) + R_j(k)}, \tag{2.2}$$

where R_i, R_j are the histograms constructed from regions i and j respectively.

The above procedure is used separately for brightness, color, and texture features. Brightness and color histograms are formed by converting the image to the CIELAB color space and using the pixel values from the brightness channel $L*$ and the color channels $a*$ and $b*$ respectively. For texture we adopt the texton approach described in [Martin 2004]. We perform this computation at multiple scales for each level of a four-level Gaussian pyramid for a total of thirteen scales. We also use eight different orientations per scale. To speed-up computation we use an integral-image implementation at each scale and orientation, with different orientations being accommodated by forming the integral image after rotating the original image. Using our MATLAB/ C++ implementation the entire feature extraction procedure takes about 20 seconds for a 321×481 image on a modern 6-core desktop.

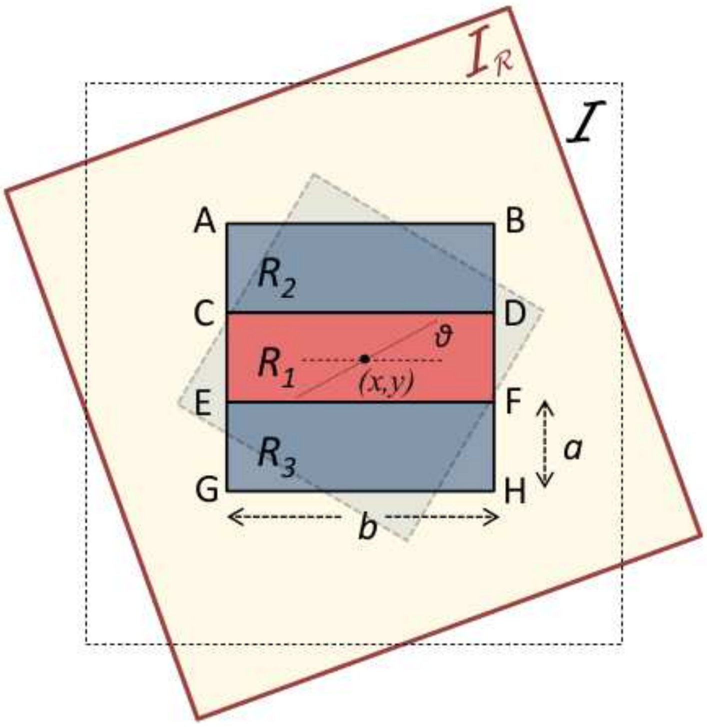


FIGURE 2.15 – Rectangles used to rapidly calculate sums in integral images. Rotation of the filters on the original integral image I is equivalent to filters aligned with axes $x-y$ in a rotated version of the image, I_R. Scale s is equal to $\frac{a}{2}$ (typically $a = 3 \cdot b$).

2.4.1.2 Spectral clustering feature

Spectral clustering [Ng 2002] has been introduced [Shi 2000] and typically used for image segmentation [Cour 2005, Maji 2011]. The basic idea of the algorithm is as follows : we construct a sparse symmetric matrix $\mathbf{W}$ representing pixel affinities, as produced by some spatial cue. Then we use the generalized eigenvectors of the graph Laplacian matrix $\mathbf{L} = \mathbf{D} - \mathbf{W}$, where $D_{ii} = \sum_j W_{ij}$, to create feature vectors for each pixel. These feature vectors are used as inputs to a clustering algorithm, *e.g.* k-means to create the partition of the image into segments.

Arbelaez *et al.* [Arbelaez 2011] compute contour affinities based on the *intervening contour* cue [Leung 1998, Foulkes 2003, Fowlkes 2004], which represents the maximum value of boundary strength along a line that connects two pixels. Using this cue high affinity values are assigned to pixels that are not separated by a strong boundary, resulting in clusters of pixels with similar appearance. Let $\mathbf{p_i}$ and $\mathbf{p_j}$ be two image pixels, $\overline{\mathbf{p_i p_j}}$ the linear segment that connects them, $P_b(\mathbf{p})$ the score of boundary detector at pixel $\mathbf{p}$, and α a constant. For all i, j such that $|\overline{\mathbf{p_i p_j}}| < r$, the elements of the affinity matrix $\mathbf{W^b}$ are given by

$$W^b_{ij} = W^b_{ji} = \exp(-\alpha \max_{\mathbf{p} \in \overline{\mathbf{p_i p_j}}} P_b(\mathbf{p})). \tag{2.3}$$

The authors observe that eigenvectors obtained using this expression, carry useful contour information, and convolve the images formed from each eigenvector with

Gaussian directional derivative filters at eight orientations to obtain the oriented signals $\nabla_\theta \mathbf{v}_k(x, y), \quad \theta \in \{0, \dots, \frac{7\pi}{8}\}$. These "spectral" derivatives effectively locate points of significant change in the eigenvector values while overlooking smooth changes that can lead to incorrect breaking up of large uniform regions.

We adopt a similar approach in order to derive a spectral feature for symmetry. We first train a detector using the histogram features described in the previous section. We use the responses of this first-stage detector to create an affinity matrix $\mathbf{W}^{\mathbf{s}}$, using a function similar to Equation 2.3. Two modifications are necessary : first, we replace boundary strength with the output of the first-stage symmetry detector. Second, we have to use a function appropriately adapted for symmetry. The exponential in Equation 2.3 assigns low affinity to pixels that are separated by a strong boundary. In our case we want to assign high affinity to pixels that lie at the same side with respect to a medial axis ; equivalently, we want to assign low affinity to pixels connected by a line that crosses a high-scoring medial axis. Using an expression similar to Equation 2.3 we construct the affinity matrix $\mathbf{W}^{\mathbf{s}}$ based on symmetry probabilities. Then we solve for the corresponding generalized eigenvectors and convolve the eigenvector images with the Gaussian directional derivative filters. As in [Arbelaez 2011], we combine the results into a single *spectral* component for our symmetry detector using

$$\text{spSym}(x, y, \theta) = \sum_{k=2}^{n} \frac{1}{\sqrt{\lambda_k}} \cdot \nabla_\theta \mathbf{v}_{\mathbf{k}}(x, y), \tag{2.4}$$

where λ_k is the $k-$th larger eigenvalue and λ_0 is a constant vector. In Figure 2.16 we plot the eigenvectors and the spectral component for $n = 5$. The value n of eigenvectors can vary depending on the image content. We found that a value between 30 and 50 gives reasonable results for most images ; in our experiments we use $n = 50$. This spectral component can now be used as an additional feature to enhance our detector's performance. A new, *global* symmetry detector is constructed as a linear combination of the histogram-based detector and the spectral feature :

$$\text{gSym}(x, y, \theta, s) = \sum_i \beta_i H_i(x, y, \theta, s) + \beta_s \cdot \text{spSym}(x, y, \theta), \tag{2.5}$$

where i is used here to index the features described in Section 2.4.1, β_i are their corresponding weights, and β_s the weight of the additional spectral feature. The algorithm we use to learn the weights β_i and β_s is described in Section 2.4.2. Note that we use the same spectral signal for all scales.

We have observed experimentally that the spectral component acts complementary to the histogram gradient features whose responses are mostly driven by local information. The spectral component extracts only the most salient ridges and its contribution consists in reducing clutter after non-maximum suppression and connecting broken parts of the same symmetry axis.

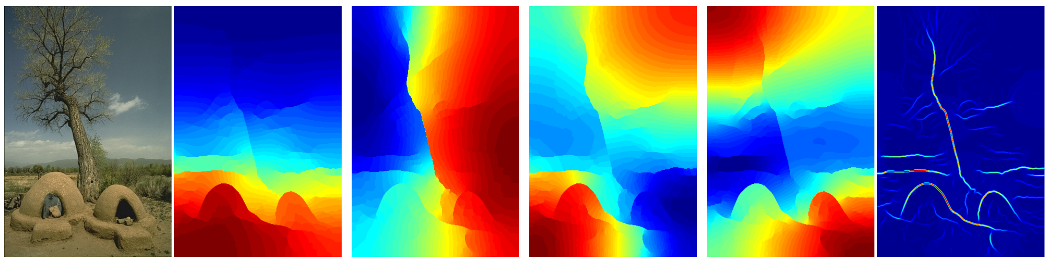

(a) Input image (b) Eigenvector images (c) Spectral cue

FIGURE 2.16 – Spectral symmetry feature : using as input image (a) we setup a generalized eigenvector problem of the form described in Section 2.4.1.2. The first four eigenvectors are shown in the center, while (c) shows the magnitude of the spectral symmetry feature.

2.4.1.3 Feature vector combinations

The process described in the previous sections produces a 13-dimensional feature vector for each pixel : three $L*$ channel histogram-difference features, one for each pair of rectangles ($H^L_{i,j}, (i,j) \in T$, with $T = \{(1,2),(1,3),(2,3)\}$), nine for the color and texture channels ($H^a_{i,j}, H^b_{i,j}$, and $H^t_{i,j}$ respectively, $(i,j) \in T$) and the spectral feature. This feature vector is extracted at all orientations and scales except for the spectral component which varies only across orientations. In the case of grayscale images we omit the features corresponding to the $a*$ and $b*$ color channels resulting in a 7-dimensional feature vector.

The selection of this feature set can be intuitively justified as follows : we believe that points lying on a symmetry axis at orientation θ and scale s exhibit high dissimilarity in terms of some of the used cues when compared to their surroundings (see Figure 2.14). Consequently, for a pixel at location (x,y), $H_{1,3}(x,y,\theta,s)$ and $H_{1,2}(x,y,\theta,s)$ are likely to have high values, while we can expect $H_{2,3}(x,y,\theta,s)$ to have a small value. Different cues and rectangle combinations provide complementary, and potentially also conflicting information ; we now proceed to describe how we learn to combine these measurements for symmetry detection.

2.4.2 Training with Multiple Instance Learning

We use Multiple Instance Learning (MIL) to train our detector as this allows us to treat both scale and orientation as latent variables. In particular, our features are scale- and orientation- dependent. Training our detector in the standard supervised setting would require choosing a scale and orientation combination for every symmetry point. Instead we leave this decision to MIL, which provides a principled, and heuristic-free approach to accommodate the unknown scale and orientation parameters.

In traditional supervised learning we have a training dataset consisting of input-output pairs. In a classification problem the inputs are the *instance examples* $\{\mathbf{x_1}, \mathbf{x_2} \ldots, \mathbf{x_n}\}$ and the outputs are *labels* that denote the class among a set of K possible classes ;

for these labels we use the notation $\{y_1, y_2 \ldots, y_n\}$. Instance examples $\mathbf{x_i}$ typically lie in $\mathbb{R}^d$, y_i in $\{0,1\}$, and the goal is to construct a classifier function $h : \mathbb{R}^d \to \{0,1\}$ that can predict outputs/labels for novel inputs. In the MIL paradigm labels are instead assigned to *sets* of instances called *bags*. The training set consists of the set of bags $\{X_1, X_2 \ldots, X_n\}$ and the bag labels $\{y_1, y_2 \ldots, y_n\}$, where $X_i = \{\mathbf{x_{i1}}, \mathbf{x_{i2}} \ldots, \mathbf{x_{im}}\}$, $\mathbf{x_{ij}} \in \mathbb{R}^d$ and $y_i \in \{0,1\}$ for a binary classification problem. A bag is considered positive if it contains at least one positive instance, whereas a bag is considered negative if all its instances are negative.

The training criterion is expressed in terms of the probabilities P_i assigned to bags that should acquire label y_i. The standard (negative) log-likelihood cost can be written as :

$$C = -\sum_{i}^{N} \{y_i \log(P_i) + (1 - y_i) \log(1 - P_i)\}. \tag{2.6}$$

The particular twist in MIL is that the bag probabilities are expressed in terms of the instance probabilities. The latter are given by a logistic function of the form $p_{ij} = (1 + e^{-(\mathbf{w}^T \mathbf{x})})^{-1}$ while the bag probabilities are obtained by a *noisy-or* combination rule :

$$P_i = 1 - \prod_j (1 - p_{ij}). \tag{2.7}$$

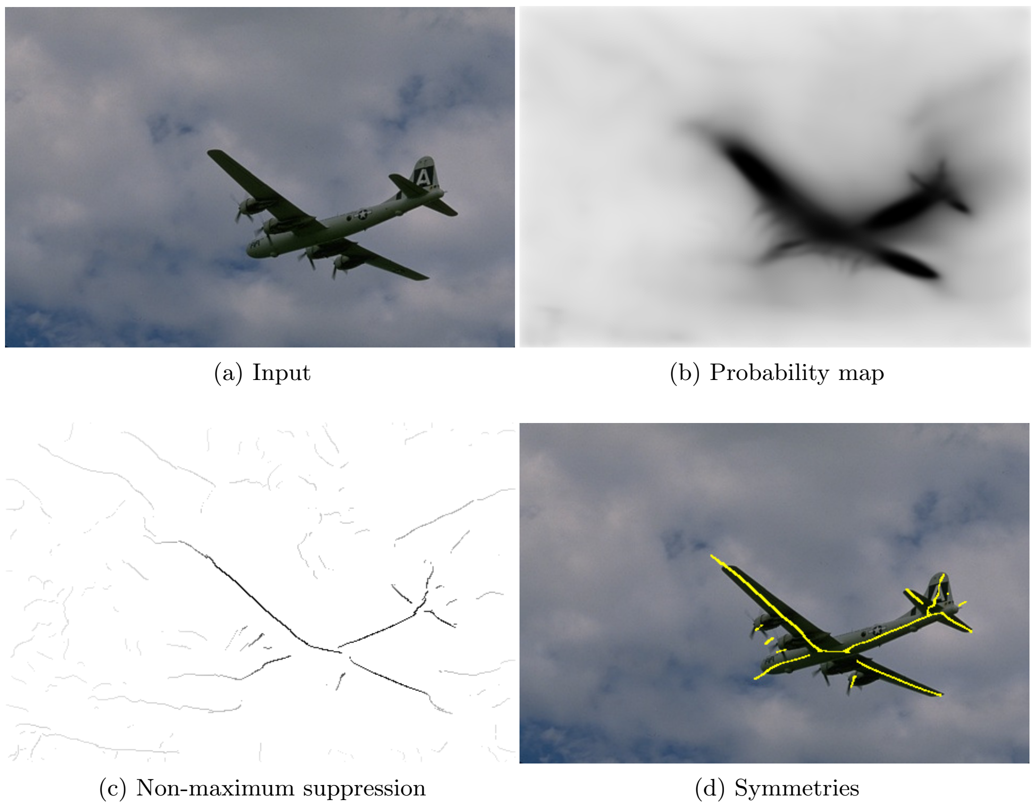
What we present above constitutes a simple setting of MIL training; we refer to [Babenko 2008] for a more thorough presentation of MIL alternatives.

In our problem every image pixel represents a bag of features. The instances contained in such a bag are the feature vectors at all orientations and scales. To give a concrete example, in our experiments we use features at 8 orientations and 13 scales for each pixel, yielding 104 instances in each bag. For the log-likelihood optimization step we use conjugate gradients to compute search directions and quadratic and cubic polynomial approximations to perform line search [Gilbert 1992].

We show the results of the trained symmetry detector in Figure 2.17. We estimate the probability-of-symmetry at every pixel and every scale and orientation combination, resulting in a 4-dimensional symmetry map. These probabilities are combined with the noisy-or rule into an aggregate symmetry response. As this can result in a diffuse response, a non-maximum suppression step is used for thinning prior to thresholding.

2.4.3 Results

We quantitatively evaluate the performance of our detector using the standard precision-recall framework [Martin 2004, Arbelaez 2011, Lim 2013, Dollár 2013]. The constructed ground truth data consist of 5-7 medial axis binary maps per image, resulting from the available segmentations in BSDS300. We begin by thresholding the detector response and matching the result with each ground truth map separately. If a detected positive is matched with at least one of the binary maps, it is classified as true positive. On the other hand, pixels that correspond to no ground truth map


(a) Input (b) Probability map

(c) Non-maximum suppression (d) Symmetries

FIGURE 2.17 – Processing steps from the initial image to the final symmetry map.

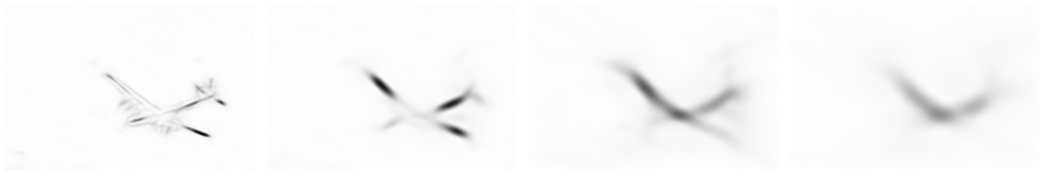


FIGURE 2.18 – Probability responses before non-maximum suppression at increasing scales from left to right.

are declared false positives. Perfect recall is achieved when every ground truth point corresponds to some point declared positive by our system. We allow correspondence between detected positives and pixels neighboring to the ground truth positives to take into account small localization errors in the ground truth.

We can assess the hardness of the task at hand by assessing human performance. A proxy for human performance can be obtained by evaluating the skeletons delivered by human segmentations. For this, we associate ground truth data with an F-measure value by picking sequentially each on eof the binary maps and treating it as a thresholded detector output. This is followed by matching with the remaining binary maps for the same image in the same way as described above. The score delivered by this procedure was F = 0.73 and the corresponding iso-curve is shown in Figure 2.19 ; we can interpret this as an upper bound on what we can expect to achieve by our machine-generated symmetry maps.

In Figure 2.19 we show the precision-recall curves associated to our method, obtained by thresholding the detector response at different values. In order to confirm the importance of each separate feature used to train our detector, we compare with the results of detectors trained with a subset of the available features. We also compare its performance against the Lindeberg ridge detector, implemented as in [Kokkinos 2006], and we see that we attain a steadily better maximum F-measure throughout the precision-recall spectrum. On the same plot we also compare the

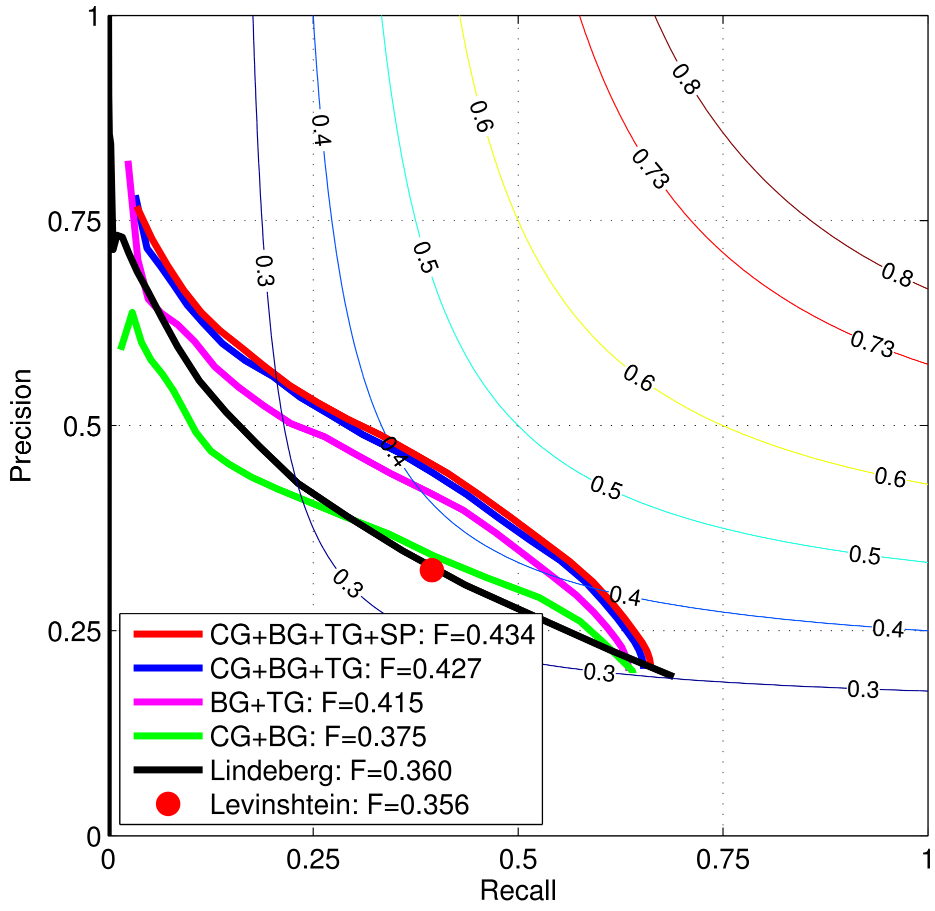


Figure 2.19 – Quantitative comparison of our detector with other methods. We plot the performance of our detector trained with different feature combinations to illustrate the boost due to color and spectral features. CG : color histogram gradient, BG : brightness histogram gradient, TG : texture histogram gradient, SP : spectral feature. The ground truth F-measure is represented by the red iso-curve.

performance of our algorithm against the more recent approach of Levinshtein et al. [Levinshtein 2009], even though they do not deal with exactly the same problem. In their work they learn affinities between adjacent superpixels of the original image and cluster them to form symmetric parts. These parts are then clustered anew in groups belonging to the same object, and the symmetry axes of these regions are extracted as the major axes of appropriately fitted ellipses. Their algorithm effectively comes up with a parsing of the image into regions, while our algorithm stops at a level before that and provides longer, continuous contours, even with large curvature, as demonstrated in Figure 2.20b. Finally, the authors in [Levinshtein 2009] train their detector on the Weizmann dataset and not in natural images, as we do.

Since Levinshtein's algorithm returns binary results we do not include a precision-recall curve in Figure 2.19. We evaluated its performance on the testing dataset, treating the binary image as a thresholded probability map and seeking pixel correspondences with the ground truth ridge maps in the same way as for the other methods. This gave an overall F-measure of 0.355. Apart from the performance difference we argue that our approach has merit due to relying on local cues, instead of using region detection as a front-end, while also providing continuous contours, which can be subsequently broken and re-grouped at will. As such, our method does not make "early commitments" from which it may be hard to recover post-hoc.

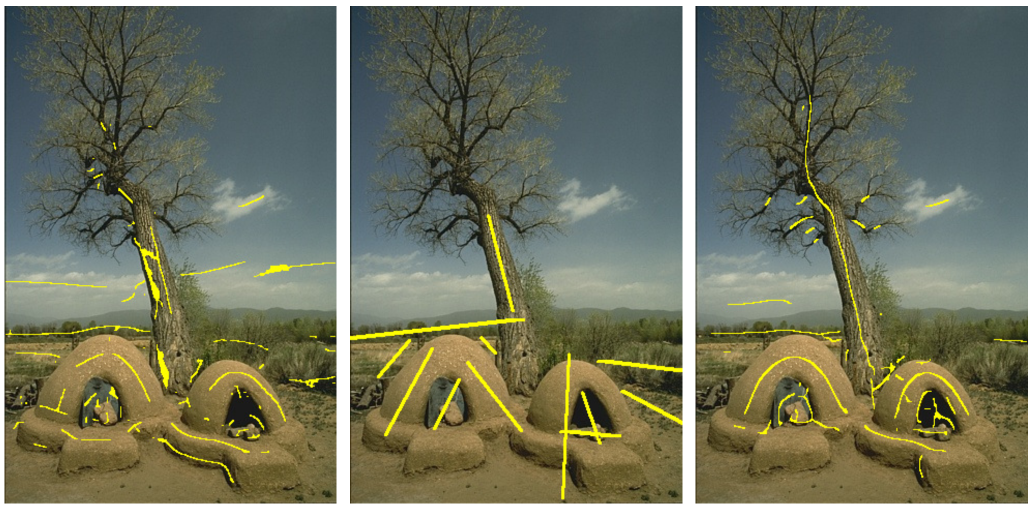

(a) Lindeberg (b) Levinshtein (c) This work

FIGURE 2.20 – Qualitative comparison of our detector with other methods. For (a) we use the implementation of [Kokkinos 2006] and for (c) the available code of [Levinshtein 2009]. More qualitative results are included in Figure 2.21.

2.5 Speeding-up Medial Axis Detection Using Random Forests

The method we introduced in Section 2.4 delivers promising results, detecting medial axes in a variety of challenging natural images. However, as we discussed in Section 2.4.1, the feature extraction process at multiple scales and orientations renders the whole system relatively slow at test time. This can be an important

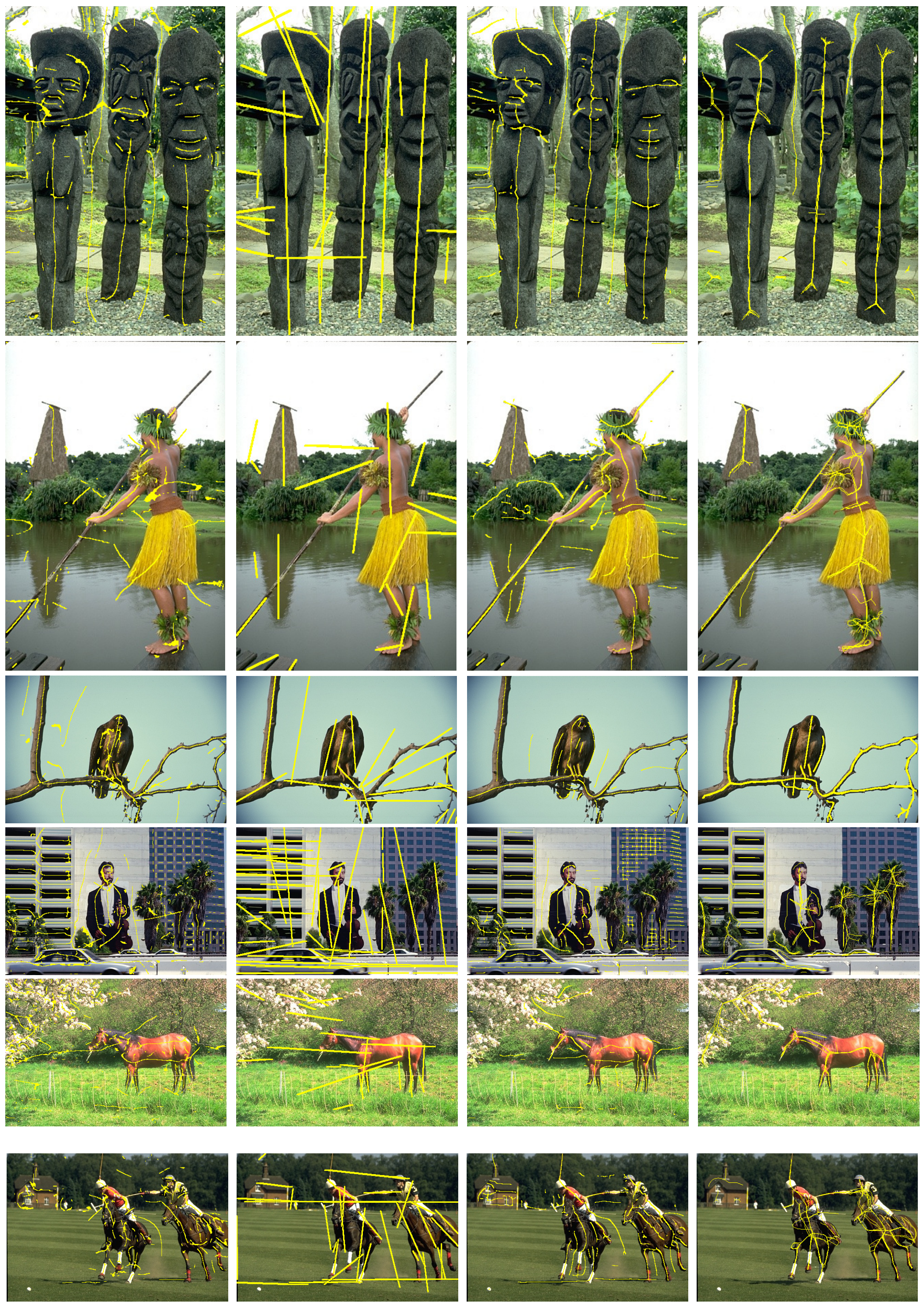

(a) [Lindeberg 1998] (b) [Levinshtein 2009] (c) This work (d) Ground-truth

FIGURE 2.21 – Qualitative results for all three compared methods and respective ground truth (experiments on SYMMAX300).

drawback if *e.g.* we want to use our detector as a feature extraction component in a more complex pipeline. In this section we discuss an alternative approach that uses random forest classifiers to decrease running time from approximately $20sec$ to less than $1sec$ on a modern CPU. We start with a brief overview of decision forests in Section 2.5.1 and the features used in Section 2.5.2. We then discuss how we adapt sketch tokens [Lim 2013] for medial axes (Section 2.5.3) and in Section 2.5.4 we present experimental results on SYMMAX500. In Section 2.5.5 we show that by grouping individual pixel responses into longer contours, we can reduce spurious responses and improve performance.

2.5.1 Decision Forests

Decision trees for classification and regression were popularized in Breiman [Breiman 1984], while an algorithm for training optimal decision trees from data was proposed by Quinlan in [Quinlan 1993].

Decision trees were originally treated as individual entities ; however, it was eventually shown that better accuracy and generalization can be achieved by using trees within *ensembles* of learners, as in Schapire's boosting algorithm [Schapire 1990, Freund 1997]. In these works a linear combination of many "weak" classifiers – classifiers that perform a little better than chance – is used to build "strong" classifiers.

The same idea can be applied in the context of trees : training individual random trees and fusing their outputs yield a decision *forest*. Decision forests and in particular, *random* decision forests have become very popular in the machine learning, computer vision and medical imaging literature [Lepetit 2006, Bosch 2007, Caruana 2008, Criminisi 2011, Girshick 2011a, Shotton 2013, Gall 2013, Lim 2013, Dollár 2013, Zitnick 2014, Hallman 2015], combining strong performance and high computational efficiency.

We now describe the basic notions regarding decision trees and random forests. A tree is a collection of nodes and edges organized in a hierarchical structure. We can divide the nodes into three types, according to the number of their parent and children nodes : i) *root* (two children, no parent), ii) *internal* or *split* nodes (one parent, two children), iii) *leaf* nodes (one parent, no children nodes) ; see Figure 2.22.

A *decision* tree, as its name suggests, is a tree used for making decisions. In the context of computer vision, the input to the tree could be an image or a part of an image, and the objective could be recognizing if the object depicted in the image is of a specific class or not. The input is pushed from the root, through the internal nodes, to the leaves, being subjected to a *test* at each internal node. It is then sent to the left or right child of the node, depending on the result of each test ; this process is repeated until we reach a leaf. Usually the leaf nodes contain a predictor (e.g. a classifier, or a regressor) that associates an output (*e.g.* a class label) to the input. In the case of forests many tree predictors are combined together, for example by averaging, to form a single forest prediction.

Training of each tree is performed offline, and is responsible for choosing the parameters of the split functions associated with the internal nodes, as well as the

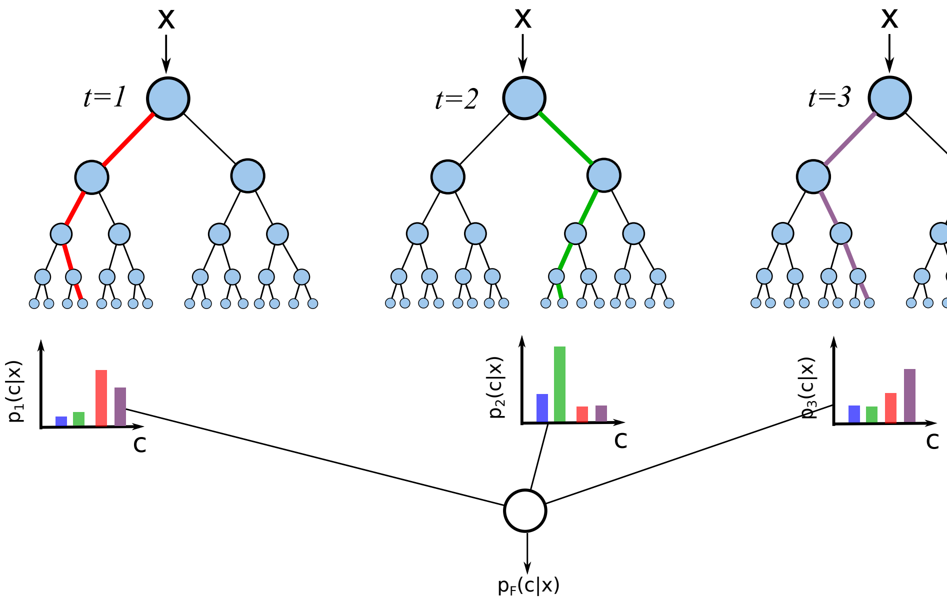


FIGURE 2.22 – The outputs of multiple decision trees are combined to form the forest output. The colored edges show the path followed by the input from the tree node to one of the leaves, for each tree.

leaf predictors. Parameter optimization is performed using some information gain criterion, *e.g.* Shannon entropy, and continues until a predefined stopping criterion is satisfied. Possible stopping criteria are a maximum branch depth D or a minimum information gain. Note, also, that trees are independent to one another and, as a result, can be trained in parallel.

Randomness is a key aspect of decision forests, hence the term *random forests*. Each tree in a random forest is randomly different than all the other trees, which makes individual tree predictions de-correlated, and improves generalization. Randomness also makes forest classifiers more robust with respect to noisy data. Randomness is injected into the trees only during *training*, in one of two ways : i) using a random subset of the training data to train each tree (*e.g.* bagging) ; ii) using a small, random subset of the parameters values available at each node. The latter enables us to train each tree using the full training dataset, whereas bagging is more efficient.

At test time, the input is provided simultaneously to all trees in the forest. Supposing we have T trees, the forest output is the combination of T single tree predictions, as shown in Figure 2.22. The most common way to combine tree outputs is by simple averaging :

$$p(c|\mathbf{x}) = \frac{1}{T}\sum_{t=1}^{T} p_t(c|\mathbf{x}) \tag{2.8}$$

where $p_t(c|\mathbf{x})$ is the posterior distribution of class c, given input $\mathbf{x}$, obtained by the t-th tree.

2.5.2 Features

We use multiple channels of features, consisting of color, gradient, and oriented gradient information [Dollár 2009], as in [Lim 2013]. We compute color features after transforming the input image into the CIE-LUV color space, and normalized gradients at varying orientations and scales, for a total of 14 channels. We also extract self-similarity features for each channel, capturing texture information based on color or gradient information. We use the same parameter values as in [Lim 2013], so we refer to that paper for more details.

We also use the real-time boundary detector of Dollar et. al. [Dollár 2013] as an additional cue in our feature pool, increasing the number of channels to 15, at a negligible extra cost. As we previously mentioned in Section 2.2 boundaries and medial axes are strongly correlated, so we expect that this additional cue will lead to improved detection accuracy. Our assumption is validated in Section 2.5.4 were we show results obtained by training our model using these two different settings.

2.5.3 Sketch tokens for medial axes

Sketch tokens were introduced in [Lim 2013] as a local, contour-based representation for mid-level features. In that work the authors exploit human-generated boundary contours that are readily available in BSDS500 to define a set of token classes. These classes are obtained in a bottom-up manner by clustering the ground truth patches and aim at capturing the wide variety of local contour structures that exist in natural images, including straight lines, t-junctions, corners, curves, and parallel lines.

In the original work of [Lim 2013] on Sketch tokens the authors consider that we have a set of images I and a corresponding set of binary images B, containing hand-drawn contours. They collect a set of fixed-size square patches $b \in B$, containing a positive label at their center pixel, and cluster them using k-means. The resulting cluster means constitute the k token classes that are used to represent varying local contour structure. In this work we use the dataset described in Section 2.3, and adapt the procedure outlined above to finding a dictionary of symmetry patterns.

One major difference between boundaries and symmetry axes is that the latter always entail *scale* information, as they imply the existence of bilateral boundaries at approximately equal distances with respect to the axis. Using image patches of a fixed size ignores this type of information, because the boundaries corresponding to a symmetry axis can possibly reside outside the patch. The challenge that arises is constructing a classifier that will be able to capture symmetry information at various scales.

In order to keep our approach simple and efficient, we keep the patch size fixed at 35×35 pixels, but only accumulate patches that correspond to fine-scale symmetries. More precisely, we use patches containing axes whose corresponding boundaries are no more than 12 pixels away, which results in approximately $3 \cdot 10^4$ patches. In our experiments we found that using this subset of training samples does not hinder

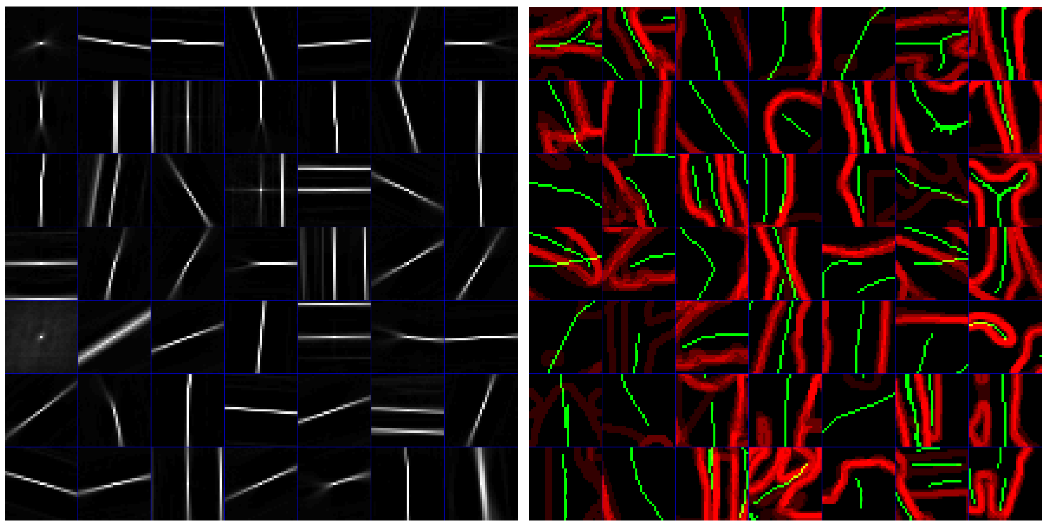


FIGURE 2.23 – **Left :** Examples of sketch tokens for symmetry. **Right :** Positive ground truth patches for symmetry(green) and their corresponding boundaries (red).

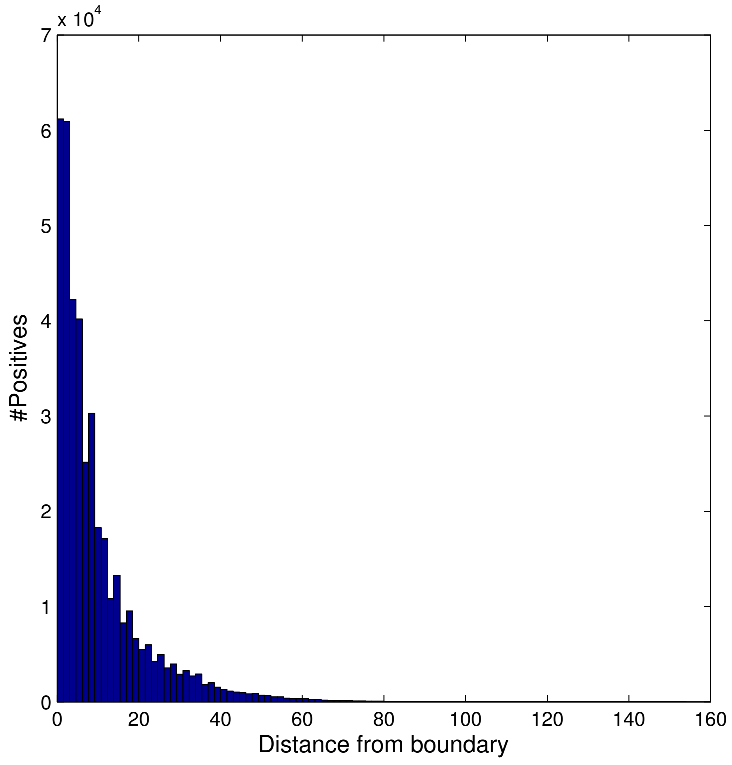


FIGURE 2.24 – Histogram of symmetry scale distribution.

detection performance. This is not surprising, given that the majority of symmetry axes occurring in our dataset correspond to fine scales, as shown in Figure 2.24, which in turn is anticipated from the scaling laws of region sizes in images, discussed e.g. in [Lee 2001] ; there it is shown that the size of regions in images needs to follow a

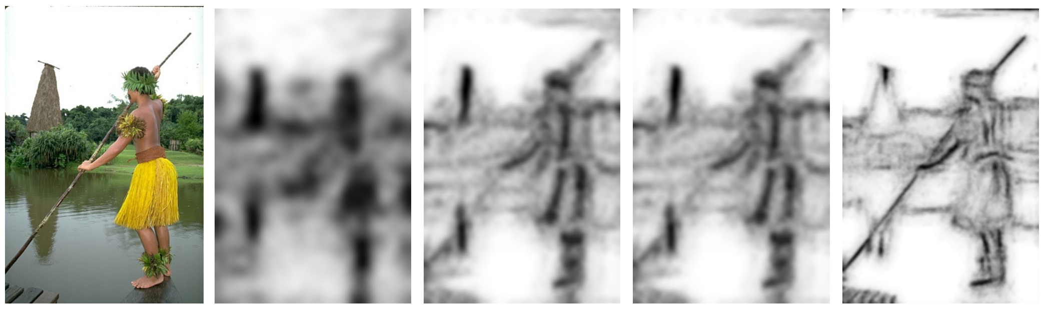


FIGURE 2.25 – Symmetry detection response at increasingly finer scales from left to right (black color corresponds to higher probability values).

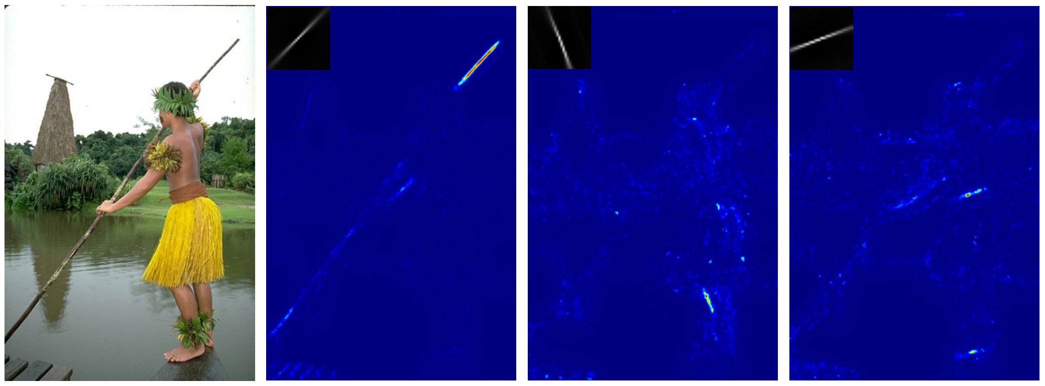


FIGURE 2.26 – Individual token responses. Different token classes capture symmetry information at varying orientations.

distribution of the form $1/r^2$, where r is the region size.

2.5.4 Results

We test our detector on the images of the BSDS500 dataset [Martin 2001], using our annotations as the medial axes ground truth. The BSDS500 dataset contains 200 training images, 100 validation images, and 200 testing images. We train the final model on the 300 *train+val* images, and evaluate detection performance on the 200 testing images. In Figure 2.27 we compare our contour detection method against our MIL-based variant, in terms of precision-recall curves. To conduct a fair comparison, we re-trained our MIL-based medial axis detector introduced in Section 2.4 on the new, manually annotated SYMMAX500 dataset.

The detector trained with random forests (RF) achieves the highest overall F-measure, improving over all other methods that rely on local cues, as well as our MIL variant, that uses additional global information. The difference in performance is most striking in the high-recall regime, where it retrieves a much higher percentage of true positives. More importantly though, our multi-scale detector runs at $0.7sec$ on a standard dual-core laptop ($1.5sec$ for the single core implementation). This makes it $30\times$-$40\times$ faster than the MIL approach using color and texture features, and $200\times-300\times$ faster than the spectral variant, offering a vast improvement in terms of efficiency.

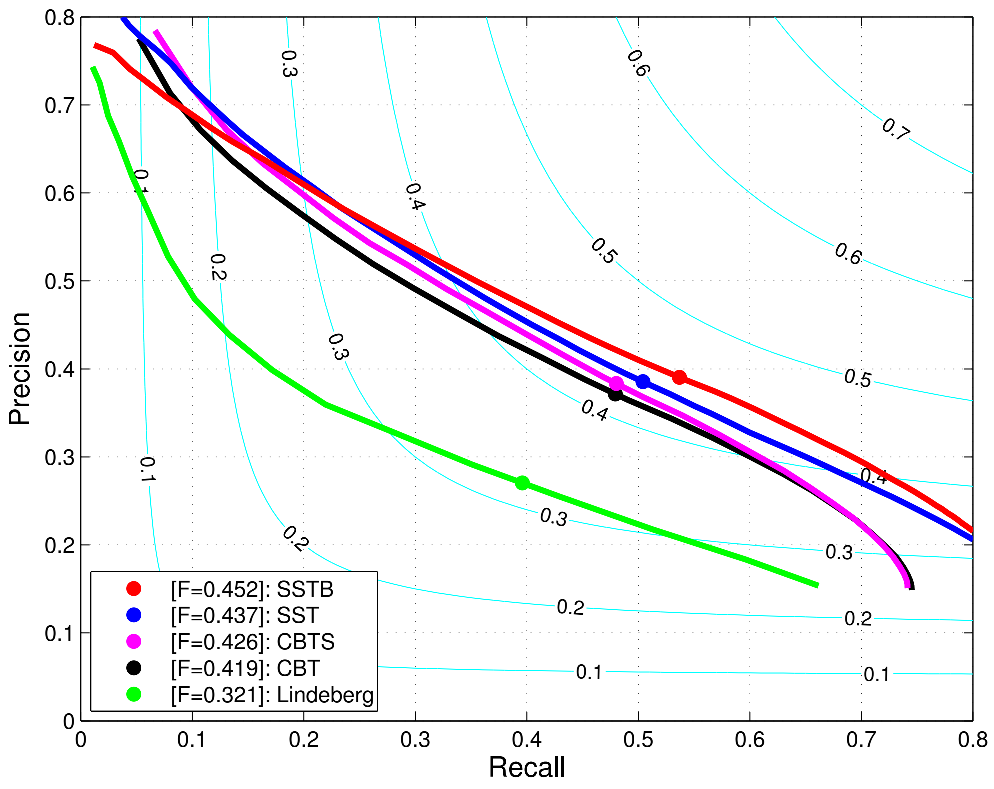
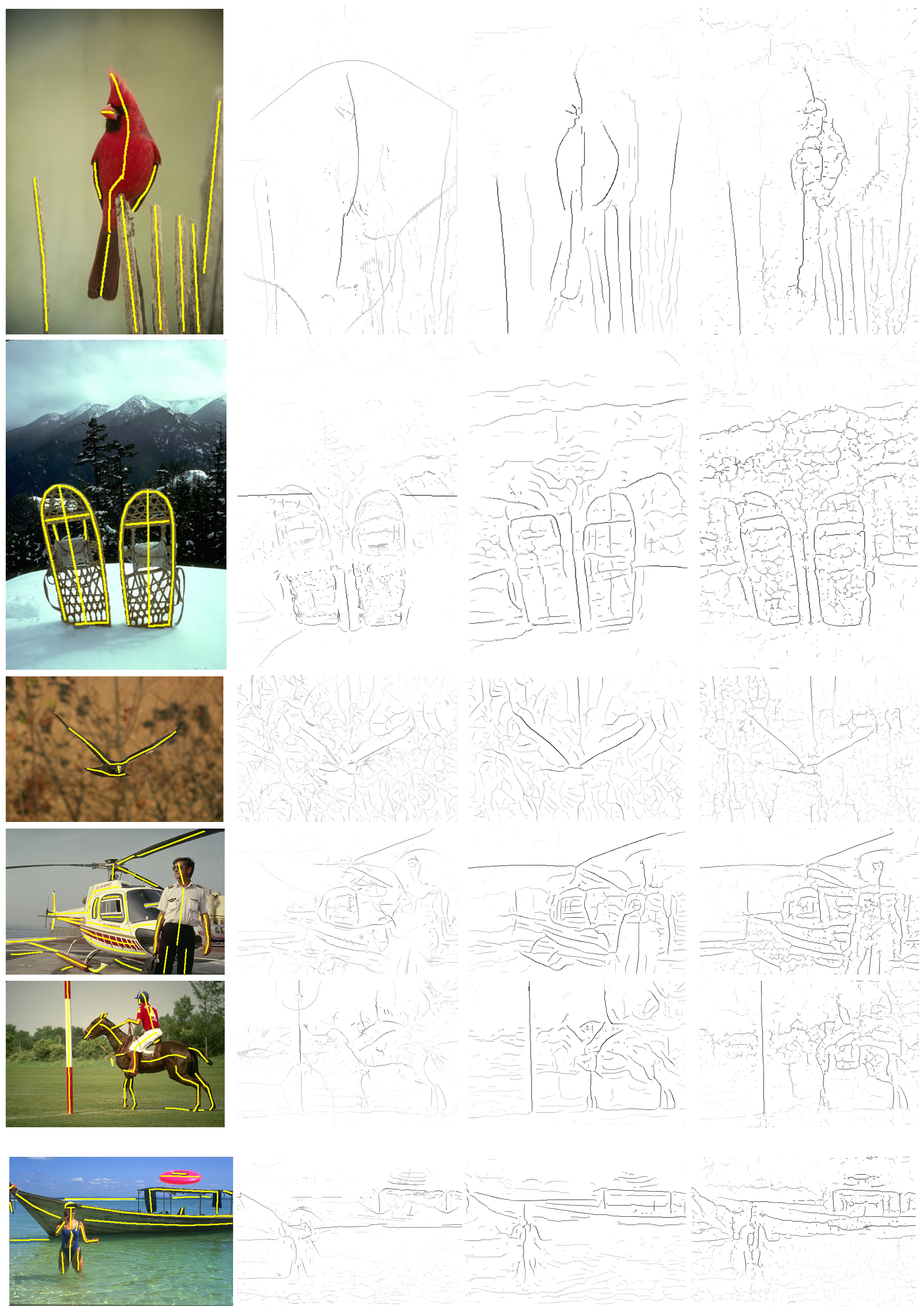


FIGURE 2.27 – Quantitative comparison of our detector's performance (SST/SST-Boundary) to earlier, MIL-based variants (CBT-CBTSpectral).



(a) Ground-truth (b) [Lindeberg 1998] (c) MIL (d) Random Forests

Figure 2.28 – Qualitative results for all three compared methods and respective ground truth.

2.5.5 Contour grouping

The system we have described so far is evaluated densely, producing a symmetry response at every pixel independently. Depending exclusively on low level information, however, leads to an increased number of false positives, resulting in fragmented contours or isolated, spurious responses. In Section 2.4.1.2 we described how incorporating global information through an additional spectral clustering cue can partly compensate for this issue. Such a step, unfortunately, comes at a significant computational cost, ranging in the minutes.

We adopt an alternative approach and try to improve the quality of our detections by grouping isolated symmetry responses into meaningful contours. Elongated, salient contours could serve subsequent processing steps, such as object recognition, and multiple works in the literature [Felzenszwalb 2006, Zhu 2007a, Kokkinos 2010] have observed the positive impact of grouping in the high-precision regime ; our own results add to this consensus. We employ the framework and software provided in [Kokkinos 2010], which treats contour grouping as a fractional-linear programming problem. We apply this algorithm on the thinned probability maps we compute using our original symmetry detector adding the extra boundary feature from [Dollár 2013].

In Figure 2.30 we show precision-recall curves after grouping for the 100, and 200 most salient curves (G_{100}, and respectively G_{200}). Grouping boosts performance in the high-precision regime in all three cases but harms recall rate. Another remark is that as we increase the number of retrieved contours, the precision gains diminish, but the same applies to the reduction of the recall rate.

These observations suggest that there is a trade-off between precision gains induced by grouping, and the overall performance of the detector. In effect, we can sacrifice high recall in favor of precision, if it proves critical for the task at hand. In our case, keeping the 200 most salient contours seems to be the "sweet spot" where we get noticeable improvement in precision and minimal recall drop. The suppression of false positives is also reflected qualitatively in Figure 2.29 where we compare symmetry responses before and after grouping.

2.6 Conclusions

In this chapter we have described a novel technique to extend the popular medial axis transform of binary shapes to natural images. Departing from the classical paradigm of computing an analytic transformation of the object boundary to its medial axis, we proposed a *learning*-based approach to the problem.

We have constructed a symmetry axis dataset on top of the Berkeley Segmentation Dataset (BSDS500), manually annotating medial axes of prominent objects. We make this dataset publicly available and hope it will spur further research in symmetry detection from natural images. As for the training learning algorithms, we considered two settings. First, we used multiple instance learning to accommodate for the unknown scale and orientation of symmetry axes during training by treating

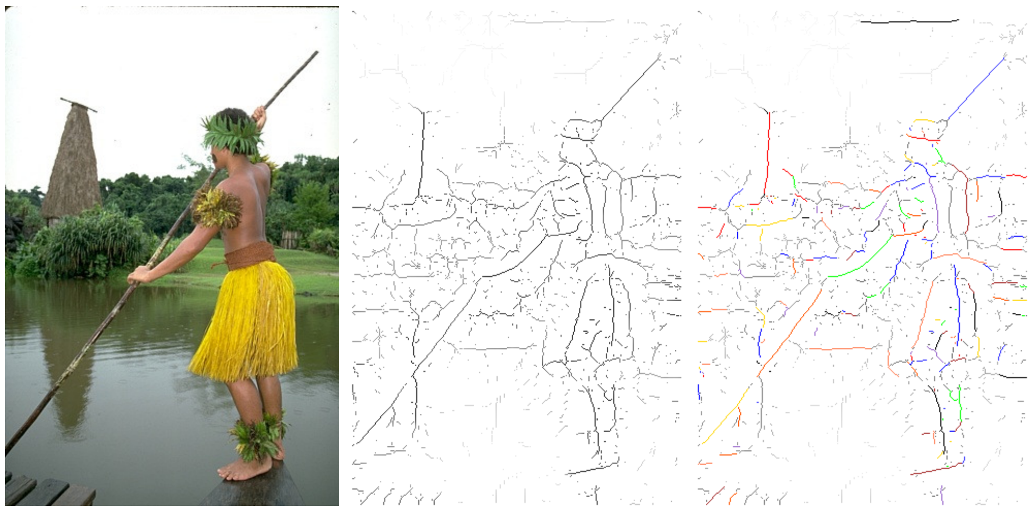


FIGURE 2.29 – From left to right : Original image, symmetry response without grouping, grouping 200 most salient contours (better viewed magnified and in color).

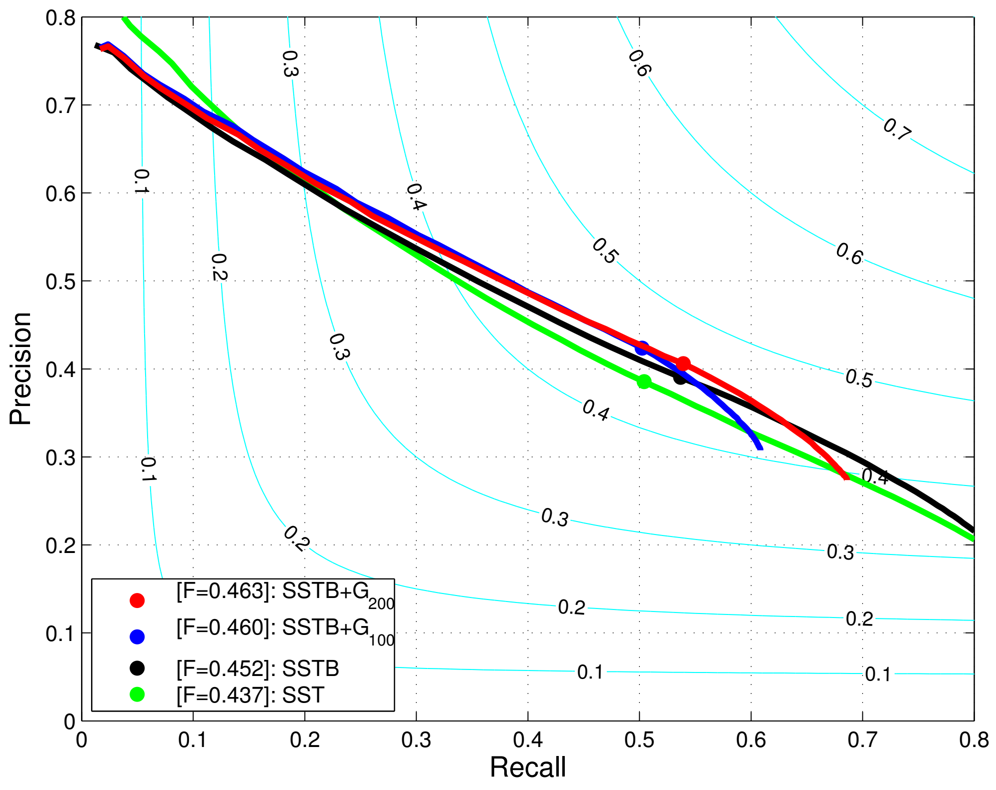


FIGURE 2.30 – Exploration of the effect of grouping. We observe that we largely outperform other detectors, while being an order of magnitude faster than CBT (color-brightness-texture) and two orders of magnitude than CBTS (color-brightness-texture-spectral) - grouping yields an additional boost in performance. SST stands for our Symmetry Sketch Token detector.

both as latent variables. Second, we trained a random forest classifier using only fine-scale examples that is evaluated in a feature pyramid of the input image at test time.

We have evaluated our approach on our dataset and validated that it outperforms existing works on symmetry detection. We demonstrate the importance of the additional features used, namely color and texture, which enhance the more common gray-scale information. In addition, our random forest variant further improves performance, while significantly reducing running time with respect to our MIL algorithm. Finally, we have grouped individual pixel responses to form longer contours and noticed that this procedure has a beneficial effect in terms of detection precision.

One of the main merits of our approach is its flexibility ; our detector can be tailored to a specific application through training over an according ground truth dataset and application-driven features. This can lead to a significant improvement in performance on the task under question, which can range from medical image analysis to body part detection. Furthermore, grouped symmetry contours can be used as an object presence indicator, complementing similar ideas based on boundary contours [Zitnick 2014] ; we intend to explore these directions in future work. Recently, the Holistically-Nested Edge Detector by Xie *et al.* [Xie 2015] significantly advanced the state-of-the-art for boundary detection on the BSDS500 employing multi-scale convolutional neural networks. We expect that with appropriate modifications, a similar system will yield analogous improvements for medial-axis detection as well.

2.7 Appendix

In this appendix we include some additional information regarding the GUI we used to construct the manual annotations for SYMMAX500 in Section 2.3.2. We also elaborate more on the protocol we used to annotated pixels as belonging to a medial axis or not.

2.7.1 An Interface for Annotating Contours in Images

The GUI we used was implemented entirely in MATLAB and supports a set of elementary actions, such as drawing a straight line, a poly-line or a curved contour. It also permits the user to edit existing annotations by pruning parts of the ground truth, or deleting it altogether and starting from scratch. On top of that, it offers a selection of morphological filtering choices that can be directly applied to the current annotation ; this way the user can easily process and visualize several versions of the same ground truth map during the annotation procedure. An example of how the GUI interface looks when annotating an image is shown in Figure 2.31.

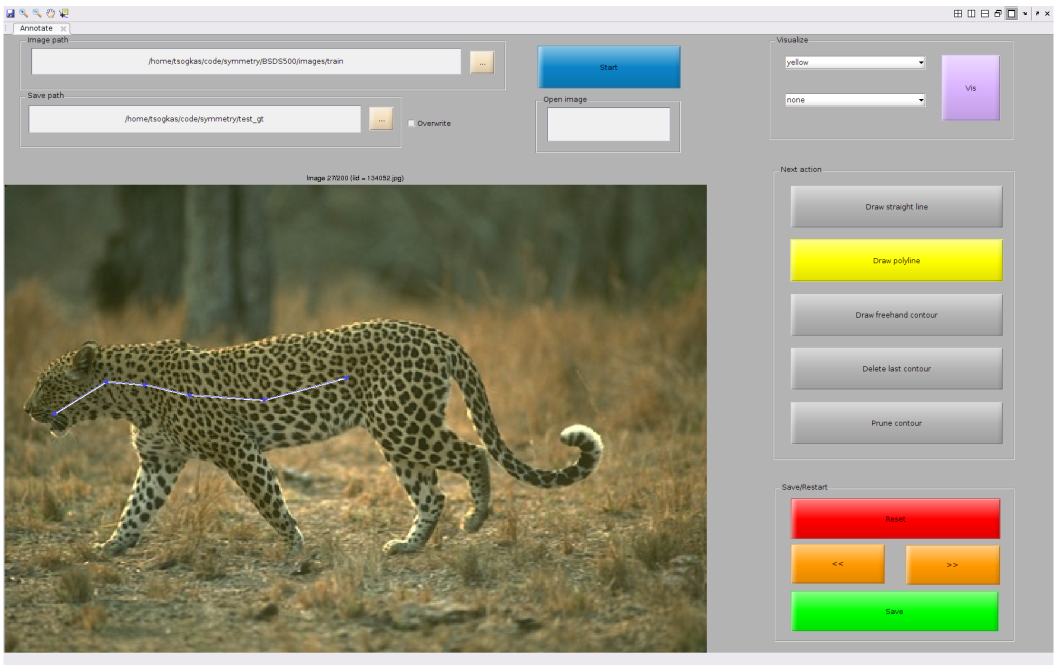


Image path: /home/tsogkas/code/symmetry/BSDS500/images/train

Save path: /home/tsogkas/code/symmetry/test_gt

☐ Overwrite

Open image:

Visualize: yellow, none

Image 27/200 (iid = 134052.jpg)

FIGURE 2.31 – Interface of the graphical user interface we used to manually annotate SYMMAX500 (best viewed magnified and in color). Our GUI is flexible, enabling the user to draw straight lines, poly-lines, and free hand contours. Other useful functionalities include deleting the last drawn contour, deleting a manually selected part of the annotation, fully resetting the annotation procedure, and browsing through images in a folder. It also supports different colors for visualization, as well as filtering operations from the MATLAB function `bwmorph`.

2.7.2 Annotation Rules

As we discussed in Section 2.1, formally defining symmetry in not easy. In our work, we are interested in detecting medial axes of elongated parts, which amount to local and approximate bilateral symmetry. However, even in this restricted setting, annotating natural images is challenging. For instance we need to decide whether we should distinguish between foreground and background when selecting symmetry positives, or how we should treat very thin symmetric structures.

In addition, objects usually appear in heavily cluttered scenes or under varying illumination conditions, their parts being occluded by other objects or shadows, further perplexing the annotation of medial axes. As humans, we can often overcome these difficulties with ease using context information ; *e.g.* recognize medial axes that are broken by occluding objects ; or identify the true medial axis of a symmetric structure that is not fully visible because of shadowing effects. In order to remain consistent while annotating the complete dataset, we decided to avoid using such complex reasoning. Instead, we follow a set of simple rules when declaring image pixels as symmetry-positive :

What to annotate ? Annotated pixels should be symmetry axes of elongated structures that are sufficiently distinguishable from their surroundings in terms of color (brightness if the image is grayscale) and/or texture. In particular we follow an "annotate what you see" approach, which means that in the case of shadowed or occluded objects, we do not make any context-based assumptions regarding the occluded/shadowed part ; we show an example in Figure 2.32. In other words, if the annotator needs to scrutinize an area to decide if it contains a symmetry axis, this means that the area should not be annotated (*i.e.* we aim at "pre-attentive" symmetry detection). Symmetric structures can resemble "2D ribbons", such as rectangles, triangles, ellipses, cylinders, or deformable "snakes", but not circles or squares. Please see Figure 2.32 for an example.

Foreground *vs.* background : Symmetric structures typically correspond to semantically distinct objects or object parts (e.g. tree trunk, arm, leaf, eyebrow). We also accept background structures that conform to the first rule, such as the river in Figure 2.32. We ignore large uniform background areas that do not contain salient entities, such as the sky in the same image. In other words, we put no symmetry axes on "stuff".

Minimum thickness : We ignore very thin structures (under 6-7 pixels wide) to ensure a clear distinction between boundaries and thin symmetric parts.

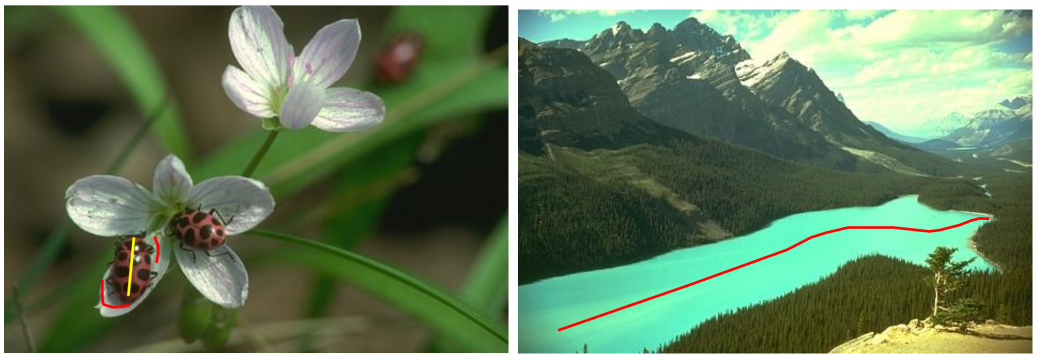


FIGURE 2.32 – Examples where annotation might be ambiguous to a human user. For instance, we *know* the flower leaf (resembling an ellipse), has a symmetry axis that is hidden by the ladybug on it. The river in the bottom figure is technically "background", yet we annotate it as a salient, elongated structure that is distinct with respect to its surroundings.

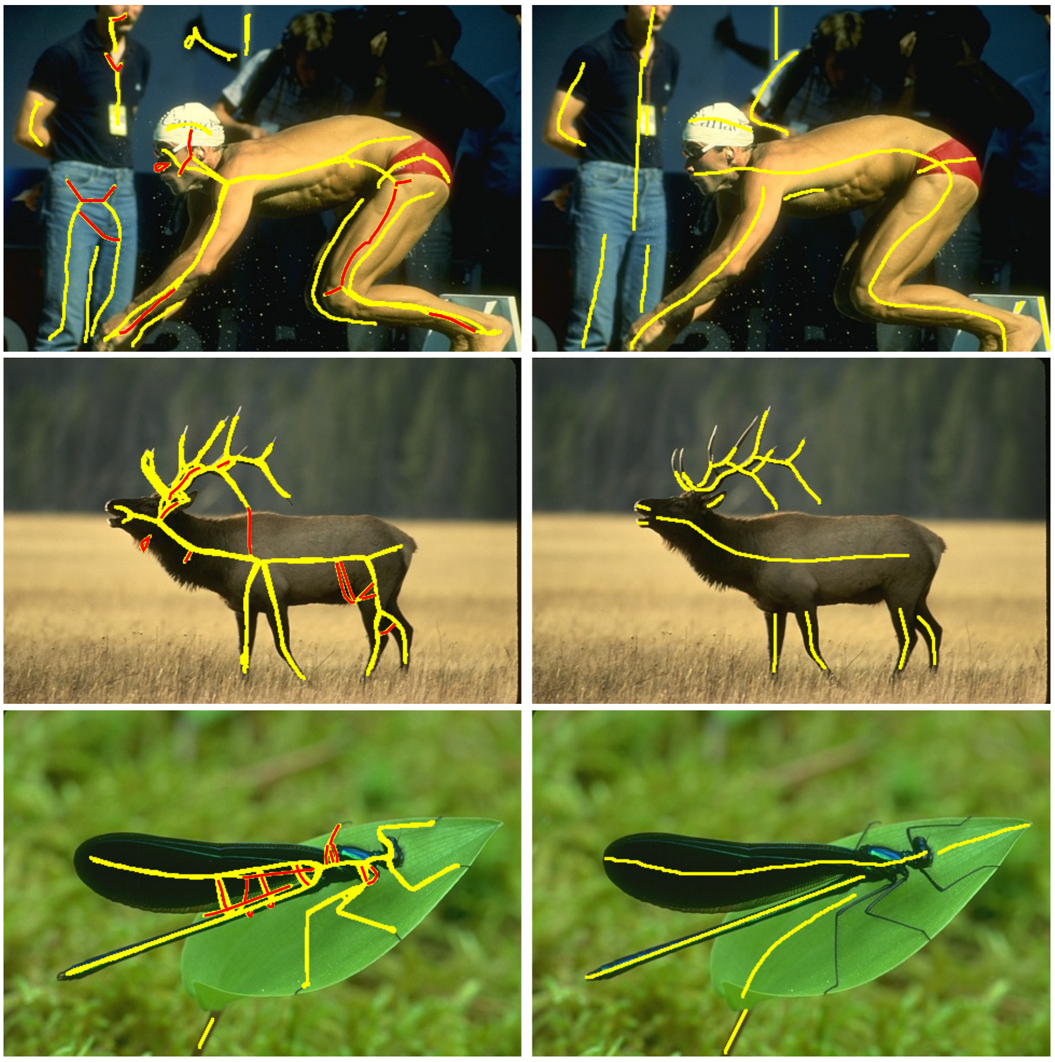


FIGURE 2.33 – More ground truth comparisons between SYMMAX300 and SYMMAX500.

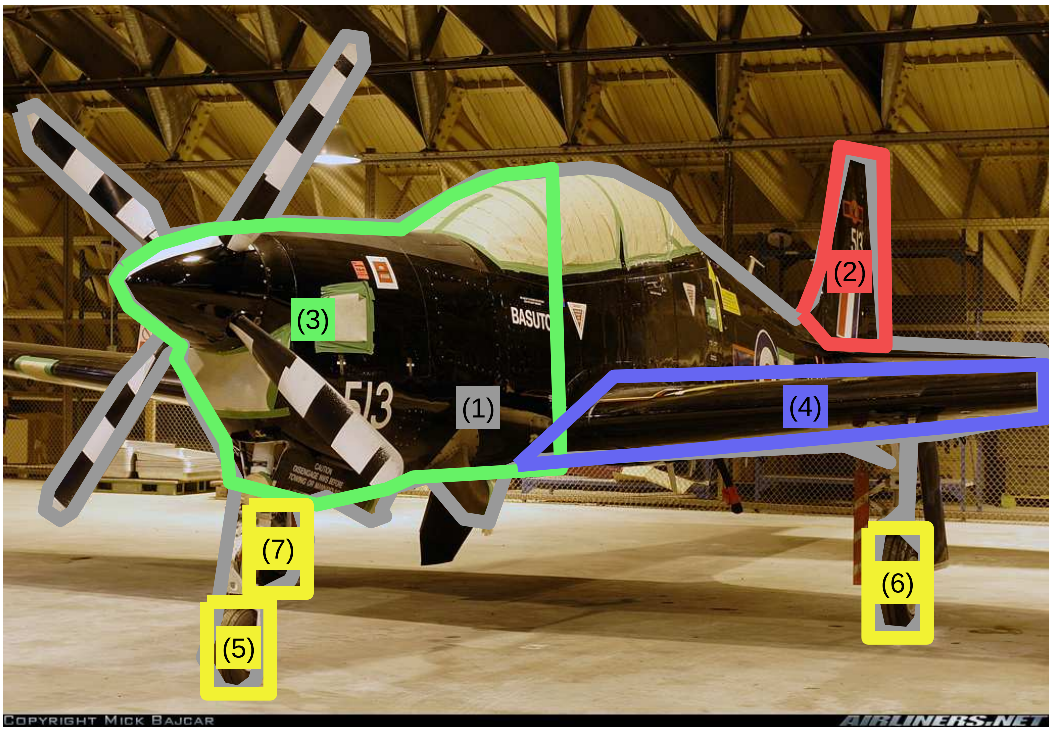
Chapter 3

Efficient Detection of Object Parts and Their Role in Object Detection

Contents

In Chapter 2 we investigated methods for detecting medial axes which are generic descriptors of object shape and are independent of the object class. In this chapter we turn our attention to class-specific mid-level representations and more specifically *object parts*. After motivating the use of parts for detailed object modeling in Section 3.1 we develop a coarse-to-fine technique that organizes and evaluates learned HOG templates in a hierarchical tree, improving detection speed at least by a factor of 4 with negligible loss in accuracy. In the second part of the chapter we turn to a complementary method of increasing the accuracy of part-based models ; in particular, we substitute histograms of gradients with convolutional features in a DPM pipeline for object detection.

Our research on efficient part detection was part of a project that started at the Johns Hopkins CLSP Summer Workshop 2012, resulting in the construction of a high-quality Objects In Detail dataset (Oid), presented in a CVPR publication [Vedaldi 2014]. Our work on integrating CNN features and DPMs was done in parallel with [Girshick 2014b] and was published as a workshop paper in ECCV [Savalle 2014].



1 **airplane** facing-direction : SW ; is-airliner : no ; is-cargo-plane : no ; is-glider : no ; is-military-plane : yes ; is-propellor-plane : yes ; is-seaplane : no ; plane-location : on ground/water ; plane-size : medium plane ; undercarriage-arrangement : one-front-two-back ; wing-type : single wing plane ; airline : UK–Air Force ; model : Short S-312 Tucano T1 2 **vertical stabilizer** tail-has-engine : no-engine 3 **nose** has-engine-or-sensor : has-engine 4 **wing** wing-has-engine : no-engine 5 **undercarriage** cover-type : retractable ; group-type : 1-wheel-1-axle ; location : front-middle 6 **undercarriage** cover-type : retractable ; group-type : 1-wheel-1-axle ; location : back-left 7 **undercarriage** cover-type : retractable ; group-type : 1-wheel-1-axle ; location : back-right

FIGURE 3.1 – **Beyond object detection : detailed descriptions.** An example annotation taken from the AirplanOID dataset [Vedaldi 2014]. The object is described in terms of its parts and attributes. Note that the attributes can be directly related to specific parts.

3.1 Introduction

Objects can often be decomposed into smaller parts that obey certain geometric constraints, have relatively consistent appearances and shapes, and usually carry specific semantic connotations. Identifying parts can facilitate object-level classification and detection but becomes particularly important for understanding objects and their properties in detail.

For example, glancing at Figure 3.1 we recognize a plane, but we can also recognize several properties of the object –usually referred to as *attributes*– that act as modifiers of object parts. For instance, we can see that the plane has “two wings, retractable single-wheeler undercarriages under the wings, pointy nose with a fourblade, black-and-white, striped propeller, a small round cockpit window, *etc.*". The same applies for most object classes, *e.g.* a person can have short hair, a round nose, a long beard, or bulging eyes. The relative position and arrangement of a person's arms and legs also define his pose and action, *e.g.* walking, running, standing and so on.

In computer vision the importance of incorporating part information into object models has been exhibited in various tasks. Detection is the best-known case, where constellations of HOG (histograms of gradients) templates accounting explicitly for object deformations, occlusions, and multiple aspects [Felzenszwalb 2008, Vedaldi 2009, Kumar 2009, Felzenszwalb 2010b, Azizpour 2012, Yang 2013, Chen 2014b], have been extensively used, outperforming simpler, monolithic models [Dalal 2005].

Parts can also be used for pose estimation. *Poselets* were introduced as parts that are highly discriminative of a person's pose [Bourdev 2009b]. For example, different poselets may correspond to frontal or profile faces, head and shoulder views, *etc.* Linear SVM classifiers with HOG features are trained to detect poselets, scanning the input image at multiple scales. The output of these detectors can be thought of as an intermediate layer of nodes, on top of which one can run a second layer of classification or regression.

In previous work that was done jointly with multiple collaborators we studied the detailed understanding of objects and their parts, using a rich set of semantic attributes [Vedaldi 2014]. In particular, we investigated how attributed parts can be modeled and recognized in images and how detailed supervision about them can be used to train better object models and diagnose them. Furthermore, we analysed the relation between parts and fine-grained attributes. An important conclusion of these analyses was that accurate part detection is highly beneficial in the prediction of certain attributes.

Given the importance of accurate part detection, we turn our focus to building powerful part detectors in Section 3.2.2. We employ HOG-based templates for parts, learned using the implementations of a popular detection system [Felzenszwalb 2010b, Girshick], and show that detection of object parts can be improved by using a significantly higher number of part templates. These results contradict previous works in the literature [Zhu 2012b], which argued that detection performance quickly saturates as the number of HOG templates increases. However, the computational complexity of such mixture models increases linearly in the number of templates used to model each part. To mitigate the added running time cost, we propose a coarse-to-fine algorithm to detect parts efficiently and accurately by organizing multiple templates in a tree hierarchy. Our method, described in Section 3.2.3, achieves a 4- to 5-fold speedup without sacrificing accuracy.

All the works we have mentioned so far rely on hand-crafted features, and specifically histograms-of-gradients. In Section 3.3 we substitute HOGs with CNN features, motivated by the ground-breaking results of deep learning and particularly CNNs, in image classification [Krizhevsky 2012] and object detection [Girshick 2014a]. We construct object and part templates using responses extracted from the last convolutional layer of the pretrained network described in [Krizhevsky 2012] and explore to what extent these successes carry over to the deformable part model paradigm, following in particular the framework laid out in [Felzenszwalb 2010b]. This straightforward change of representations yields a substantial improvement in detection performance with respect to a baseline using HOG features of 15% mean average precision (mAP). A complication that arises from the increased feature dimensionality is the increased computational cost ; during training we address this issue applying the LDA analysis previously used in [Girshick 2013], whereas for testing we make use of the Fast Fourier Transform (FFT), as in [Dubout 2012], to decrease running time to less than 2 seconds. In Section 3.4 we summarize and discuss our results and conclusions.

3.2 Efficient Detection of Object Parts

In this section we propose a coarse-to-fine approach for efficient part detection of object part templates built on HOG (histograms of gradients) features [Dalal 2005]. In Section 3.2.1 we perform a synopsis of previous works related to HOG-based modeling of object and their parts, while in Section 3.2.3 we describe our method in detail and provide a quantitative evaluation of performance on the OID dataset.

3.2.1 HOG-based Models for Objects and Parts

We review works in which object and their parts are modeled using HOG templates. We devote Section 3.2.1 to DPMs as they are one of the most heavily used detection frameworks in the last decade, and they have been the state-of-the-art in object detection until the re-emergence of convolutional neural networks. In the second part (Section 3.2.1) we list works that focus on accelerating DPMs during training and/or testing.

Object Parts and Deformable Part Models

One of the first attempts to model the relationship between an object and its parts was the *pictorial structure representation* by Fischler and Erlanger [Fischler 1973]. The authors model an object by a collection of its parts in a deformable configuration. This configuration is characterized by spring-like connections between certain pairs of parts, and each part encodes local visual properties of the object. This model can be used to find the best match in an image, by minimizing an energy function that takes into account both local matchings for the parts, and the deformation cost for each pair of connected parts.

Almost thirty years later, the pictorial structure representation inspired a series of works that eventually led to the *deformable part model* (DPM) [Burl 1998, Fergus 2001, Felzenszwalb 2005, Girshick 2011b, Felzenszwalb 2010a, Felzenszwalb 2010b]. In [Felzenszwalb 2005], Felzenszwalb and Huttenlocher address some basic shortcomings of the original pictorial structure representation. They provide an efficient algorithm for the energy minimization problem described in [Fischler 1973] for the case where the connections between parts form a tree structure. They also develop a method for learning object models from training examples and finding good object hypotheses in new images using the learned models. Marrying these ideas with established feature descriptors [Dalal 2005] and a novel, *latent SVM* formulation, resulted in a state-of-the-art object detection system [Felzenszwalb 2010b].

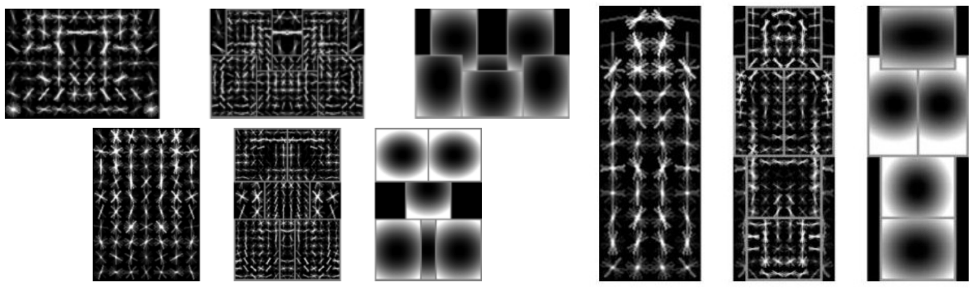


FIGURE 3.2 – **Left :** Root and part HOG templates trained for two different viewpoints for the bicycle class. Top row shows the side view and bottom row shows the frontal view). The coarse root template is on the left, the fine resolution parts in the middle and the deformation fields for the parts in the right column. **Right :** Root and part templates for the person class. Note that parts correspond to semantically meaningful human parts.

Deformable part models represent an object as a tree of filter templates : a "root" template that coarsely captures the appearance of the whole object and a fixed number of part high-resolution templates that capture finer details. The nominal positions of the part templates with respect to the root are learned from training data, but parts are allowed to move around their nominal positions to capture the inherent variability in object-part configurations. Datasets used for detection rarely include part annotations, save some exceptions, *e.g.* [Azizpour 2012]. This has two implications : first of all, there is no ground truth information concerning the exact position of the parts w.r.t. the object. This is catered for by using a greedy heuristic to initialize part coordinates and then optimize over their positions relative to the root during latent-SVM training. Second, it is clear that without appropriate part annotations, it is impossible to assign a semantic connotation to each template. Generally, the number of parts varies across object classes and even humans may choose different parts as components that form the same object. Deformable part models deal with this ambiguity by using a fixed number of part templates (usually six or eight) for all classes. Although there is not a strict correspondence between the index of a template and an object part, the learned filters often end up capturing appearances of meaningful structures, for instance the wheel of a car, or a the head of a person, as shown in Figure 3.2.

Formally, a model for an object with n parts can be defined by an $(n+2)$-tuple $M = (w_0, P_1, \ldots, P_n, b)$, where w_0 is a root filter, P_i is a model for the i-th part, and b is a bias term. P_i is a tuple containing the filter for the i-th part w_i, the two-dimensional vector specifying the nominal or "anchor" position for part i relative to the root position, and a four-dimensional vector defining a deformation cost for each possible placement of the part relative to the anchor. Part filters are square, whereas the root filters are rectangles of varying aspect ratio. For simplicity, we omit the bias and deformation terms and use the simplified expression $M = (w_0, w_1, \dots, w_n)$. For a more detailed analysis we refer the reader to [Felzenszwalb 2010b].

Considering an image at a single scale, an object hypothesis can be defined as $z = (z_0, z_1, \dots, z_n)$, where z_0 specifies the location of the root filter and z_i the location of the i-th part filter. Location is a two-dimensional vector holding the coordinates of the upper-left or center point of the root and part filters in the image plane. Let us also denote by $f(z)$ the representation of an image patch around image location z in some arbitrary feature space. The score of the object hypothesis with a given part configuration z under model M is obtained through

$$\begin{aligned} M(z) &= U_0(z_0) + \sum_{p=1}^{P} [U_p(z_p) + B_p(z_p, z_0)] && (3.1) \\ &= \langle\, w_0, f(z_0) \,\rangle + \sum_{p=1}^{P} [\langle\, w_p, f(z_p) \,\rangle + B_p(z_p, \bar{z}_p)]. && (3.2) \end{aligned}$$

The total score is a combination of *unary* terms U_p, capturing similarity between the learned templates and the feature representation of the local image patch, and *binary* terms B_p, penalizing large deviations of parts from their anchor positions $\bar{z}_p$. The representation used to model local image appearance is built on histograms of oriented gradients (HOG) [Dalal 2005], while the binary terms are quadratic functions of $z_p - \bar{z}_p$. Allowing for spatial uncertainty, we can find the optimal configuration of parts that maximizes 3.2 :

$$S^*(z) = \max_{z_0, \dots, z_n} M(z_0, \dots, z_n). \qquad (3.3)$$

So far we have only discussed hypothesis scoring at a single location and scale. Detecting objects at various locations for a single scale is straightforward : Equation 3.3 is computed *densely* in a sliding window fashion to quantify evidence of object presence at every possible image location z. The same model can be employed to detect objects at multiple scales, using features extracted from resized versions of the input image (a *feature pyramid*).

Finally, one of the factors that make detection challenging, is *intra*-class variability. Objects of the same class can appear different types of background clutter, poses, and viewpoints. Using a single model to capture this level of variation is not enough. In the DPM paradigm training data are divided into clusters of similar aspect ratio bounding boxes. The bounding box aspect ratio acts as a surrogate for distinguishing instances at different viewpoints and a different model (often referred to as *component*) is trained using only training instances in the corresponding viewpoint cluster, creating a *mixture* model. The number of components trained typically ranges from four to six, taking into account mixtures for horizontally flipped instances, and at test time the algorithm chooses the appropriate mixture component to score each instance.

Speeding-up Detection in Deformable Part Models

Deformable part models have been popular because of their good detection performance (recently surpassed by CNN-based systems), but training and testing of DPMs is costly due to the high number of templates, dense evaluation, and inability to share information between different object classes. To put things in a perspective, training a DPM on a single class from PASCAL dataset [Everingham 2010] takes several hours, while evaluating the trained detector on a single input image requires approximately 4 seconds. As a result, there has been significant effort to reduce both training and testing time, usually either by sharing computation among templates or through a coarse-to-fine approach, for instance a computational cascade.

Ott *et al.* propose an extension to the original DPM, which allows for *sharing* of object part model among multiple mixture components, as well as among different object classes [Ott 2011]. This approach reduces the total number of parameters to be learned, leading to more compact models that can generalize better given a finite size of training data, and scale to a large number of classes at a tractable computational expense. [Kokkinos 2013] follows a similar path, identifying and exploiting structures that are shared across different object categories. He uses a Shift-Invariant flavor of Sparse Coding to learn mid-level elements dubbed *sufflets* that can translate during coding ; the same shufflet can be used at many location to reconstruct *both* part *and* root filters. Yang and Ramanan suggest using a novel representation of deformable part models, that uses a mixture of small, non-oriented parts to describe objects at different poses and views [Yang 2013]. A spring model represents connections between parts and allows them to translate in the image plane to approximate small warps ; this approach is both flexible and efficient, since computation is shared across similar warps. The authors define parts to be located at joints, since most human pose datasets include images with labeled joint positions, making this a supervised problem that is solved using a custom implementation of dual-coordinate descent.

The idea of using cascades to reduce the computational burden of evaluating a classifier at every point of a large search space has been deployed in the past, mostly for face detection [Viola 2001, Fleuret 2001, Bourdev 2005, Šochman 2005, Gangaputra 2006]. Felzenszwalb *et al.* build a cascade algorithm tailored to star-shaped models, such as DPMs [Felzenszwalb 2010a]. They use a sequence of thresholds to prune partial hypotheses, coupled with dynamic programming and generalized distance transforms, to achieve much faster detection than in the original DPM. The thresholds are selected so that they lead to a small detection error with high probability.

Sun *et al.* employ a part-based model for joint object detection and pose estimation of articulated objects [Sun 2011]. The proposed Articulated Part-based Model (APM) is a hierarchical, coarse-to-fine model that recursively represents an object as a collection of parts at multiple levels of detail, through a parent-child relationship. A top-down search strategy is used to find strong object hypotheses, starting from a coarse level (full object) and moving to increasing levels of granularity (parts), while a bottom-up strategy is used to guide the top-down search to the best pose configuration hypothesis. An important characteristic of APMs is that they share parts, allowing them to capture an exponentially large number of pose configurations, using only a few models. Accelerating DPMs via a coarse-to-fine procedure was also addressed in [Kokkinos 2011a, Kokkinos 2012]. The author adapts the dual trees data structure [Gray 2003] for DPM-based detection to compute probabilistic bounds for the true part scores. The branch-and-bound algorithm exploits these bounds to focus only on promising image locations, yielding the same detection results but 10-20 times faster on average.

Other techniques for speeding up detection with DPMs include sparse feature encodings of high-dimensional features [Vedaldi 2012], hashing [Dean 2013], or a combination of several speedup mechanisms, such as HOG-filter interpolation, vector quantization and an efficient GPU implementation of a DPM cascade [Sadeghi 2014]. Girshick and Malik incorporate a latent version of linear discriminant into the DPM training pipeline [Girshick 2013].

Latent LDA uses efficient closed-form updates and does not require an expensive search for hard-negative examples, reducing training time considerably, maintaining high average precision.

3.2.2 Improving Part Detection

As mentioned in the introduction, part detection plays a major role in fine-grained attribute prediction. In this section we seek ways of improving the detection of object parts, revisiting the question "Do we need more data or better models" for object detection posed by [Zhu 2012b]. In their work they showed that on PASCAL VOC datasets a leading approach for detection, mixtures of deformable part-based models tends to saturate on performance with three (3) mixtures on most categories. Even with the addition of 10× more data the authors noted that the performance of the models does not improve significantly — more mixtures simply added robustness by "taking away" noisy training examples.

In our experiments we use the OID dataset to train HOG templates for different parts of the aeroplane. For the final experiments and results that are reported in this section we used the *trainval* set for training and the *test* subset of the OID for evaluation. We train our models using the latest implementation of deformable part models [Girshick]. Since we associate a single template with each part, we deactivate the second part of the training that adds deformable parts in the original "root" template.

Do more mixtures help ? We first perform an experiment where we train mixtures of HOG templates for parts, varying the number of mixture components $k =$ 6, 20, and 40. We found that, for most parts, detection performance saturates at around 40 mixture components, which is an order of magnitude higher than the same number on the PASCAL dataset. We use aspect-ratio based clustering to initialize the mixtures. Figure 3.4 shows that the performance of the learned models for detecting noses are respectively 57%, 60%, and 62%, improving consistently as the number of mixtures increases. Table 3.1 shows similar gains for other parts. This can be because unlike those in the PASCAL VOC dataset, objects in Oid are of significantly higher resolution and variety, and hence can benefit from the details that more mixture components can provide.

Part	$k = 6$	$k = 20$	$k = 40$	Shape
Nose	57	60	62	68
Vertical stabilizer	42	54	52	60
Wings (grouped)	15	19	22	28

Table 3.1 – Part detection performance (MAP%) as a function of the number of components. Shapes results were obtained with $k = 40$ components, and part segmentations to initialize clusters.

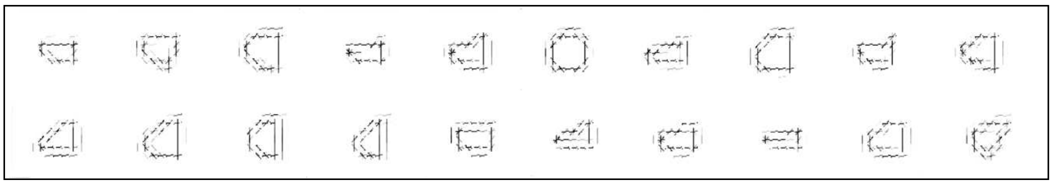


Figure 3.3 – Nose shape clusters.

Do semantic attributes help ? In addition to aspect ratio based clustering we can use a supervised left-right clustering which improves performance from 62% to 67%. Additionally, we use the segmentation of the noses to cluster the shapes using the HOG features of the foreground object. This initialization improves the performance to 68%. A similar trend is observed for other parts and the overall object as seen in the last column of Table 3.1. Thus, better initialization using semantic attributes can improve detection accuracies a trend observed by several others [Zhu 2012b, Matzen 2013, Bourdev 2009a].

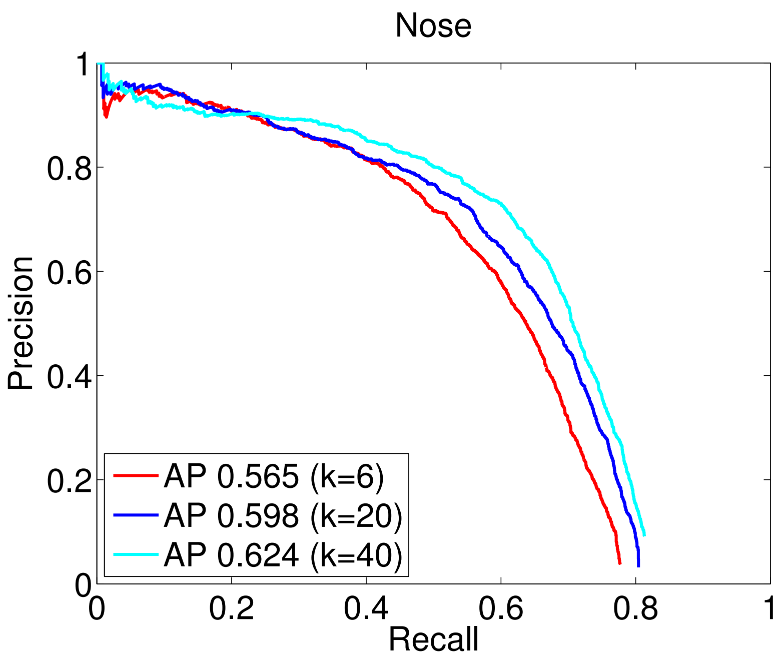


Figure 3.4 – **Nose detection results (unsupervised).** Detection AP using $k = 6$, 20, and 40 mixture components based on aspect-ratio clustering to initialize the latent SVM.

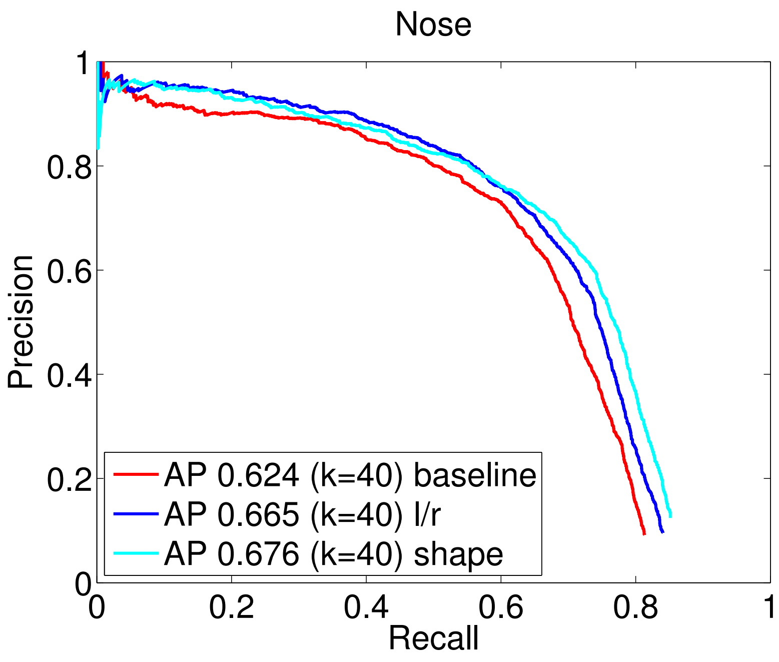


Figure 3.5 – **Nose detection results (supervised).** Average precision for the baseline clustering, left-right clustering from [Girshick], and our supervised shape clustering for $k = 40$.

3.2.3 Hierarchical Part Detection Cascades

Having outlined the merit of using multiple shape clusters for part detection, we now address computational efficiency. The refined processing advocated in Section 3.2.2 incurs an increase in computational cost, as it requires the evaluation of multiple classifiers at every candidate part location.

One of our technical contributions is a method to efficiently process an image with a fine-grained model through a hierarchical part detection cascade. We build on the broader theme of sequential testing [Amit 1999, Amit 2004] and organizing multiple classifiers in a tree-structured hierarchy [Andreetto 2012, Deng 2011], and integrate it with bounding-based detection [Kokkinos 2013].

We develop a coarse-to-fine algorithm that originally gets rough and quick score estimates for sets of similar components, and then recursively refines such scores by working with increasingly smaller sets of components. For this, we start in Section 3.2.3.1 by recovering an hierarchy of parts from a set of discriminatively trained components. Then, in Section 3.2.3.2 we use this hierarchy at test time to quickly recover the filters that score above a predetermined threshold. This is accomplished by recursively constructing a probabilistic upper bound of the part scores lying below a tree node, and pruning accordingly.

3.2.3.1 Part hierarchy computation

We establish a tree-hierarchy to represent the k component filters learned in Section 3.2.2 ; we use agglomerative clustering and force the learned tree to be binary, ensuring that it has depth at most $\lceil \log k \rceil$. As shown in Figure 3.6, the leaf nodes of the hierarchy learned for the "nose" part correspond to the individual components-filters while higher-level nodes represent the ensemble of filters below them.

Starting with the leaf nodes, we measure the alignment-based similarity of components i and j :

$$D[i,j]=\min_{h',v'} \sum_{h,v} \big(f_i(h,v,d)-f_j(h+h',v+v',d)\big)^2 \tag{3.4}$$

where h, v, d are the horizontal, vertical, and direction indexes of a HOG template respectively, h', v' indicates an amount of translation applied to f_j, while we treat different sizes of f_i, f_j by zero-padding.

This provides us with a $k \times k$ dissimilarity matrix D between parts ; we greedily pick the most similar pair, remove the respective rows and columns from D, and repeat until all leaves have been paired. Each pair i, j is represented by its parent, l, in terms of the aligned mean :

$$f_l(v,h,d) = \frac{1}{2}(f_i(v,h,d) + f_j(v+v^*, h+h^*, d)), \tag{3.5}$$

where (v^*, h^*) is the minimizer of Eq. 3.4. We repeat this procedure at the next level, halving the number of nodes present at every hierarchy level ; for $k = 2^i$, the hierarchy will thus contain $2k - 1$ nodes and $i + 1$ levels.

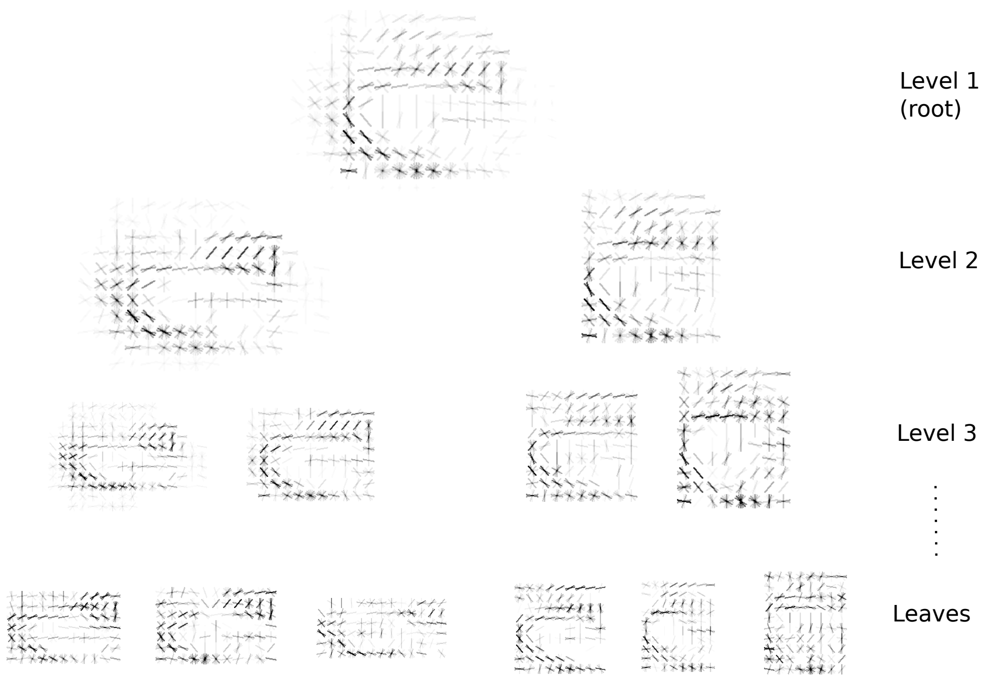


FIGURE 3.6 – **Hierarchical filter tree.** The discriminative power of the multi-component model is summarized in a single super-node (root). Its left child corresponds to low aspect ratio leaf nodes, whereas the right child captures characteristics of high aspect ratio leaf nodes. This hierarchy can result in bounding schemes for the efficient pruning of large candidate locations of the image.

3.2.3.2 Hierarchical pruning with probabilistic bounds

We use the constructed hierarchy to accelerate detection ; at any pixel we start at the root, visit the left and right children, check if any of them holds promise for delivering a score above threshold and then accordingly recursively refine or stop. If the score estimate at all nodes upper bounds the leaf scores below it, this procedure is guaranteed to deliver all parts scoring above threshold. The main technical hurdle is the bound construction. For this we adapt the *probabilistic bounds* [Mitzenmacher 2005] used recently in [Kokkinos 2013] to our coarse-to-fine filter evaluation scheme. While no longer being deterministic, in practice these bounds incur only negligible changes in performance.

In particular, consider having M filters, $f_1, \ldots, f_M$ lying below a node l in the part hierarchy. Given an input HOG feature I, we consider how the average filter $\hat{f} = \frac{1}{M}\sum_{m=1}^{M} f_m$ can be used to bound the individual filter scores, $s_m = \langle f_m, I\rangle$. As in [Kokkinos 2013], rather than taking the $\hat{s} = \langle \hat{f}, I\rangle$ estimate at face value, we model $\varepsilon_m = s_m - \hat{s}$ as a random variable, construct an interval $[-\alpha, \alpha]$ that contains ε_m with high probability, and then bound s_m from above by $\overline{s} = \hat{s} + \alpha$. The value of α is determined by Chebyshev's inequality :

$$P(|X| > \alpha) \leq \frac{V}{\alpha^2}, \tag{3.6}$$

which relates the second moment $V = E\{X^2\}$ of a zero-mean random variable X with the probability that its absolute exceeds α. Namely, X lies outside $[-\alpha, \alpha]$ with probability smaller than V/α^2, or, equivalently, X lies in $[-\sqrt{V/p_e}, \sqrt{V/p_e}]$ with probability larger than $1 - p_e$.

Unlike [Kokkinos 2013], rather than s_m we now need to bound $\max_m s_m$. This requires two modifications : first, we deal with the 'max' operation as follows :

$$P(\max_m s_m > \bar{s}) = P(\vee_m \{s_m > \bar{s}\}) \leq \sum_m P(s_m > \bar{s}) < M p_e, \tag{3.7}$$

where $\vee_m$ indicates a logical-or of the M events, the first inequality follows from the union-bound, and the second inequality holds for a p_e such that $P(s_m > \bar{s}) < p_e$, $\forall l$.

This brings us to our second modification : constructing $\bar{s}$ so that $P(s_m > \bar{s}) < p_e, \forall m$ involves bounding the different variables $s_1, \ldots, s_M$ with a common expression $\bar{s}$. For this we rewrite s_m as summations over HOG cells :

$$s_m = \langle f_m, I \rangle = \sum_c \sum_d f_m(c,d) I(c,d), \tag{3.8}$$

where $c = (v, h)$ indexes vertical/horizontal positions and d indexes the HOG cell dimensions. We can thus write :

$$\varepsilon_m = \sum_c \varepsilon_{c,m}, \text{ with } \varepsilon_{c,m} = \sum_d \left[\hat{f}(c,d) - f_m(c,d)\right] I(c,d) \tag{3.9}$$

At any cell c we assume the approximation errors $\hat{f}(c,d) - f_m(c,d)$ can be modelled as zero-mean, independent, identically distributed (iid) variables, and estimate their second moment using the reconstruction error of the respective filter cell :

$$V_{c,m} = \frac{1}{D} \sum_{d=1}^{D} \left(\hat{f}(c,d) - f_m(c,d)\right)^2, \tag{3.10}$$

where $D = 32$ is the HOG cell dimensionality. Treating $\varepsilon_{c,m}$ as the weighted-by-$I(c,d)$ sum of D iid variables its second moment of ε_c will be :

$$E\{\varepsilon_{c,m}^2\} = V_{c,m} \|I_c\|_2^2, \tag{3.11}$$

where $\|I_c\|_2^2$ is the ℓ_2 norm of the 32-D vector formed from the c-th HOG cell. We further consider the individual error contributions of the different cells as independent and express the second moment of ε_l as follows :

$$E\{\varepsilon_m^2\} = \sum_c E\{\varepsilon_{c,m}^2\} = \sum_c V_{c,m} \|I_c\|_2^2 \tag{3.12}$$

The last expression provides us with the error variance needed in Eq. 3.6 to construct the upper and lower bounds to the score. This however is dependent on the filter index m. In order to drop the dependence on m, we also upper bound the variance in Eq. 3.12 by maximizing $V_{c,m}$ over m :

$$E\{\varepsilon_m^2\} \leq \sum_c \left(\max_m V_{c,m}\right) \|I_c\|_2^2 \doteq \overline{E} \tag{3.13}$$

Having derived an expression for the variance that no longer depends on m, we can now bound $s_m, m = 1, \ldots, M$ with a single expression, as required. Namely, with probability of error at most p_e, we can write :

$$\max_{m \in \{1,\ldots,M\}} s_m \leq \overline{s} \doteq \hat{s} + \sqrt{\frac{\overline{E}M}{p_e}}, \tag{3.14}$$

with $\hat{s}$ being the mean filter's score, M the number of filters (by the union bound, Equation 3.7), and $\overline{E}$, in Eq. 3.13 is an upper bound of the variance of the individual approximation errors, $\varepsilon_m = s_m - \hat{s}$. We note that other than independence of the errors we did not make further assumptions, which makes our method fairly generic.

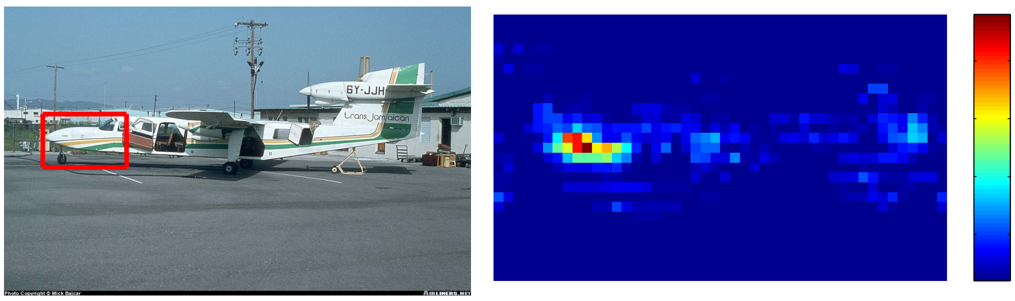


FIGURE 3.7 – **Pruning of candidate locations.** Detection example and the number of visits by some node in the filter tree. By pruning candidate locations as we move from the root to the leaves, the exact score is evaluated at only a fraction of the image domain(right).

3.2.3.3 Experimental Settings

We build two tree cascades, one for the left-facing and another for the right-facing filters, each comprising 20 mixture components. We set $p_e = 0.01$ for the individual parts, which translates to $p_e = 0.2$ for the root filters of the respective hierarchies. We use the upper bounds from Section 3.2.3.2 for the construction of probably-accurate empirical thresholds, based on training data [Felzenszwalb 2010a]. In particular, we use the 1-th percentile of high-scoring positive samples going through each tree node as a pruning threshold.

To facilitate comparison, we use single-threaded implementations for the baseline and our approach. Both methods can easily be parallelized ; for example we can

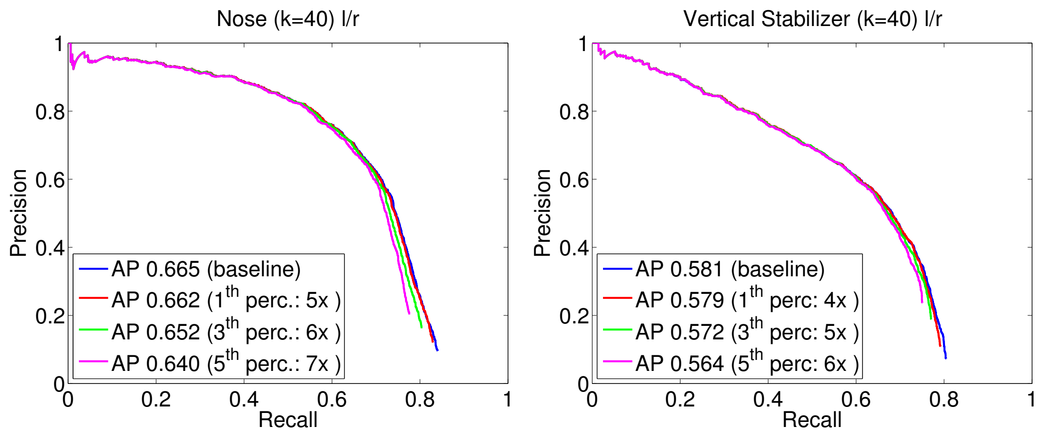


FIGURE 3.8 – **Cascade detection results.** Precision-recall curves for different percentile values used to determine empirical thresholds, and corresponding speedup with respect to the baseline.

compute the convolutions over multiple scales at the same time. All timings were conducted on a 4-core Intel desktop CPU. Our implementation for a hierarchy of 40 filters provides us with at least a 4× speedup, while attaining the same detection performance as the full-blown convolution result. In Figure 3.8 we show we can achieve even larger speedups, by increasing the percentile of positive samples, with only a small drop in AP by 0.01-0.02. Our code is publicly available at `https://github.com/tsogkas/oid_1.0`.

3.2.3.4 Empirical Validation of Probabilistic Bounds

In this section we provide empirical evidence for the validity of the probabilistic bounds described in Section 3.2.3.2. In particular, we made the assumption that the approximation errors between the individual filters and the "mean filter" behave as independent, identically distributed random variables, per HOG cell. In general, this might not be true, cancelling the validity of the resulting probabilistic bounds. Here we demonstrate empirically that this is not the case for our problem.

For this, we compare (i) the empirical probability that the probabilistic upper bound is violated and (ii) the probability predicted by Chebyshev's inequality, combined with the union bound (Eq. 3.6, 3.7). We perform this comparison at multiple levels of the part hierarchy, comparing always the actual maximum ($\max_k s_k$, in Equation 3.7) to the probabilistic upper bound to it ($\bar{s}$, in Equation 3.14). Results are obtained by averaging over multiple images, and scales.

As illustrated in Figure 3.9, we are on the safe side : the probability of actually erring is lower than that predicted by Chebyshev's inequality.

In Figure 3.10 we visually compare the exact values of the quantities being bounded ($\max_k s_k$- left) to the bounds computed at different levels of the part hierarchy ($\bar{s}$ - right). On the top row we show results at the topmost level of the part hierarchy, on the bottom row we show the results at the penultimate level of

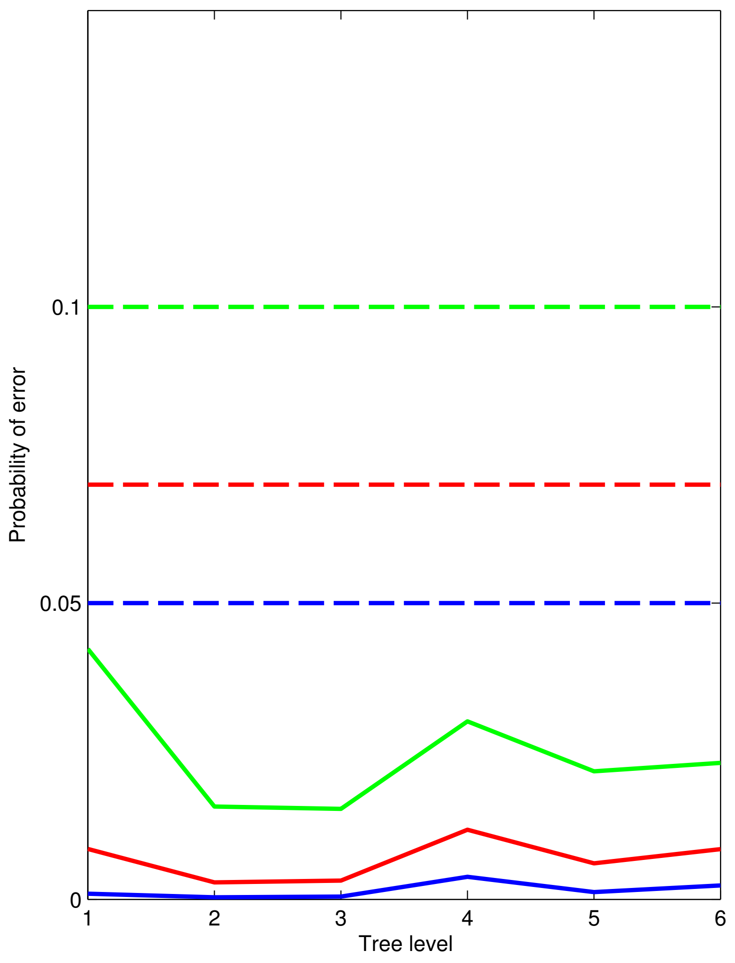


FIGURE 3.9 – Empirical (solid) estimate of the probability of error p_e versus theoretically predicted (dashed) p_e, for three different (color-coded) values of p_e, at different levels of the part hierarchy ('tree level'). We verify that the empirical probability of error that is lower than that predicted by Chebyshev's inequality (Eq. 3.6, 3.7), and we therefore are "on the safe side".

the hierarchy. We note that (i) we never actually compute the quantities on the left – these are used only for illustration (ii) we use instead the quantities on the right as quickly computable proxies to the exact maxima. We observe that for lower levels of the hierarchy the bounds become tighter, and as such can lead to more aggressive pruning, while higher up the bounds are loose, effectively avoiding "early commitments".

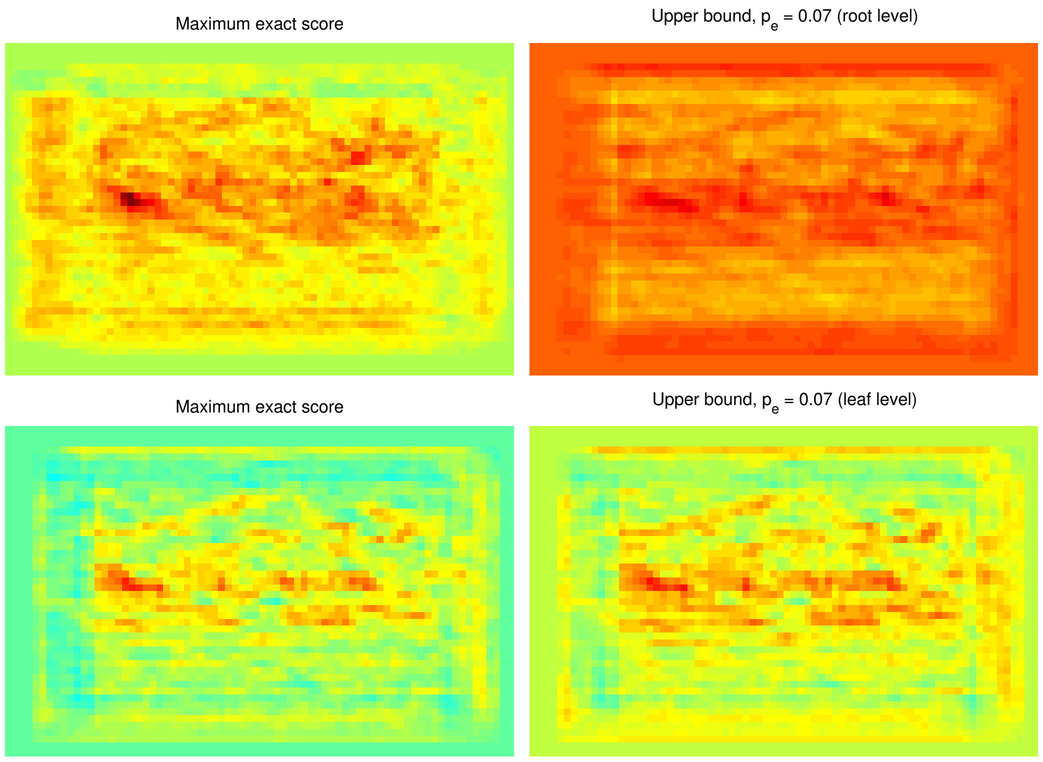


FIGURE 3.10 – Comparison of $\max_k s_k$ vs $\overline{s}$ at the root level (top), where k spans the whole set of leaf indices, and at the penultimate hierarchy level (bottom), where the upper bound bounds a single pair of leaf nodes.

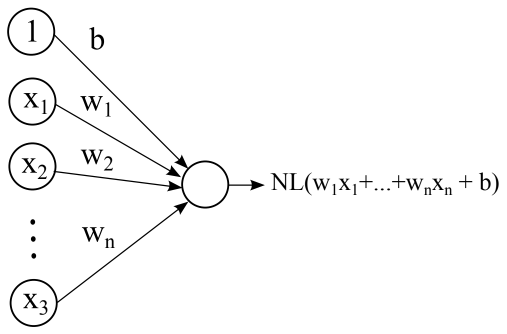
3.3 Improving Object Detection with CNN Features

In Section 3.2 we modeled object parts using a set of HOG templates. These filters are built on class-independent, local features (histograms of gradients) and measure similarity of a well-localized part of the image to the pattern under question. In this section we explore the effect of using convolutional features for modeling objects and their parts in a DPM pipeline for object detection, driven by recent successes of convolutional neural networks (CNNs) in several tasks, and especially image classification [Krizhevsky 2012]. Convolutional features capture information within a much larger spatial extent as opposed to the limited support of HOG templates. Due to the hierarchical arrangement of convolutional layers they learn to represent increasingly complex structures, as demonstrated in [Zeiler 2014] ; therefore, they intrinsically encode spatial and semantic relationships between objects and their parts. We start with a brief review of convolutional neural networks and their recent successes in classification and detection in Section 3.3.1. We proceed by describing how we integrate CNN features into the DPM pipeline, along with technical contributions to reduce running time during training and testing. In Section 3.3.3 we show that our approach significantly improves detection performance on the VOC2007 dataset (14.5% absolute mAP increase), running at approximately 2 seconds for a single image, despite the increased feature dimensionality.

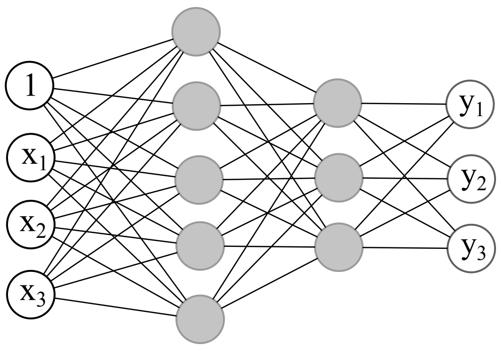
3.3.1 Convolutional Neural Networks for Classification and Detection

Convolutional neural networks are a variant of multilayer perceptrons in which each neuron in layer l is connected to a subset of nodes in layer $l-1$, the neuron's *receptive field*. This type of local connectivity aims at exploiting the strong spatial correlation present in natural images, since each unit is responsive only to variations within its receptive field. It also ensures that artificial neurons learn to produce the strongest response to a spatially local input pattern. The connection parameters (weights and bias) are usually shared for a given layer, which means that the same linear filter is applied on –possibly overlapping– regions of the entire visual field, forming a *feature map*. The operation we described is equivalent to a convolution with a set of n_l convolution kernels of size $k_l \times k_l$, l being the layer index. The kernel size, k_l, is the size of the receptive field with respect to the input coming from the previous layer. Note that stacking many layers one after another leads to convolution filters that become increasingly "global", *i.e.* responsive to a larger region with respect to the *pixel space*. Convolutions are often followed by a *max-pooling* operation.

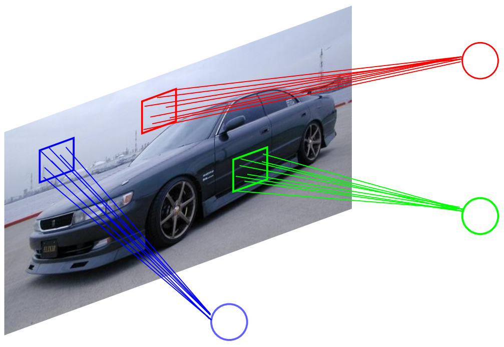
CNNs were popularized by Lecun *et al.* for the task of digit recognition [LeCun 1998]. They later fell out of favor, however, and it was not until recently that they reemerged as the method of choice for image classification, yielding unprecedented accuracy compared to alternative approaches [Krizhevsky 2012]. The key difference between the architecture proposed by Krizhevsky (commonly referred to as *Alex-*



(a) Perceptron



(b) Multi-layer perceptron



(c) Neurons in the first hidden layer of a CNN

FIGURE 3.11 – **3.11a :** A single perceptron combines its inputs linearly (adding some bias b) and passes the result to a non-linear function. Arranging multiple neurons in an hierarchical layer structure, as in **3.11b**, results in a *multi-layer* perceptron. The intermediate layers between input the input vector $\mathbf{x}$ and the output vector $\mathbf{y}$ are marked in gray color. These "hidden" units learn to represent important features of the task domain during training. We omit the non-linearities for the sake of clarity. **3.11c :** Connections in CNNs are sparse (in contrast to MLPs) with each neuron being connected to a small image patch (each image pixel is viewed as an input node).

net) and previous architectures is the increased number of layers (depth). Deeper networks profit from the volume of data available in ImageNet, encoding representations of increased complexity without overfitting. Alexnet consists of five pairs of convolutional and max-pooling layers, followed by two fully-connected layers (we intentionally ignore intermediate normalization and dropout layers in our description), as shown in Figure 3.12.

Results in [Krizhevsky 2012] showed that, given enough training data, wider (more filters per layer) and deeper (more layers) networks can learn more complex feature representations and yield improved performance on difficult tasks. This conclusion was further solidified in later works that use even deeper CNNs [Szegedy 2014,

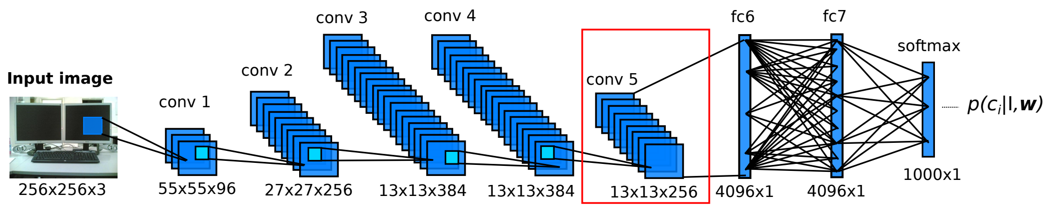


FIGURE 3.12 – A schematic representation of Alex Krizhevsky's network used for classification. Between two consecutive convolutional layers there are a max-pooling and a local response normalization layer, that are not depicted to avoid cluttering the figure. The output of a convolutional or fully-connected layer can be used as an intermediate feature representation. In Section 3.3.2 we use features from the last convolutional layer (conv5), because they retain spatial layout information.

Simonyan 2014, Papandreou 2014]. Simonyan *et al*. [Simonyan 2014] augment the original Alexnet, pushing the number of total layers from 8 to 16 or 19 (depending on the variant, VGG16 or VGG19). They decrease the receptive field of the first layers using smaller 3×3 convolution kernels, but what makes the decision function more discriminative is the higher number of non-linear rectification units. Szegedy *et al*. [Szegedy 2014] avoid a sudden blow-up of computational demands in their 22-layer GoogLeNet by applying dimensionality reductions and projections wherever the computational requirements would increase too much otherwise.

A first approach to object localization and detection with this class of models is reported in the *OverFeat* system [Sermanet 2014]. They employ a deep network trained for image classification but apply the last two fully-connected layers in a convolutional fashion to produce spatial activation responses for test images larger than the input images used for training. They jointly train two CNNs : one predicts the class of the object and another the coordinates of the bounding box containing it. These networks are then fed with all possible windows from the original images, at different scales. A squared loss is used as the loss for bounding box regression.

Girshick *et al*. moved away form the typical detection paradigm in [Girshick 2014a]. Instead of searching objects densely in the input image, they use Selective Search [Uijlings 2013] to extract region proposals that are subsequently treated as independent input images. These salient regions are efficiently generated and warped to a fixed size window, which is then used as input to a CNN, thus reducing object detection to image classification. Combining this idea with a network finetuning stage during training, and a bounding box regression step for better localization yields a state-of-the-art mean average precision (mAP) of 58.5% on VOC2007, and of 31.4% on ILSVRC2013. Substantial acceleration and further improvements in performance ha been achieved in [He 2014] by combining R-CNNs with spatial pyramid pooling and extensions of the original R-CNN paper [Girshick 2015, Ren 2015].

3.3.2 Integrating Convolutional Features Into DPMs

Having provided a brief summary of previous works on CNNs, we now turn to our contribution, presented in [Savalle 2014], and conducted in parallel with [Girshick 2014b], where we explored the potential of integrating convolutional features in deformable part models. Our motivation was the observation that the region proposal strategy of R-CNN only partially captures the complexity of visual objects. For tasks such as pose estimation [Yang 2013] or facial landmark localization [Zhu 2012a], one may still need DPMs to optimize over the relationships between object parts. Using a sliding window (as in DPMs) can also potentially achieve a better recall than generic region proposal mechanisms, which may miss certain objects altogether.

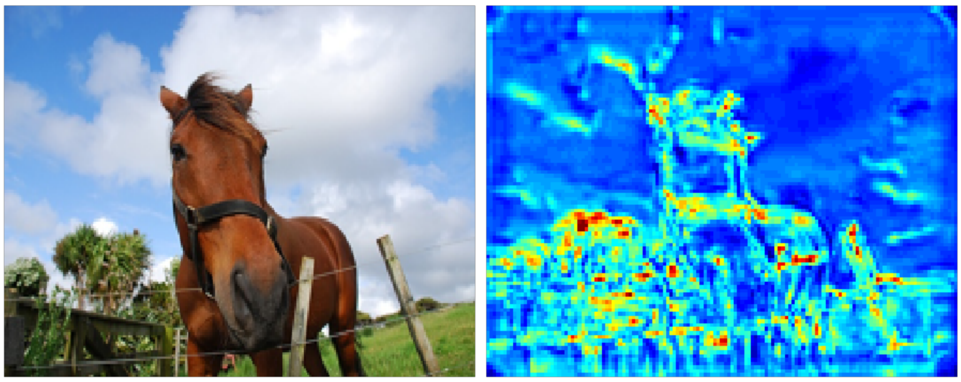
A first work in this direction was presented in DenseNet [Iandola 2014], which proposed to compute a feature pyramid based on the topmost convolutional layers of Krizhevsky's network. In order to efficiently obtain a multi-scale representation the patchwork of scales approach [Dubout 2012] is used. Although [Iandola 2014] demonstrates how to efficiently compute feature pyramids based on a convolutional network, no quantitative evaluation on a detection task is provided.

In this work we push this line of work a step further, integrating CNN features into DPM training and testing. In particular, the standard input of Krizhevsky's network consists of a fixed-size, $224 \times 224 \times 3$ patch, which is transformed to a $13 \times 13 \times 256$ patch at the topmost (fifth) convolutional layer. An important observation is that the need for fixed-size inputs is imposed by the fully connected layers only - convolutional layers have no such restriction.

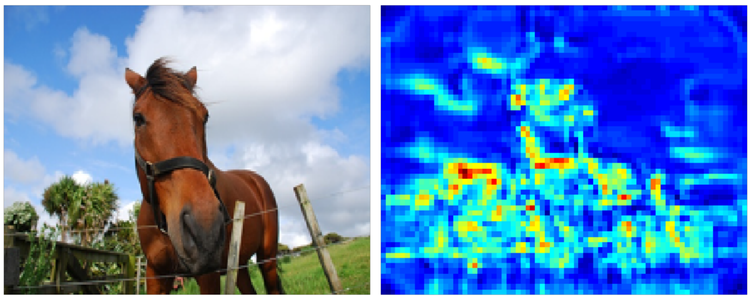
Rather than working with fixed-size patches, we provide as input to a convolutional network an arbitrarily-sized image ; following [Iandola 2014] we do this for multiple rescaled versions of the original image, obtaining a multi-scale CNN feature pyramid that substitutes the HOG feature pyramid typically used in DPMs. The convolutional network we use has the same architecture as the first five (convolutional) layers of Krizhevsky's but uses fine-tuned parameters from [Girshick 2014a]. A major technical challenge is that of making the integration of CNN features with DPMs computationally efficient. Compared to using HOG features, using CNN features corresponds to an eight fold increase in the dimension (from 32 to 256), while the DPM framework is already quite computationally expensive.

To achieve efficiency during training we exploit the LDA-based acceleration to DPM training of [Girshick 2013], using a whitened feature space constructed for CNN features ; this reduces the computation time typically by a factor of four. To achieve efficiency during convolutions with the part templates (used both during training and testing), we perform convolutions using the Fast Fourier Transform, along the lines of [Dubout 2012]. This reduces the convolution cost from typically 12 seconds per object (using an optimized SSE implementation) to less than 2 seconds.

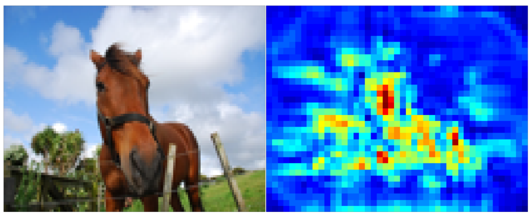
A parameter that turned out to be central to improving detection performance was the subsampling factor, denoted by `sub`, between the original input and the layer-5 feature representation. For Krizhevsky's network, `sub` $= 16$, meaning that a a block of 16×16 pixels in the input image is represented by a single layer-5 feature



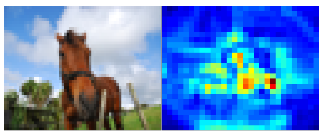
(a)



(b)



(c)



(d)

FIGURE 3.13 – A CNN feature pyramid : subsampled versions of the original image and the corresponding L^2 norm of the 5-th layer activations.

vector. As this corresponds to substantially larger bins than the ones typically used in HOG, we instead oversample our image by a factor of two before computing features, which effectively leads to `sub` $= 8$. We only report results with `sub` $= 8$, as `sub` $= 16$ leads to significantly worse APs, while `sub` $= 4$ turned out to be computationally prohibitive.

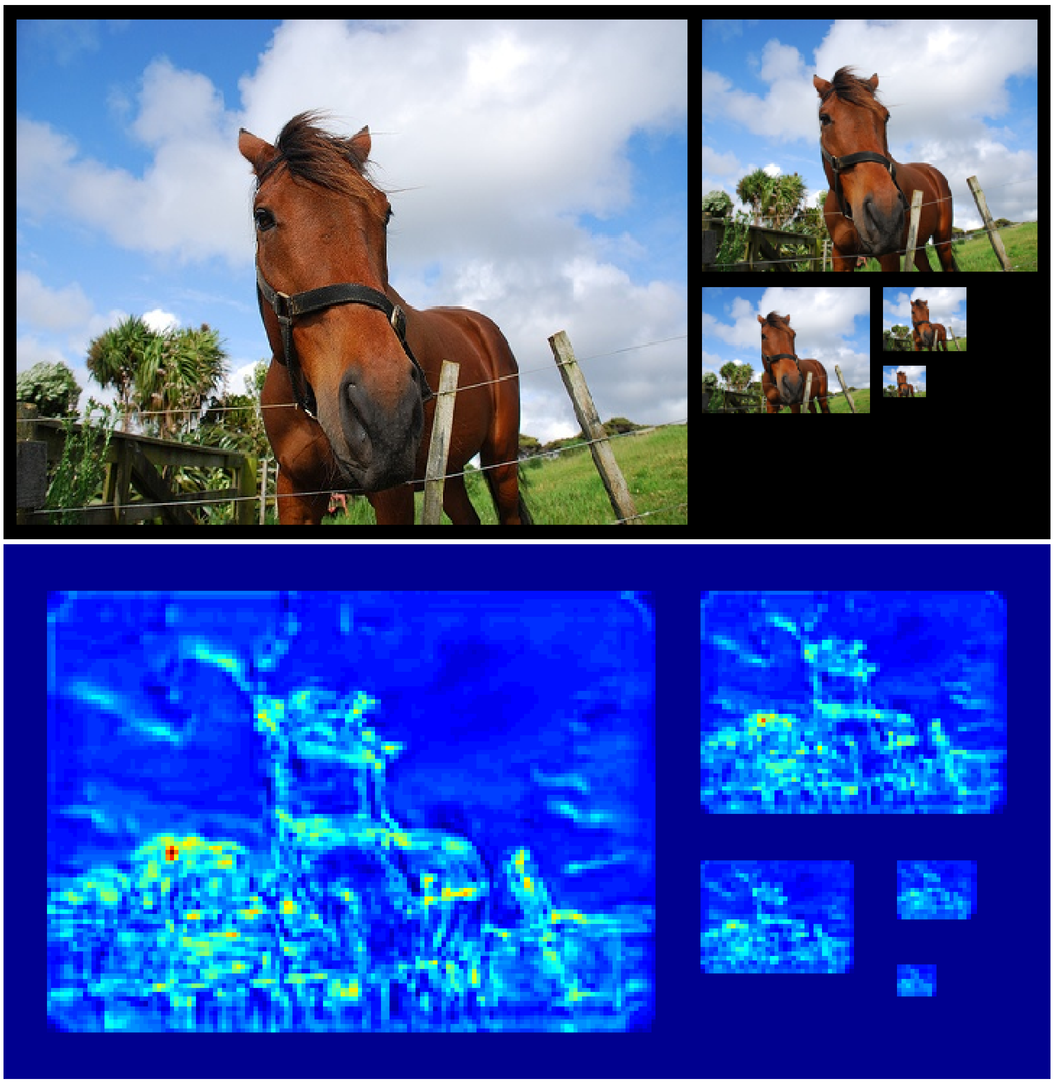


FIGURE 3.14 – **Top :** We tile rescaled versions of the original image into a single, bigger canvas that can be processed efficiently by the convolutional layers. **Bottom :** The L^2 norms of the obtained CNN activations in the 5-th layer. After computing convolutions on the patchwork image, we decompose the result into responses corresponding to individual scales and synthesize a pyramid of CNN responses.

3.3.3 Experiments

Our results are reported in Table 3.2 and Table 3.3. We consider two variants of our method : the first one, C-DPM, combines sliding window detection followed by non-maximum suppression ; the second one, C-DPM-BB, is augmented with bounding box regression, using the original bounding box coordinates as input features.

We compare these two variants to the following methods : DPMv5 refers to the baseline DPM implementation using HOG features and bounding-box regression, as in [Felzenszwalb 2010b], while RCNN5, RCNN7, RCNN7-BB correspond to the

Method	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	dtbl
C-DPM	39.7	59.5	35.8	24.8	35.5	53.7	48.6	46.0	29.2	36.8	45.5
C-DPM-BB	50.9	64.4	43.4	29.8	40.3	56.9	58.6	46.3	33.3	40.5	47.3
DPMv5	33.2	60.3	10.2	16.1	27.3	54.3	58.2	23.0	20.0	24.1	26.7
C-DPM-BB vs. DPMv5	+17.7	+4.1	+33.2	+13.7	+13.0	+2.6	+0.4	+23.3	+13.3	+16.4	+20.6
RCNN7-BB	68.1	72.8	56.8	43.0	36.8	66.3	74.2	67.6	34.4	63.5	54.5
RCNN7	64.2	69.7	50.0	41.9	32.0	62.6	71.0	60.7	32.7	58.5	46.5
RCNN5	58.2	63.3	37.9	27.6	26.1	54.1	66.9	51.4	26.7	55.5	43.4
C-DPM vs. RCNN5	-18.5	-3.8	-2.1	-2.8	+9.4	-0.4	-18.3	-5.4	+2.5	-18.7	+2.1

Table 3.2 – Results on PASCAL VOC 2007 : average precision in percent

performance of (fine-tuned) RCNN using layer 5 features, layer 7 features, or layer 7 features with an extra bounding box regression based on (richer) CNN features, respectively.

The last rows of the second and third blocks indicate the difference between the AP achieved by our method and DPMv5 or RCNN5, respectively. To have comensurate performance measures we compare DPMv5 with our variant that includes bounding box regression, (C-DPM-BB), and RCNN5, which does not include bounding box regression, to C-DPM.

From the second block it becomes clear that we significantly improve over HOG-based DPMs, while employing the exact same training pipeline ; this is indicating the clear boost we obtain simply by changing the low-level image features.

However the results are not as clear-cut when it comes to comparing to RCNN. Even when comparing only to RCNN-5, we have a moderate drop in performance, while our DPMs are still quite behind RCNN-7. The difference with respect to RCNN-7 can be attributed to the better discriminative power of deeper features and could be addressed by incorporating non-linear classifiers, or computing all features up to layer 7 in a convolutional manner.

But what we find most intriguing is the difference in performance between RCNN-5 and C-DPM, since both use the same features. One would expect DPMs to have better performance (since they do not rely on region proposals, and also come with many mixtures and deformable parts), but this is not the case. We suspect that this is because (i) DPMs split the training set into roughly 3 subsets (for the different aspect ratios/mixtures), effectively reducing by 3 the amount of training data and (ii) DPMs are somewhat rigid when it comes to the kind of aspect ratio that they can deal with, (3 fixed ratios) which may be problematic in the presence of large aspect ratio variations ; by contrast RCNN warps all region proposals images onto a single canonical scale.

3.4 Summary

In this chapter we have introduced a coarse-to-fine technique for efficient detection of object parts in images, using deformable part models. Our method organises part templates in a binary tree and evaluates them in an hierarchical fashion, speeding up detection by four to seven times. The efficiency of our algorithm stems from the computation of probabilistic bounds that are employed as estimations of

Method	dog	hors	mbike	person	plant	sheep	sofa	train	tv	mAP
C-DPM	42.0	57.7	56.0	37.4	30.1	31.1	50.4	56.1	51.6	43.4
C-DPM-BB	43.4	65.2	60.5	42.2	31.4	35.2	54.5	61.6	58.6	48.2
DPMv5	12.7	58.1	48.2	43.2	12.0	21.1	36.1	46.0	43.5	33.7
C-DPM-BB vs. DPMv5	+30.7	+7.1	+12.3	-1.0	+19.4	+14.1	+18.4	+15.6	+15.1	+14.5
RCNN7-BB	61.2	69.1	68.6	58.7	33.4	62.9	51.1	62.5	64.8	58.5
RCNN7	56.1	60.6	66.8	54.2	31.5	52.8	48.9	57.9	64.7	54.2
RCNN5	43.1	57.7	59.0	45.8	28.1	50.8	40.6	53.1	56.4	47.3
C-DPM vs. RCNN5	-1.1	0.0	-3.0	-8.4	+2.0	-19.7	+9.8	+3.0	-4.8	-3.9

Table 3.3 – Results on PASCAL VOC 2007 (continues from table 3.2).

the true part score in the image domain. This way we can avoid computing costly convolutions at a large portion of the image, and focus only on promising locations. Importantly, we show that such gains in speed come at minimal cost in detection accuracy, allowing us to maintain a high-performing part detector. The motivation for our work has its roots in evidence from previous joint works [Vedaldi 2014], showing that accurate part detection is important for fine-grained object understanding and attribute prediction.

We have also investigated the effect of integrating CNN features into a DPM pipeline. Although we are able to significantly improve with respect to HOG-based DPMs, the mean AP that we achieve is still below the performance of recent methods such as R-CNN that use features from fully-connected layers. However, using only features from convolutional layers, we achieve a mean AP close to what R-CNN achieves based only on these layers, but within a more general structured part-based framework. Girshick *et al.* obtained similar results in a work that was done in parallel with ours [Girshick 2014b]. Our main positive conclusion from this work has been that there is great potential for the use of CNN features in conjunction within structured representations such as DPMs. For tasks where the explicit representation of structure is necessary, such as pose estimation or facial landmark localization this seems to be the method of choice– but when parts are only a means to an end, the use of implicit parts in CNNs seems advantageous. In our next works we therefore focused our research on entirely CNN-based representations, as described in the following Chapter.

Chapter 4

Deep Learning for Semantic Part Segmentation

Contents

In this chapter we use *fully convolutional* neural networks [Long 2014], which are particularly suited for dense image labelling, and focus on the task of segmenting an object into its parts for two diverse image domains : natural images (Section 4.1) and medical images (Section 4.2). In our experiments in Section 4.1.4 and Section 4.2.3 we show state-of-the-art performance for pedestrian parsing, face parsing and segmentation of sub-cortical brain structures. In Section 4.3 we conclude with a discussion on our findings. The work presented in this chapter has resulted in two conference submissions, to one machine learning and one medical image processing conference.

4.1 Semantic Segmentation of Object Parts

In this section we investigate the use of DCNNs to address the problem of semantic part segmentation, namely segmenting an object into its parts. This is illustrated in Figure 4.1. Given the bounding box of an object, the goal is to decompose it into its constituent parts.

Part segmentation is an important sub-problem for tasks such as recognition, pose estimation, tracking, or applications that require accurate segmentation of

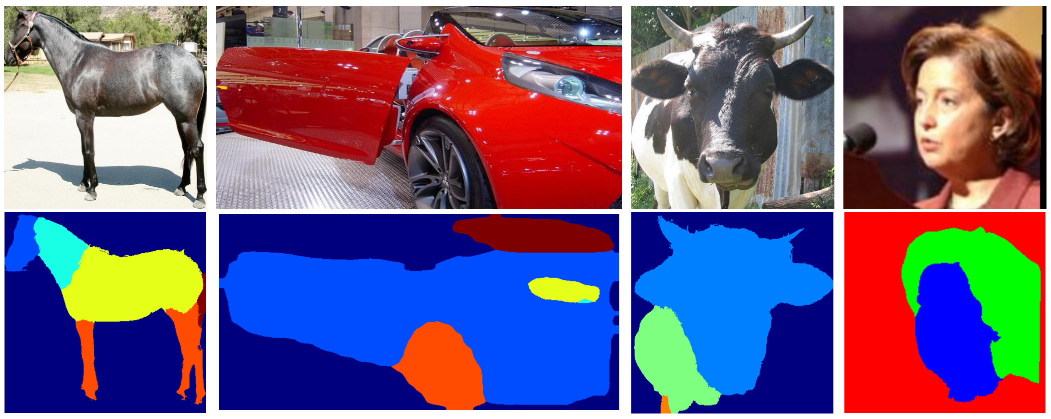


FIGURE 4.1 – In this work we address the problem of part segmentation. Compared to object localization (finding a bounding box) and object segmentation (finding the object mask), this problem is significantly more difficult. **Top :** Input images. **Bottom :** The output of our method.

complex shapes, such as medical applications dealing with radiotherapy of precisely delineated structures in sensitive areas of the human body. Another example is state-of-the-art methods for fine-grained categorization that rely on the localization and/or segmentation of object parts [Zhang 2014]. Part segmentation is also interesting from a modeling perspective, as the configurations of parts are, for most objects, highly structured. Incorporating prior knowledge about the parts of an object lends itself naturally to structured prediction, which aims at learning a function whose output space has a well defined structure. The key question addressed in this section is how DCNNs can be combined with structured output prediction to effectively parse object parts. In this manner, one can combine the discriminative power of CNNs to identify part positions and prior information about object layout to recover from possible failures of the CNN. Integrating DCNNs with structured prediction dates back to [LeCun 1998] who used Graph Transformer Networks for parsing 1D lines into digits. However, the combination of DCNN with models of shape, such as the Shape Boltzmann Machines, or of object parts, such as Deformable Part Models, are recent [Tompson 2014, Schwing 2015, Wan 2014].

This work makes several contributions. First, we show that by adapting the semantic segmentation system of [Chen 2014a] (Section 4.1.2), it is possible to obtain excellent results in part segmentation. This system uses a dense Conditional Random Field (CRF) applied on top of the output of a DCNN. This simple and non-specialized combination often outperforms specialized approaches to part segmentation and localization by a substantial margin. Second, we turn to the problem of augmenting this system with a statistical model of the shape of the object and its parts in Section 4.1.3. A key challenge is that the shape of parts is subject to substantial geometric variations, including potentially a variable number of parts per instance, caused by variations in the object pose. We model this variability using Restricted Boltzmann Machines (RBMs). These implicitly incorporate rich distributed mixture models in a representation that is particularly effective at capturing complex localized variations in shape. In order to use RBMs with DCNNs in a structured-output prediction formulation, we modify RBMs in several ways : first, we use hidden CRF training to estimate the RBM parameters in a discriminative manner, aiming at maximizing the posterior likelihood of the ground truth part masks given the DCNN scores as input. We demonstrate that this can yield an improvement over the raw DCNN scores by injecting high-level knowledge about the desired object layout. Third, in Section 4.1.5 we relax the requirement of having a tight bounding box around the object we want to parse and propose a simple approach for segmenting objects "in the wild". Extensive experimental results, reported in Section 4.1.4, confirm the merit of our approach on four different datasets.

4.1.1 Previous Work

We now review previous related works, dividing the section into three parts ; in Section 4.1.1.1 we discuss works that have adapted CNNs for tasks other than image classification ; in Section 4.1.1.2 we describe works tackling semantic segmentation of objects and parts ; in Section 4.1.1.3 we include papers related to Restricted Boltzmann Machines.

4.1.1.1 Convolutional Neural Networks : Beyond Image Classification

The breakthrough results of convolutional networks in image classification [Krizhevsky 2012] quickly stimulated interest in deep learning in other domains and tasks, such as object detection [Girshick 2014a, Girshick 2015, Ren 2015], texture classification [Cimpoi 2014], image segmentation [Farabet 2013, Long 2014, Chen 2014a], and shape recognition [Yu 2015]. There are two obstacles towards this goal. Firstly, deep neural networks rely to a great extent on large-scale, high-quality datasets, such as ImageNet in the case of image classification [Deng 2009]. Such datasets are not always available for other tasks, which increases the risk of overfitting. Secondly, even if we have at our disposal enough data, training a large network is time-consuming, lasting days or even weeks in some cases [Simonyan 2014].

These drawbacks can be circumvented by transferring knowledge across different domains. Knowledge transfer consists in exploiting already learned models or re-adapting them to address a different problem, or include an additional test. In the context of convolutional neural networks, instead of initializing the network parameters with random values, one can use a network that has been "pre-trained" on a large-scale task (*e.g.* Alexnet [Krizhevsky 2012] or VGG16 [Simonyan 2014]) to initialize the weights of some layers in the new model. One obvious restriction is that the the pre-trained network must have a similar structure to the new network, with only a few new, task-specific layers being introduced or modified. The learning rates for the layers that use pre-trained initialization are usually decreased or frozen entirely, favoring learning in the task-specific layers. With this approach training can converge faster, and overfitting can be avoided by limiting the number of parameters learned. The procedure we just described is referred to as *finetuning* and has become fairly standard in the deep learning community, with Alexnet and the deeper VGG16 architecture commonly used for weight initialization.

Recent examples of such a re-adaptation were [Long 2014, Chen 2014a] for semantic segmentation and entailed several technical modifications of the standard architectures used so far for classification. Convolutional neural networks are able to process input images of arbitrary dimensions, and produce dense feature maps of size dependent on their parameters. When used for classification, the connectivity type changes from sparse (convolution) to *fully-connected* towards the last layers. This is necessary in order to obtain a feature vector of fixed size that can be used in combination with a classifier. As a result, structured information present in the image is thrown away, making them inappropriate for tasks where the objective is a dense response at the pixel level. Long *et al.* build *fully convolutional (FC)* networks, replacing the last fully-connected layers with convolution layers, and train them for semantic object segmentation [Long 2014]. In their experiments they finetune on the PASCAL segmentation dataset, exploring the effect of various initializations for the convolution weights (Alexnet [Krizhevsky 2012], VGG16 [Simonyan 2014] and GoogleLeNet [Szegedy 2014]).

Two technical hurdles faced when applying convolutional neural networks to image labelling tasks, are the reduction of signal resolution caused by the repeated combination of max-pooling and sub-sampling, and spatial "insensitivity" (invariance). These traits impair precise localization and labelling at pixel accuracy, which is desired in the case of semantic segmentation. Papandreou *et al.* skip subsampling in the last two convolution layers and introduce *holes* in the respective convolution kernels [Chen 2014a], revisiting the "atrous" (with holes) algorithm, popularized in the wavelet community by Mallat [Mallat 1999]. Instead of deconvolution layers they use simple bilinear interpolation to upsample the output to the original image dimensions, making training and testing much more efficient. Finally, a fully connected CRF is employed as a post-processing step on the class scores, to refine segmentation results.

Another successful example of re-purposing a CNN for a task that goes beyond image classification is the work of Cimpoi *et al.* where the authors utilize the network of [Krizhevsky 2012] as a feature extractor, on which they construct two different types of descriptors for textured regions. The first one is simply the feature vector extracted from the penultimate fully-connected layer of a CNN(VGG-M [Chatfield 2014]), including the non-linear gating function. This can be considered as an object descriptor because features from the fully-connected layers capture the overall shape of the object contained in the region. The second descriptor is obtained by computing the output of the last convolutional layer of the CNN at multiple scales and pooling those features into a single Fisher Vector representation. The authors use low-level image cues to propose a set of regions that are subsequently classified into different types of texture using the above descriptors, showing state-of-the-art performance.

In contrast with the majority of recent works that leverage natural color photographs, Yu *et al.* use CNNs for sketch recognition [Yu 2015]. Free-hand sketches are potentially useful as a means of communication, especially in the context of modern multi-media devices, such as phones, tablets, watches, but their recognition can pose difficulties, due to the high degree of abstraction, detail, and variance. The authors in [Yu 2015] design a multi-scale, multi-channel, deep neural network that is one of first surpassing human performance on the same task. An important observation is that photo-oriented DCNNs led to little improvement over hand-crafted feature based methods when applied directly on sketches ; the proposed architecture and parameters are explicitly tailored for the task domain and the network is trained starting from a random initialization.

Besides demonstrating the effectiveness of CNNs in various and diverse tasks and domains, these works indicate that features learned in the hidden layers of DCNNs trained for large-scale image classification, encode generic, object- and shape-*independent* knowledge. This knowledge can be *transferred* to systems targeting dissimilar tasks (*e.g.* image-level classification versus pixel-wise labelling), *when the data domain is the same* (in this case natural RGB images). However, when the data domain changes radically –as in the case of free-hand sketches described above– it may be advantageous to change the network architecture and train starting from a random initialization.

4.1.1.2 Semantic Part Segmentation

The models presented above explicitly encode the decomposition of an object into parts, using at best a few mixture components. We now turn to the method we use in our work, which allows us to construct hidden variable models for part segmentations, giving us an additional level of freedom for the modeling of multi-model distributions.

Having presented how the knowledge embedded in the parameters of a CNN can be transferred to new tasks, we now focus on the particularities of the task we consider in this work. The layout of object parts (shape, for short) obeys statistical constraints that can be both strict (e.g. head attached to torso) and diverse (e.g. for hair). As such, accounting for these constraints requires statistical models that can accommodate multi-modal distributions. Statistical shape models traditionally used in vision, such as Active Appearance Models [Cootes 2001] or Deformable Part Models [Felzenszwalb 2010b] need to determine in advance a small, fixed number of mixtures (*e.g.* 3 or 6), which may not be sufficient to encompass the variability of shapes due to viewpoint, rotation, and object deformations.

A common approch in previous works has been combining appearance features with a shape model to enforce a valid spatial part structure. In [Bo 2011], the authors compute appearance and shape features on oversegmentations of cropped pedestrian images from the Penn-Fudan pedestrian dataset [Wang 2007]. They use color and texture histograms to model appearance and spatial histograms of segment edge orientations as the shape features. The label of each superpixel is estimated by comparing appearance and shape features to a library of exemplar segments. Small segments are sequentially merged into larger ones and simple constraints, (such as that "head" appears above "upper body" and that "hair" appears above "head") enforce a consistent layout of parts in resulting segmentations.

Winn *et al*. proposed the Layout Consistent Random Field (LayoutCRF) [Winn 2006] for detecting and segmenting partially occluded objects of a *known* class (car). LayoutCRF defines an initial part labeling that covers the whole object and then imposes asymmetric local spatial constraints on those labels, to ensure a consistent part layout. This way, for example, car wheels are constrained to lie below the car body, not vice-versa. LayoutCRF only works for side views of cars, a restriction that is relaxed in [Hoiem 2007], by virtue of a 3D model that is able to register parts across instances during training, allowing detection in a continuous range of viewpoints and scales.

Maire *et al*. treat detection and segmentation as a single, generalized eigenproblem to detect and segment people in the PASCAL 2010 dataset [Maire 2011]. They propose a framework that integrates low-level image cues wit top-down part detections from [Bourdev 2009b]. Pixels and parts appear as nodes in a graph whose edges encode both affinity and ordering relationships, while each part node takes the average value of its member pixel nodes to enforce consistency between part and pixel representations.

Arbeláez *et al*. focus on the representation and classification of individual regions, rather than on modelling relations between parts [Arbeláez 2012]. They start by generating an hierarchical tree of regions represented as a boundary image called Ultrametric Contour Map(UCM) [Arbelaez 2011]. These regions are then coupled with poselet part detections and global features [Hariharan 2011, Lazebnik 2006, Lazebnik 2006, Van De Sande 2010] to form features for semantic region classification. Individual regions are finally combined and projected to image pixels to create a segmentation mask for the full object.

More recent works capitalize on PASCAL Parts, a dataset comprising high quality part annotations for PASCAL 2010 train and validation sets [Chen 2014b]. Lu *et al*. formulate car parsing as a landmark identification problem [Lu 2014]. These keypoints are considered to lie on boundaries between neighboring car parts (body, window, lights, licence plates and wheels). The SWA hierarchical segmentation algorithm is used to provide segments at multiple scales [Sharon 2006], and graphical models with landmark points as nodes are used to represent cars at different viewpoints. Each pixel is associated with an image segment and the energy function for each graphical model is composed of two types of terms : a) unary terms at the landmark points and b) binary terms that encode spatial deformations and appearance consistency between their corresponding neighboring segments.

In [Wang 2015a] the authors draw inspiration from previous work on hierarchical models for objects [Zhu 2007b, Zhu 2010a, Jin 2006, Kokkinos 2011b] and propose a mixture of *compositional models* that represent horses and cows in terms of their boundaries and the boundaries of semantic parts. Their algorithm starts by segmenting large parts first, such as head, neck, torso, and moves on to segment legs, which are deformable and thus much more difficult to segment. Wang *et al*. introduce the concept of semantic compositional parts (SCP), according to which similar semantic parts are grouped and shared among objects of different class [Wang 2015b]. They use a two-stream fully convolutional neural network to compute potentials for SCPs and the full object, and then combine them to jointly infer object and part labels, treating SCPs as nodes in a fully-connected CRF.

4.1.1.3 Restricted Boltzmann Machines

Multi-modal distributions can be naturally captured through distributed representations [Hinton 1984], which represent data through an assembly of complementary patterns that can be combined in all possible ways. Restricted Boltzmann Machines (RBMs) [Smolensky 1986, Hinton 2006] provide a probabilistic distributed representation that can be understood as a discrete counterpart to Factor Analysis (or PCA), while their restricted, bipartite graph topology makes sampling efficient. Stacking together multiple RBMs into a Deep Boltzmann Machine (DBM) architecture allows us to build increasingly powerful probabilistic models of data, as demonstrated for a host of diverse modalities e.g. in [Salakhutdinov 2009].

The early studies of using RBMs and their deeper variants for digit modelling, *e.g.* [Hinton 2006] can be understood as the construction of statistical models on shapes. In [Eslami 2014] RBMs are thoroughly studied and assessed as models of shape. The authors additionally introduce the Shape Boltzmann Machine (SBM), a two-layer network that combines ideas from part-based modelling and DBMs, and show that it is substantially more flexible and expressive than single-layer RBMs. The same approach was extended to deal with multiple region labels (parts) in [Eslami 2014] and coupled with a model for part appearances. The layered architecture of the model allows it to capture both local and global statistics of the part shapes and part-based object segmentations, while parameter sharing during training helps avoid overfitting despite the small size of the training datasets.

RBMs are typically trained in a generative manner : given a set of shapes, the goal is to learn a probabilistic model that models the given shapes accurately and can generalize to unseen instances of the same shape category. Discriminative algorithms for training RBMs have mostly focused on using RBMs for classification tasks [Larochelle 2008, Mnih 2012] where the hidden variables of an RBM are used to solve a K-ary classification problem. By contrast, we are interested in predicting the structured data presented to the RBM during training - and as such need to resort to variants of RBMs tuned to Structured Output Prediction tasks [Mnih 2012].

The discriminative training of RBMs has been pursued in shape modelling by [Kae 2013] in a probabilistic setting and by [Yang 2014] in a max-margin setting. We pursue a probabilistic setting and detail our approach in Section 4.1.3. Despite small theoretical differences, the major practical difference between our method and the aforementioned ones is that we do not use any superpixels, pooled features, or boundary signals, as [Kae 2013, Yang 2014] do, but we rather entirely rely on the CNN scores.

Multi-layer Boltzmann machines can generate realistic multi-label segmentations but produce blurred boundaries among adjacent part regions. Conditional Random

Fields (CRFs) are well-suited to modelling local interactions between neighboring regions (*e.g.* superpixels) and are particularly effective when there is a clear difference between those regions. However, CRFs may fail to draw boundaries between regions with similar appearances, or when a region mingles with the background. In [Kae 2013], Kae *et al.* jointly train a CRF and a Restricted Boltzmann Machine (RBM) [Salakhutdinov 2007] to build a hybrid model that combines global consistency and clear region boundaries.

4.1.2 DCNNs for Semantic Part Segmentation

Following [Chen 2014a], we adopt the architecture of the state-of-art 16-layer classification network of [Simonyan 2014] (VGG-16). We employ it in a *fully-convolutional* manner, turning it into a dense feature extractor for semantic image segmentation. In particular, as in [Long 2014, Sermanet 2014, Oquab 2014], we treat the last fully-connected layers of the DCNN as 1×1 spatial convolution kernels. This allows us to utilize the DCNN as a feature extraction module that efficiently provides features, or class posteriors for a regularly sampled set of image rectangles. We learn the DCNN network parameters using training images annotated with semantic object parts at the pixel-level, minimizing the cross-entropy loss averaged over all image positions with Stochastic Gradient Descent (SGD). We initialize the network parameters from the ImageNet-pretrained VGG-16 model of [Simonyan 2014].

The model's ability to capture low-level information related to region boundaries is enhanced by employing the fully-connected Conditional Random Field (CRF) of [Krähenbühl 2011], exploiting its ability to combine fine edge details with long-range dependencies. We set the dense CRF hyperparameters by cross-validation, performing grid search to find the values that perform best on a small held-out validation set for each task.

In order to simplify the evaluation of the learned networks we fine-tune one network per object category, even though improvements for all part segmentation tasks could be possible by using a joint training procedure. The system is thoroughly evaluated in Section 4.1.4. Our results show that the method is effective in segmenting parts even for objects such as horses that exhibit complicated, articulated deformations. Second, although the model is not trained with a structured loss, it is apparent that it can implicitly capture some form of contextual information ; in many cases, it would be hard to successfully segment parts if it only used evidence local to each pixel.

While this model is very powerful, it can still make gross errors ; such errors could be corrected by introducing knowledge of the layout of objects, allowing for a better, more principled use of global information. Integrating this information is the goal of Section 4.1.3.

4.1.3 Conditional Boltzmann Machines

The aim of this section is to construct a probabilistic model of image segmentations that can capture prior information on the layout of an object category. The goal of this model is to complement and correct information extracted bottom-up from an image by the DCNN as explained in the previous section.

In order to construct this model, we introduce three types of variables : (i) the output $\mathbf{v}$ of the densely-computed DCNN that is visible during both training and testing ; (ii) the binary latent variables $\mathbf{h}$ that are hidden during both training and testing ; and (iii) the ground-truth segmentation labels $\mathbf{y}$ that are observed during training and inferred during testing. The latter is a one-hot vector for each pixel, with $\mathbf{y}_{i,k} = 1$ indicating that pixel i takes label k out of a set of K possible choices (the parts plus background).

The conditional probability $P(\mathbf{y}, \mathbf{h}|\mathbf{v}; W)$ of the labels and hidden variables given the observed DCNN features is the Boltzmann-Gibbs distribution

$$P(\mathbf{y}, \mathbf{h}|\mathbf{v}; W) = \frac{\exp(-E(\mathbf{y}, \mathbf{h}, \mathbf{v}; W))}{\sum_{\mathbf{y},\mathbf{h}} \exp(-E(\mathbf{y}, \mathbf{h}, \mathbf{v}; W))} \tag{4.1}$$

where $E(\mathbf{y}, \mathbf{h}, \mathbf{v}; W)$ is an energy function described below. The posterior probability of the labelling is obtained by marginalizing the latent variables :

$$P(\mathbf{y}|\mathbf{v}; W) = \sum_{\mathbf{h}} P(\mathbf{y}, \mathbf{h}|\mathbf{v}; W). \tag{4.2}$$

The goal is to estimate the parameters W of the energy function during training and to use $P(\mathbf{y}|\mathbf{v}; W)$ during testing to drive inference towards more probable segmentations.

Before describing the energy function $E(\mathbf{y}, \mathbf{h}|\mathbf{v}; W)$ in detail note that (i) the DCNN-based quantities $\mathbf{v}$ are always observed and the model does not describe their distribution ; in other words, we construct a *conditional* model of $\mathbf{y}$ [Lafferty 2001, He 2004] ; (ii) unlike common CRFs, there are also hidden variables $\mathbf{h}$, which results in a Hidden Conditional Random Fields (HCRFs) [Quattoni 2007, Murphy 2012] ; (iii) however, unlike the loopy graphs used in generic HCRFs, the factor graph in this model is bipartite, which makes block Gibbs sampling possible (Section 4.1.3.1).

Consider now the relationship between the DCNN output $\mathbf{v}$ and the pixel label $\mathbf{y}$ and recall that $\mathbf{v}$ are obtained from the last layer of the DCNN. The DCNN is trained so that, for a given pixel i, $\mathbf{v}_i$ contains the class posteriors produced by a softmax operation :

$$P(\mathbf{y}_{i,k} = 1|\mathbf{v}) = \frac{\exp(\mathbf{v}_{i,k})}{\sum_{k'=1}^{K} \exp(\mathbf{v}_{i,k'})}. \tag{4.3}$$

This suggests that $\mathbf{v}_{i,k}$ can be used as a bias term for $\mathbf{y}_{i,k}$ in the energy model, such that a larger value of $\mathbf{v}_{i,k}$ rewards the assignment $\mathbf{y}_{i,k} = 1$. The raw values of $\mathbf{v}$ are rescaled using a set of learnable parameters that can be calibrated during training. The contribution to the energy term is then :

$$E_{\text{CNN}}(\mathbf{y}, \mathbf{v}; W) = \sum_{i,k,k'} \mathbf{w}^{\text{C}}_{k,k'} \mathbf{y}_{i,k} \mathbf{v}_{i,k'}, \tag{4.4}$$

where the CNN calibration parameters, $\mathbf{w}^{\mathrm{C}}_{\cdot,\cdot}$ are contained in the overall model parameters W. Note also that this formulation allows to learn interactions between classes as class k' as predicted by the DCNN can vote through weight $\mathbf{w}^{\mathrm{C}}_{k,k'}$ for class k in the energy.

We can now write the term linking output and hidden variables, which takes the form of an RBM :

$$E_{\mathrm{RBM}}(\mathbf{y}, \mathbf{h}; W) = \sum_{i,j,k} \mathbf{y}_{i,k} \mathbf{w}^{\mathrm{R}}_{i,j,k} \mathbf{h}_j + \sum_{i,k} \mathbf{y}_{i,k} \mathbf{w}^{\mathrm{R}}_{i,k}. \tag{4.5}$$

Note that this does not include any 'lateral' connection between the observed variables, or between the hidden variables (this would correspond to terms in which pairs of the same type of variables are multiplied). Instead, there are two types of terms. The first type has biases $\mathbf{w}^{\mathrm{R}}_{i,k}$ for each pixel location i and label k, favoring certain labels based on their spatial location only. The second type expresses the interaction between labels and latent variables through the interaction weights $\mathbf{w}^{\mathrm{R}}_{i,j,k}$. These weights determine the effect that activating the hidden node $\mathbf{h}_j$ has on labelling position i as part k. Activating $\mathbf{h}_j$ will favor or discourage simultaneously the activation of labels at different locations according to the pattern encoded by the weights $\mathbf{w}^{\mathrm{R}}_{\cdot,j,\cdot}$ – intuitively latent variables can in this manner encode segmentation fragments.

The overall energy is obtained as the sum of these two terms :

$$E(\mathbf{y}, \mathbf{h}, \mathbf{v}; W) = E_{\mathrm{CNN}}(\mathbf{y}, \mathbf{v}; W) + E_{\mathrm{RBM}}(\mathbf{y}, \mathbf{h}; W) \tag{4.6}$$

By aggregating the output variables $\mathbf{y}$, the hidden variables $\mathbf{h}$, and the observable variables $\mathbf{v}$ into a single vector $\mathbf{z}$, the energy above can be rewritten in the form :

$$E(\mathbf{z}; W) = \mathbf{z}^T \mathbf{W} \mathbf{z} \tag{4.7}$$

where W is a matrix of interactions. Importantly, this matrix is block-diagonal to reflect the bipartite structure of the RBM, where only certain pairwise interactions are allowed. All these parameters can be estimated using discriminative training of RBMs [Mnih 2012, Kae 2013] as detailed below.

4.1.3.1 Parameter Estimation for Conditional RBMs

Given a set of M training examples $\mathcal{X} = \{(\mathbf{y}^1, \mathbf{v}^1), \ldots, (\mathbf{y}^M, \mathbf{v}^M)\}$, parameter estimation aims at maximizing the conditional log-likelihood of the ground truth labels :

$$S(\mathbf{W}) = \sum_{m=1}^{M} \log P(\mathbf{y}^m | \mathbf{v}^m; \mathbf{W}) \tag{4.8}$$

$$= \sum_{m=1}^{M} \log \sum_{\mathbf{h}} \frac{\exp(-E(\mathbf{y}^m, \mathbf{h}, \mathbf{v}; \mathbf{W}))}{Z(\mathbf{v}^m)}, \tag{4.9}$$

$$\text{where} \quad Z(\mathbf{v}^m) = \sum_{\mathbf{y},\mathbf{h}} \exp(-E(\mathbf{y}, \mathbf{h}, \mathbf{v}; \mathbf{W})). \tag{4.10}$$

Using the notation of Equation 4.7, a parameter $\mathbf{W}_{k,m}$ connects nodes $\mathbf{z}_k$ and $\mathbf{z}_m$ that can be either hidden or visible. The partial derivative of the conditional log-likelihood with respect to $\mathbf{W}_{k,m}$ is given by [Murphy 2012] :

$$\frac{\partial S}{\partial \mathbf{W}_{k,m}} = \sum_{m=1}^{M} \langle \mathbf{z}_k \mathbf{z}_m \rangle_{P(\mathbf{h}|\mathbf{y}^m,\mathbf{v}^m;\mathbf{W})} - \langle \mathbf{z}_k \mathbf{z}_m \rangle_{P(\mathbf{h},\mathbf{y}|\mathbf{v}^m;\mathbf{W})}$$

where $\langle \cdot, \cdot \rangle$ denotes expectation.

In order to compute the first term, the $\mathbf{y}$ and $\mathbf{v}$ components of the $\mathbf{z}$ vector are given and one has to average over the posterior on $\mathbf{h}$ to compute the expectation of $\mathbf{z}_k\mathbf{z}_m$. To do so, one starts with the CNN scores ($\mathbf{v}$) and the ground-truth segmentation maps ($\mathbf{y}$) and computes the posterior over the hidden variables $\mathbf{h}$, which can be obtained analytically. Then one computes the expectation of the product of any pair of interacting nodes, also in closed form.

In order to compute the second term one needs to consider the joint expectation over segmentations $\mathbf{y}$ and hidden variables $\mathbf{h}$ when presented with the CNN scores $\mathbf{v}$. The exact computation of this term is intractable, and is instead computed through Monte Carlo approximation using Contrastive Divergence [Hinton 2010]. Namely we initialize the state $\mathbf{y}$ to $\mathbf{y}^m$, perform $C = 10$ iterations of Block-Gibbs sampling over $\mathbf{y}$ and $\mathbf{h}$, and use the resulting state as a sample from $P(\mathbf{h}, \mathbf{y}|\mathbf{v}^m; \mathbf{W})$.

This training algorithm is identical to RBM training with the difference that the partition function is image-dependent, resulting in minor algorithmic modifications.

4.1.3.2 Implementation Details

In our implementation for training the RBM we used the whole batch for each update, so as to reduce training time and the number of parameters that had to be cross-validated. We observed that due to the small number of images in most of our training datasets (100-200) we had to use dataset augmentation to avoid overfitting. For every training image we generated 81 replicates by translating the images within a regular grid.

We used 200 iterations for training, while the decay terms (amounting to regularization coefficients) were set with cross-validation. Some indicative results are shown in Figure 4.2 ; we observe that each component is tuned to a different style, while combining these allows us to synthesize a rich set of face segmentation masks.

4.1.4 Experiments

We evaluate our method on four datasets (LFW,Penn-Fudan,CUB and PASCAL-parts) and report qualitative and quantitative results. We compare the accuracy of our pipeline before and after refining part boundaries using the fully-connected CRF, and also report on the improvements delivered by the combination of RBMs with CNNs on three categories (faces, cows, horses). While using the exact same settings

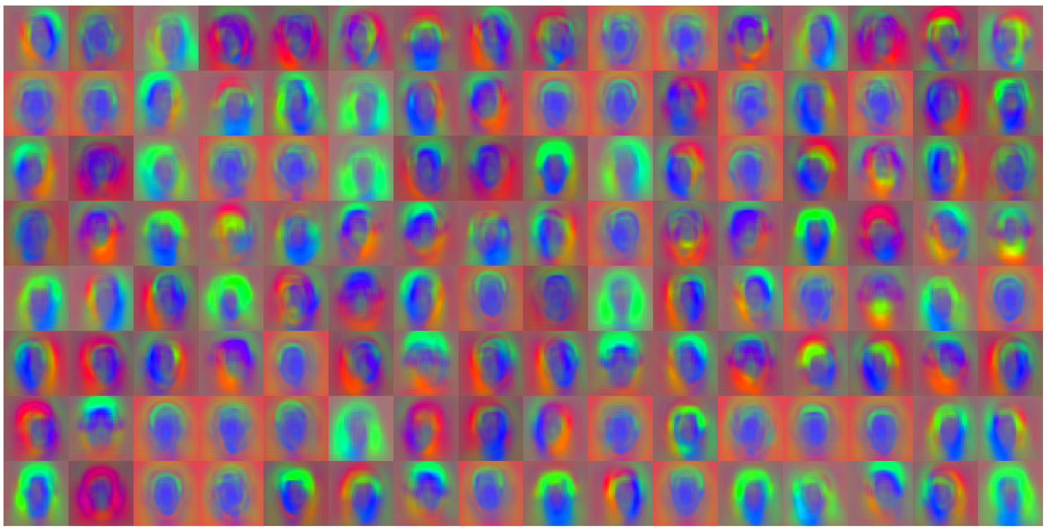


FIGURE 4.2 – Weights for 128 components of an RBM trained for faces on the LFW dataset ; green is for hair, red for background and blue for skin (best seen in color).

for the network and parameter values described in Section 4.1.2, we obtain state-of-the-art results when comparing to carefully engineered approaches for the individual problems.

Penn-Fudan pedestrian datataset

The Penn-Fudan dataset [Wang 2007, Bo 2011] provides manual segmentations of 170 pedestrians into head, hair, clothes, arms, legs and shoes/feet. This dataset does not come with a train/test split, so we had to train our networks on a different dataset. We train our network on the Pascal *person* category, using all images and corresponding part annotations from [Chen 2014b].

A complication is that in PASCAL-Parts clothing is not taken into account when segmenting people - the only regions are “torso”, “arms”, “legs” and “feet” ; whereas in Penn-Fudan the semantic parts used are “hair”, “face”, “upper clothes”, “arms”, “lower clothes”, “legs” and “shoes/feet”. To facilitate comparison of the methods, we merge “torso” and “arms” from PASCAL and “upper clothes” and “arms” from Penn-Fudan into “upper body” ; similarly we merge “legs” and “feet” from PASCAL and “lower clothes”, “legs” and “feet” from Penn-Fudan into “lower body”. Other methods also report results on these two super-regions, making comparison possible. Detailed numbers for Intersection-over-Union (IOU) for each part are included in Table 4.1.

Labeled Faces in the Wild

Labeled Faces in the Wild (LFW) is a dataset containing more than 13000 images of faces collected from the web. For our purposes, we used the “funneled” version of the dataset, in which images have been coarsely aligned using a congealing-style joint alignment approach [Huang 2007a]. This is the subset also used in [Kae 2013]

Method	head	upper body	lower body	FG	BG	Average
SBP [Bo 2011]	51.6	72.6	71.6	73.3	81.0	70.3
DDN [Luo 2013]	60.2	75.7	73.1	78.4	85.0	74.5
DL [Luo 2013]	60.0	76.3	75.6	78.7	86.30	75.4
Ours (CNN)	**67.8**	77.0	76.0	83.0	85.4	77.8
Ours (CNN+CRF)	64.2	**81.5**	**80.9**	**84.4**	**87.3**	**79.7**

Table 4.1 – Segmentation accuracies on the Penn-Fudan dataset [Wang 2007].

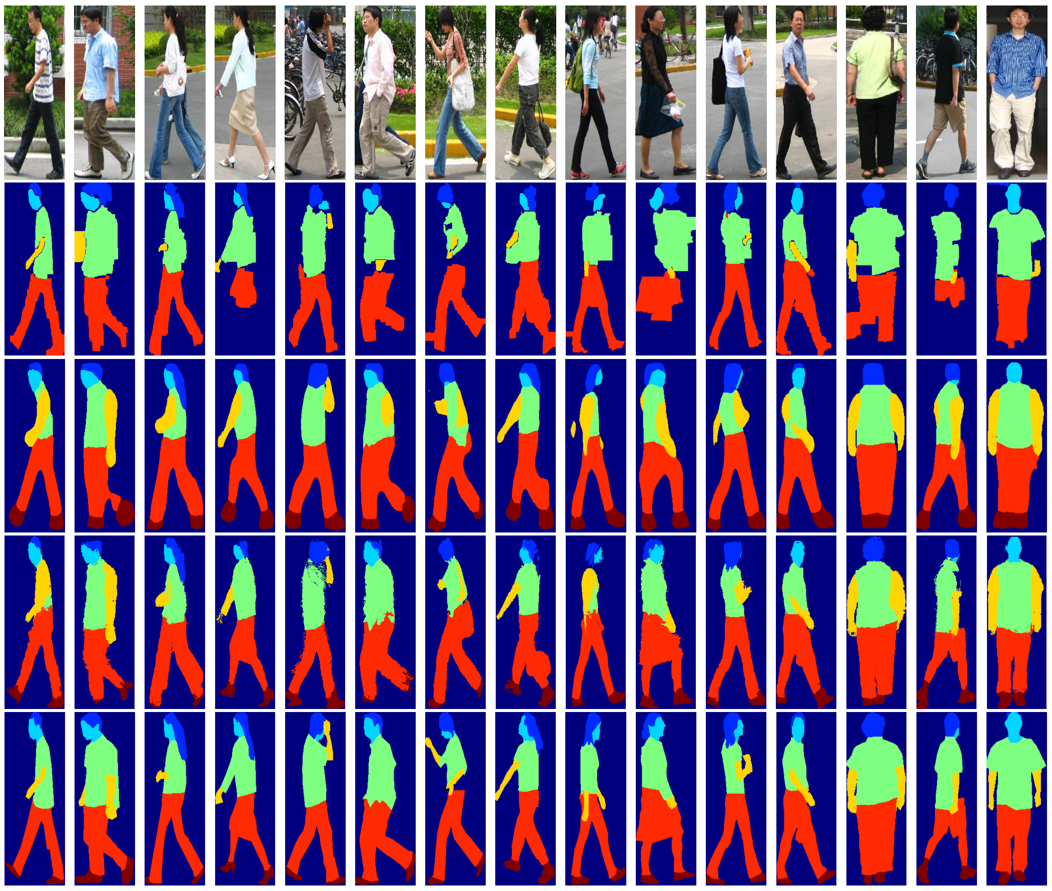


Figure 4.3 – Pedestrian parsing results on Penn-Fudan dataset. From top to bottom : a) Input image, b) SBP [Bo 2011], c) Raw CNN scores, d) CNN+CRF, e) Groundtruth.

and consists of 1500 train, 500 validation and 927 testing images of faces, and their corresponding superpixel segmentations, with labels for background, hair (including facial hair) and face. We train our DCNN on the 2000 trainval images and evaluate on the 927 test images, using superpixel accuracy as in [Kae 2013] for the purpose of comparison. Since our system returns pixelwise labels for each image, we employ a simple scheme to obtain superpixel labels : for each superpixel we compute a histogram of the pixel labels it contains and choose the most frequent label as the superpixel label.

Method	Accuracy (SP)
GLOC [Kae 2013]	94.95%
Ours (CNN)	96.54%
Ours (CNN+RBM)	96.78%
Ours (CNN+CRF)	96.76%
Ours (CNN+RBM+CRF)	**96.97%**

Table 4.2 – Superpixel accuracies on the LFW dataset [Huang 2007b].

Caltech-UCSD Birds-200-2011

CUB-200-2011 [Wah 2011] is a dataset for fine-grained recognition that contains over 11000 images of various types of birds. CUB-200-2011 does not contain segmentation masks for parts, however Zhang *et al.* [Zhang 2014] provide bounding boxes for the whole bird, as well as for its head and body. In that work, the authors describe a system for detecting object parts under two different settings : 1) When the object bounding box is given, and 2) when the location of the object is unknown.

We can assess the performance of our system by converting segmentation masks of bird parts to their (unique) corresponding bounding boxes. We train our system on the trainval bird part annotations in the Pascal Parts dataset and use the CUB-200-2011 test set (5793 images) for evaluation. We consider five parts : head, body, wings and legs and compare with [Zhang 2014]. We only focus on the case where the bounding box is considered to be known. Given the bird's bounding box, we compute the segmentation masks of four parts using our network : head, body, wings and legs. We then use the label masks for head and body to construct bounding boxes. Since our final goal is to convert a segmentation mask to a bounding box, sharp boundaries are not mandatory and we only utilize the coarse CNN scores. Results of our experiments are shown in table 4.3.

Method	head	body
Part R-CNN[Zhang 2014]	**68.19%**	79.82%
Ours	64.41%	**81.79%**

Table 4.3 – Percentage of Correctly Localized Parts (PCP) on the CUB-200 dataset

We measure accuracy in terms of PCP (Percentage of Correctly Localized Parts). Our simple approach proves effective and outperforms [Zhang 2014] by 2% in detecting bounding boxes for the bird's body. The part R-CNN is ahead by 4% in localizing birds' heads but this is a system that was specifically trained for this task. Furthermore, R-CNN capitalizes on the large number of region proposals (typically more than 1000) returned by the Selective Search algorithm [Uijlings 2013]. These bottom-up proposals can potentially be a bottleneck when trying to localize small parts, or when a higher IOU score is required [Zhang 2014]. In contrast, our approach yields a single, high-quality segmentation proposal for each object part,

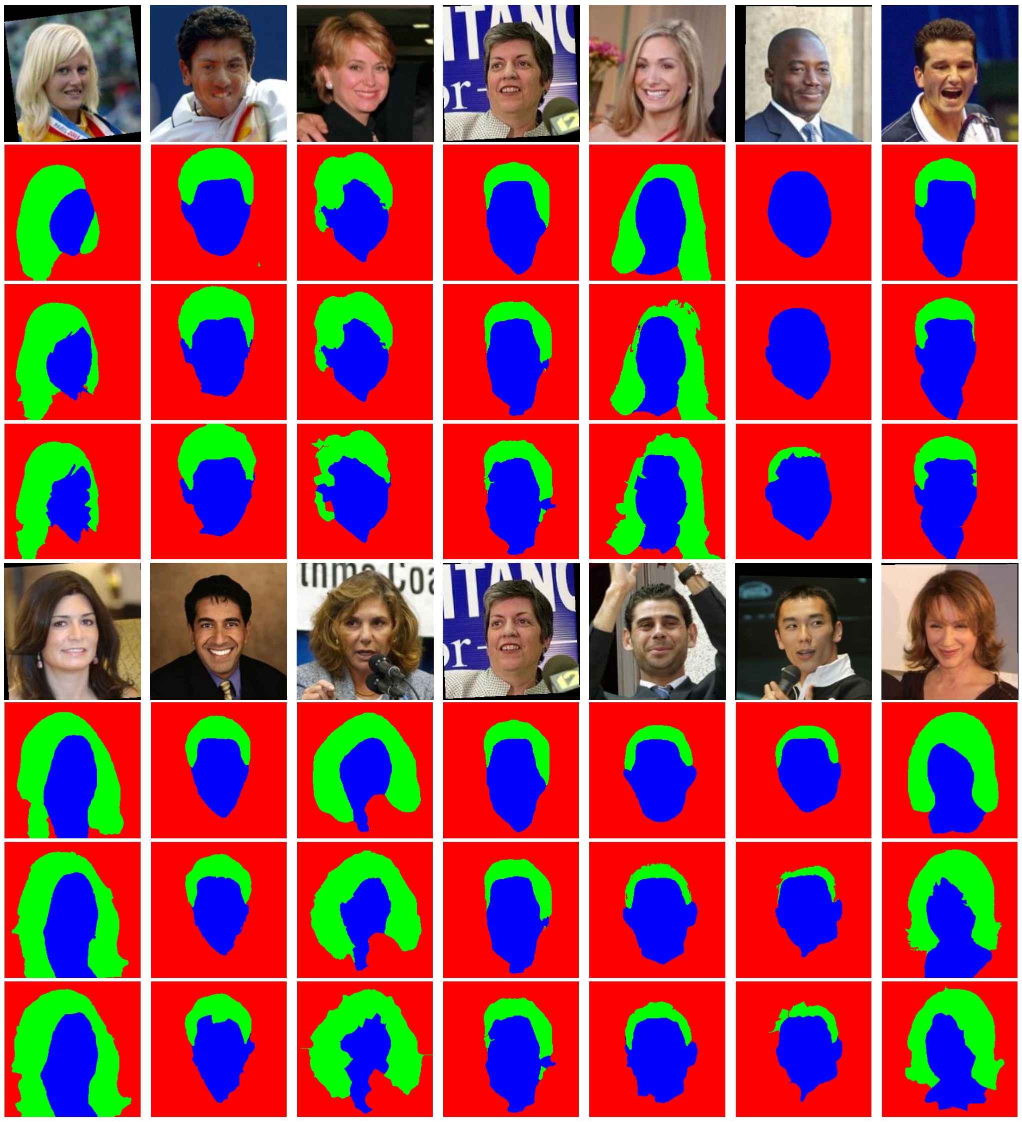


FIGURE 4.4 – Face parsing results on LFW. From top to bottom : a) Input image, b) Masks from raw CNN scores, c) CNN+CRF, d) Groundtruth. CRF sharpens boundaries, especially in the case of long hair. In the 6th image in the first row we see a failure case : our system failed to distinguish the man's very short hair from the similar-color head skin.

removing the need to score "partness" of hundreds or thousands of individual regions.

Pascal Parts dataset

In our last experiment, we evaluate our system on the PASCAL Parts dataset [Chen 2014b]. This dataset includes high quality part annotations for the 20 PASCAL object classes (train and val sets), but was released fairly recently, so there are not many works reporting part segmentation performance. The only work that we know of is by Lu *et al.* [Lu 2014] on car parsing, but the authors do not provide

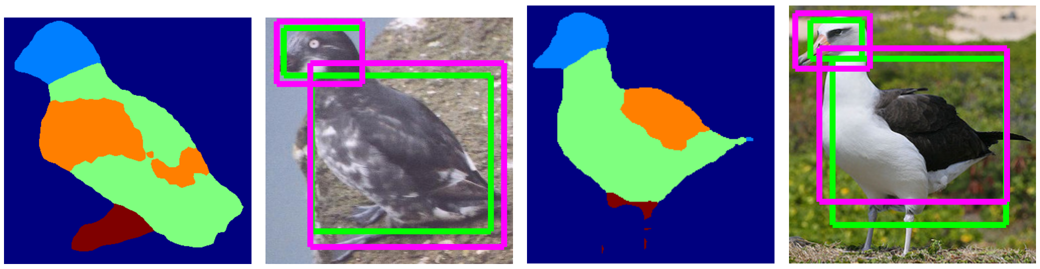


FIGURE 4.5 – Bird part segmentations returned by our network and inferred bounding boxes for head and body. We use green color for the groundtruth and magenta for our output.

Method	head	neck	torso	legs	tail	BG	Average
CNN	55.0	**34.2**	52.4	**46.8**	37.2	76.0	**50.3**
CNN+CRF	**55.4**	31.9	**53.6**	43.4	**37.7**	**77.9**	50.0

(a) Intersection-over-union (IOU) scores on PASCAL-Parts horse class

Method	head	torso	legs	tail	BG	Average
CNN	57.6	62.7	**38.5**	**11.8**	69.7	48.03
CNN+CRF	**60.0**	**64.8**	34.8	9.9	**72.4**	**48.38**

(b) Intersection-over-union (IOU) scores on PASCAL-Parts cow class.

Method	body	plates	lights	wheels	windows	BG	Average
CNN	73.4	**41.7**	**42.2**	**66.3**	61.0	67.4	**58.7**
CNN+CRF	**75.4**	35.8	36.1	64.3	**61.8**	**68.7**	57.0

(c) Intersection-over-union (IOU) scores on PASCAL-Parts car class.

TABLE 4.4 – Intersection-over-union (IOU) scores on PASCAL-Parts.

quantitative results in the form of some accuracy percentage, making comparison challenging.

Nevertheless, we report our own results for *horse, cow* and *car*, which could serve as a first baseline. For each class we train a separate DCNN on the train set annotations (using horizontal flipping to augment the training dataset), and test on the validation set. Our quantitative results are compiled in Tables 4.4, while in Figures 4.6 and 4.7 we show qualitative results.

In Tables 4.2 and 4.5 we report on the relative performance of the CNN-based system compared to the CNN-RBM combination, as well as the results we obtain when combined with the CRF system. For the "cow" and "horse" categories we also consider a separate subset of images containing poses of only moderate variation, to focus on cases that should be tractable for an RBM-based shape prior.

We observe that while the RBM typically yields a moderate improvement in per-

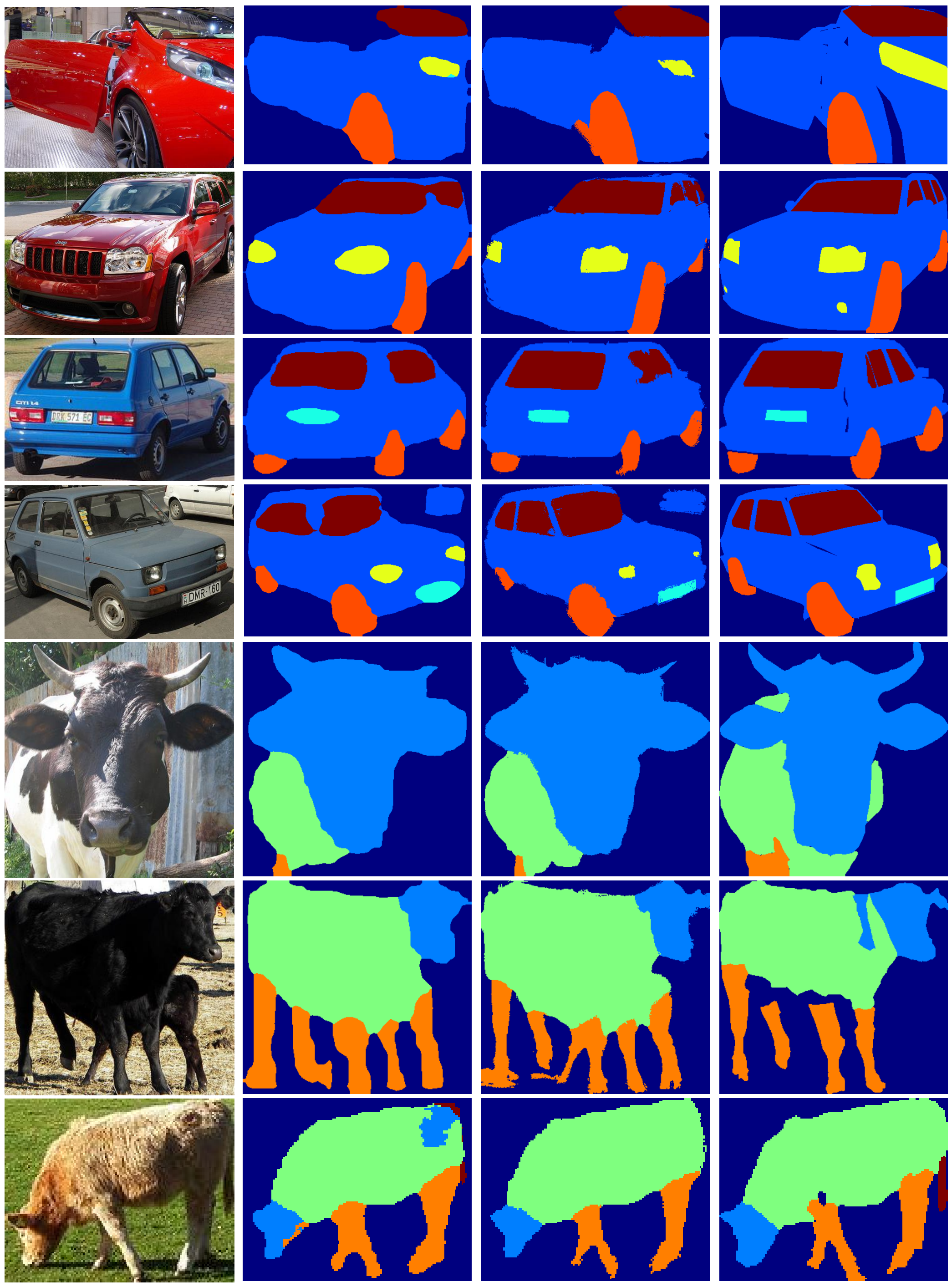


FIGURE 4.6 – Part segmentation results for car, and cow on the PASCAL-Parts dataset. From left to right : a) Input image, b) Masks from raw CNN scores, c) Masks from CNN+CRF, d) Groundtruth. Best seen in color.

formance over the CNN, this does not necessarily always carry over to the combination of these results with the CRF post-processing module. This suggests that also

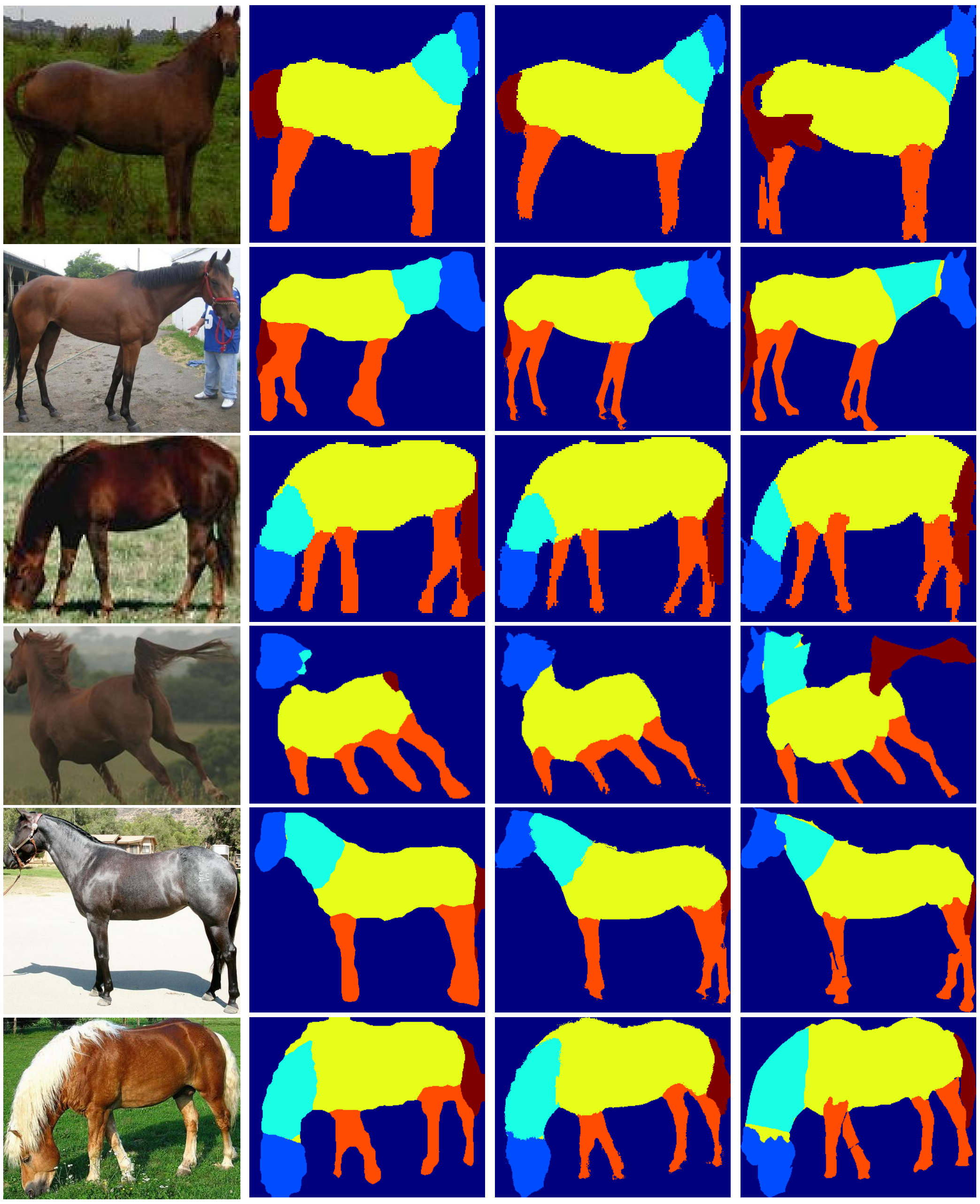


Figure 4.7 – Part segmentation results for horse on the PASCAL-Parts dataset. From left to right : a) Input image, b) Masks from raw CNN scores, c) Masks from CNN+CRF, d) Groundtruth. Best seen in color.

the CRF stage should be trained jointly, potentially along the lines of [Kae 2013].

Method	Val set	Val subset
CNN	77.3	83.7
CNN+RBM	77.7	84.4
CNN+CRF	79.1	85.1
CNN+RBM+CRF	79.2	84.7

(a) PASCAL-cow

Method	Val set	Val subset
CNN	76.6	86.4
CNN+RBM	76.3	86.9
CNN+CRF	77.6	88.1
CNN+RBM+CRF	76.7	87.6

(b) PASCAL-horse

TABLE 4.5 – Results due to joint RBM and CNN training.

4.1.5 Multi-scale Semantic segmentation in the Wild

In all the experiments described so far we assume that we have a tight bounding box around the object we want to segment. However, knowing the precise location of an object in an image is a challenging problem in its own right. In this section we investigate possible ways to relax this constraint, by applying our system on the full input image and segmenting object parts "in the wild". There are two ways to attack this task. An obvious approach would be to simply run the DCNN on the input image ; since the network is fully convolutional, the input can be an image of arbitrary height and width. A complication that arises is that our system has been trained using examples resized at a canonical scale (321×321), whereas an image might contain objects at various scales. As a consequence, using a single-scale model will probably fail to capture fine part details of objects deviating from its nominal scale. Another approach is to utilize an object detector to obtain an estimate of the object's bounding box in the image, resize the cropped box in the canonical dimensions, and segment the object parts as in the previous sections. The obvious drawback is that potential errors in detection – recovering a misaligned bounding box or missing an object altogether – harm the final segmentation result.

We explore a simple way of tackling these issues, by coupling our system with a recent, state-of-the-art object detector [Ren 2015]. We focus on the *person* class from the PASCAL-Parts dataset, and perform a new experiment in which we train our part segmentation network on the *train* set and use *val* to test our performance. Our approach consists of the following steps : We start by applying the CNN over the full image domain, in the original dimensions and also after upsampling by a factor of $1.5, 2$, for a total of three scales ; we did not use finer resolutions due to GPU RAM constraints. We then use [Ren 2015] to obtain a set of region proposals, along with their respective class scores. We keep *all* proposals, omitting any non-maximum suppression or thresholding steps, and use their bounding boxes as an indicator for scale selection. We associate each bounding box with its "optimal scale", namely the scale at which the bounding box dimensions are closer to the nominal dimensions of the network input

$$s_o = \arg\min_s |h_b^s - h_N| + |w_b^s - w_N|, \tag{4.11}$$

h_b^s, w_b^s being the box's height and width at scale s, and $h_N = w_N = 321$ being the

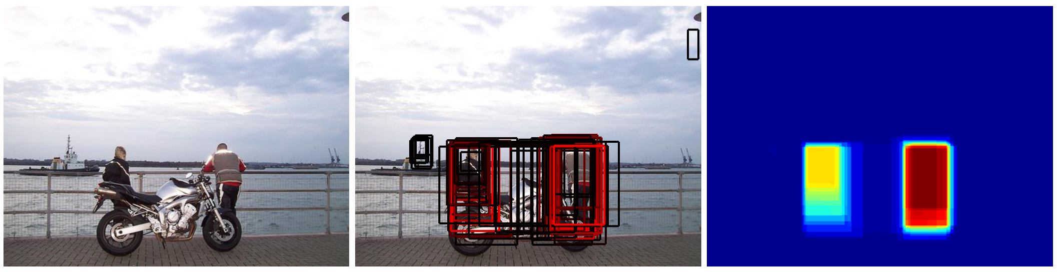

(a) Input image (b) Box proposals (c) Score contributions

FIGURE 4.8 – **4.8b :** Box proposals from [Ren 2015] for the *person* class. Box intensity is proportional to the corresponding detection score. **4.8c :** Heat map of the mean detection scores per pixel. Red indicates areas that are covered by more bounding boxes, with higher detection scores.

nominal scale at which the network was trained. CNN scores at an image location $\mathbf{x}$ are selected from the optimal scale of the box that contains it ; if $\mathbf{x}$ is contained in multiple boxes, we use the scale and scores supported by the box with the highest detector score.

This approach allows us to synthesize a map of part scores, combining patches from finer resolutions when the object is small, and coarser resolutions when the object is large. At test time, we extract all ground truth bounding boxes for the *person* class in PASCAL-Parts *val* set and calculate pixel accuracy within the boxes. As a baseline for comparison we use the naive evaluation of the CNN, applied on a single scale (original image dimensions). This simple approach boosts performance from 73.9% pixelwise accuracy to 74.7% *without* training the network with multi-scale data, even though end-to-end training could yield further improvements.

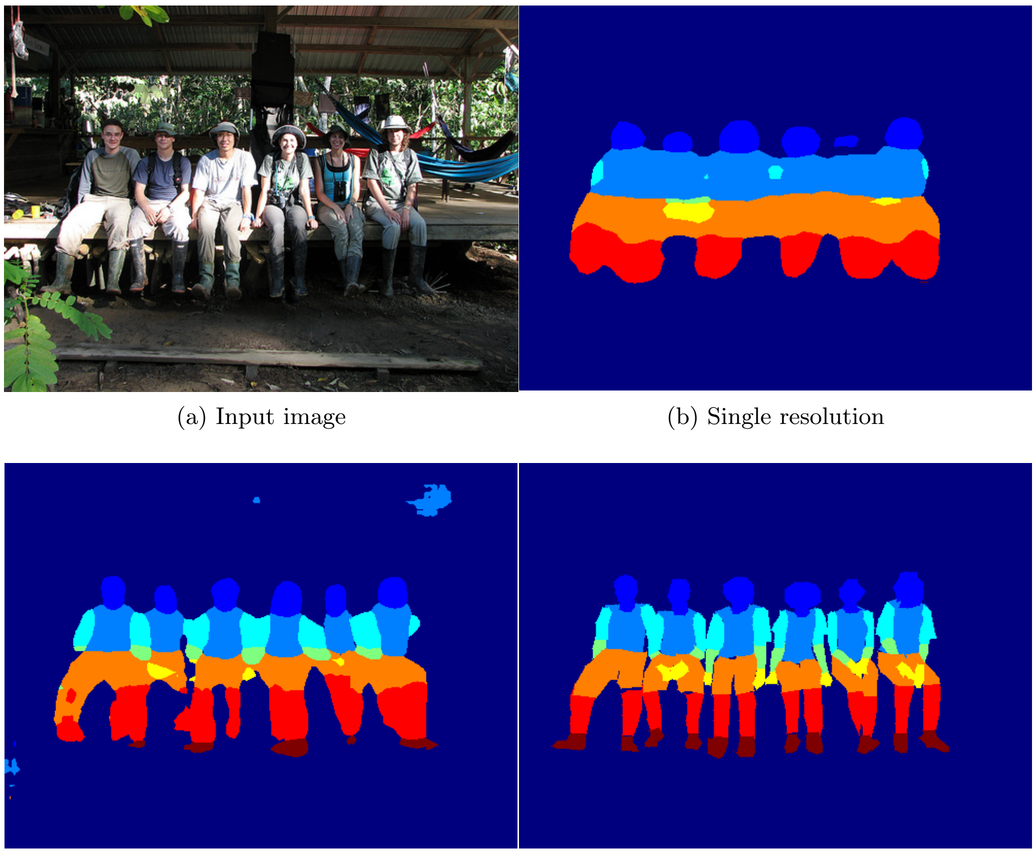

(a) Input image
(b) Single resolution
(c) Multi-scale scheme
(d) Ground truth

FIGURE 4.9 – Combining CNN features from multiple scales, we can recover segmentations for finer object parts.

4.2 Sub-cortical Brain Structure Segmentation Using Fully Convolutional Neural Networks

Image segmentation is a fundamental process in several medical applications. Diagnosis, treatment, planning and monitoring, as well as pathology characterization, benefit from accurate segmentation. In this section we investigate the segmentation of brain sub-cortical structures located at the frontostriatal system. Previous studies have shown the involvement of the frontostriatal structures in different neurodegenerative and neuropsychiatric disorders, including schizophrenia, Alzheimer's disease, attention deficit, and subtypes of epilepsy [Chudasama 2006]. Segmenting these parts of the brain enables a physician to extract various volumetric and morphological indicators, facilitating the quantitative analysis and characterization of several neurological diseases and their evolution.

In this section we propose a deep learning approach for segmenting sub-cortical structures of the human brain in Magnetic Resonance (MR) image data. We view the Deeplab network used in our experiments in Section 4.1 as a starting point and design a new architecture adapted to the task. Unlike previous CNN-based methods, our model operates directly on a full 2D image instead of image patches, without any alignment or registration steps at testing time. We further improve these first segmentation results by interpreting the CNN output as potentials of a Markov Random Field (MRF), whose topology corresponds to a volumetric grid. Alpha-expansion is used to perform approximate inference imposing spatial volumetric homogeneity to the CNN outputs. We compare the performance of the proposed pipeline with a similar system using Random Forest-based outputs, as well as state-of-art segmentation algorithms, and show promising results on two different brain MRI datasets.

Our work is similar in spirit to [Prasoon 2013], but with some notable differences. In [Prasoon 2013] the authors train one CNN for each of the three orthogonal views of MRI scans, for knee cartilage segmentation, with the loss being computed on the concatenated outputs of the three networks. The inputs to each CNN are 28×28 image patches and the output is a softmax probability of the central pixel belonging to the tibial articular cartilage. In contrast, our method operates on full 2D image slices, exploiting context information to accurately segment regions of interest in the brain. In addition, we use *fully convolutional* CNNs to construct dense segmentation maps for the whole image, instead of classifying individual patches. Furthermore, our method handles multiple class labels instead of delivering a foreground-background segmentation, and it does that efficiently, performing a single forward pass in $5ms$.

CNNs are characterized by large receptive fields that allow us to exploit context information across the spatial plane. Processing 2D slices individually, however, means that we remain agnostic to *3D context* which is important, since we are dealing with volumetric data. The obvious approach of operating directly on the 3D volume instead of 2D slices, would drastically reduce the amount of data available for training, making our system prone to overfitting, while increasing its computational requirements. Alternatively, we construct a Markov Random Field on top of the CNN output in order to impose volumetric homogeneity to the final results. The CNN scores are considered as unary potentials of a multi-label energy minimization problem, where spatial homogeneity is propagated through the pair-wise relations of a 6-neighborhood grid. For inference we choose the popular alpha-expansion technique that leads to guaranteed optimality bounds for the type of energies we define [Boykov 2001].

4.2.1 CNN Architecture

Given the very different nature of natural RGB images and MR image data (RGB *vs.* grayscale, varying *vs.* black background), we decided to train a CNN from scratch, instead of fine-tuning a pre-trained network such as VGG-16 [Simonyan 2014] or Alexnet [Krizhevsky 2012]. Training a deep network from a random initialization presents us with some challenges, discussed already in Section 4.1.1.1. Medical image datasets tend to be smaller than natural image datasets, and segmentation annotations are generally hard to obtain. In our case, we only have a few 3D scans at our disposal, which increases the risk of overfitting. In addition, the repeated pooling and sub-sampling steps that are applied to the input images as they get propagated through a CNN network decrease the output resolution, making it difficult to detect and segment finer structures in the human brain. To address these challenges, we make a series of design choices for our network : first, we opt for a shallower network, composed of five pairs of convolutional/max pooling layers. We sub-sample the input only for the first two max-pooling layers, and keep a stride of 1 for the remaining layers, introducing holes. This allows us to keep increasing the effective receptive field of filters, without further reducing the resolution of the output response maps. A $1-$pixel stride is used for all convolutional layers and 0.5 activation probability for all dropout layers. The complete list of layers and important parameters is given in Table 4.6.

Block	conv kernel	# filters	hole stride	pool kernel	pool stride	dropout
1	7×7	64	1	3×3	2	no
2	5×5	128	1	3×3	2	no
3	3×3	256	2	3×3	1	yes
4	3×3	512	2	3×3	1	yes
5	3×3	512	2	3×3	1	yes
6	4×4	1024	4	no pooling		yes
7	1×1	39	1	no pooling		no

Table 4.6 – Layers used in our architecture. All convolutional layers have a stride of one pixel ; a hole stride of "1" means that we introduce no holes.

4.2.2 Multi-label Segmentation Using CNN-based Inputs

At test time, a 2D image is provided to the CNN described in Section 4.2.1 and the output is a three-dimensional array of probability maps (one for each class), obtained via a softmax operation ; each map in the 3D array indicated the probability of every pixel to be part of a given brain structure $l \in \mathcal{L}$ (label). To obtain a brain segmentation at this stage, we simply resize the output to the input image dimensions using bilinear interpolation and assign at each pixel the label with the highest probability. However, we still need to impose volumetric homogeneity to the solution. We propose to do it using Markov Random Fields.

We consider the volume $P_i^{\text{CNN}}(l) : \mathcal{L} \rightarrow [0, 1]$ formed by the stacked CNN output slices, as evidence for the 3D structures of the brain, where i is indicating a voxel from our original image. Let $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ be a graph representing a Markov Random Field, where nodes in $\mathcal{V}$ are variables (voxels) and $\mathcal{E}$ is a standard 6-neighborhood system defining a 3D grid. Variables $i \in \mathcal{V}$ can take labels l_i from a labelspace $\mathcal{L}$. A labeling $\mathcal{S} = \{l_i \mid i \in \mathcal{V}\}$ assigns one label to every variable. We define the energy $E(\mathcal{S})$ which consists of unary potentials V_i and pair-wise potentials V_{ij} such that it is minimum when $\mathcal{S}$ corresponds to the best possible labeling.

Unary terms are defined as $v_i(l_i) = -\log(P_i^{\text{CNN}}(l_i))$, so that they assign low energy to high probability values. Pair-wise terms encode the spatial homogeneity constraint by simply encouraging neighbor variables to take the same semantic label. In order to align the segmentation boundaries with intensity edges, we made this term inversely proportional to the difference of the intensity I_i and I_j associated to the given voxels. The pair-wise formulation is $V_{i,j}(l_i, l_j) = w_{ij}.[l_i \neq l_j]$ where $w_{ij} = e^{-\frac{|(I_i - I_j)|^2}{2\sigma^2}}$. Finally, the energy minimization problem is defined as :

$$\mathcal{S}^* = \arg\min E(\mathcal{S}) = \arg\min \sum_{i \in \mathcal{V}} V_i(l_i) + \lambda \sum_{(i,j) \in \mathcal{E}} V_{i,j}(l_i, l_j). \tag{4.12}$$

$\mathcal{S}^*$ represents the optimal label assignment. Note that this energy is a metric in the space of labels $\mathcal{L}$; thus, it is guaranteed that using alpha-expansion technique we can find a solution $\hat{\mathcal{S}}$ for which the corresponding energy lies within a factor of 2 with respect to the optimal energy (i.e. $E(\hat{\mathcal{S}}) \leq 2.E(\mathcal{S}^*)$). Alpha-expansion is a well know move-making technique to perform approximate inference using graph cuts, which has shown to be accurate in a big range of vision problems. We refer the reader to [Boykov 2001] for a complete discussion about energy minimization using alpha-expansion.

4.2.3 Experiments and Discussion

We used the proposed method to segment a group of sub-cortical structures located at the frontostriatal network, including thalamus, caudate, putamen and pallidum. We evaluated our approach on two brain MRI datasets.

The first one is a publicly available dataset provided by the Internet Brain Segmentation Repository (IBSR) [Rohlfing 2012]. It contains 18 labeled 3D T1-weighted

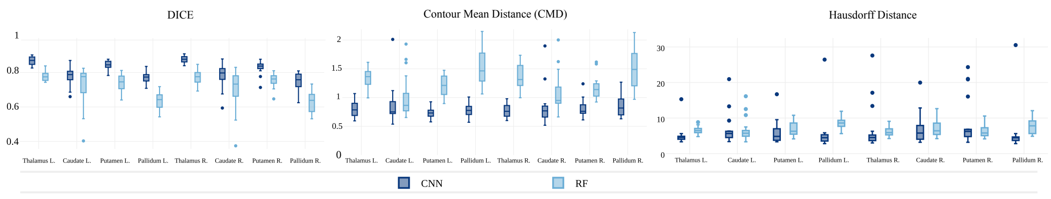


FIGURE 4.10 – Average Dice coefficient, Hausdorff distance, and contour mean distance on eight subcortical structures of IBSR dataset. The proposed CNN-based method outperforms the RF-based approach (better viewed in color and magnified).

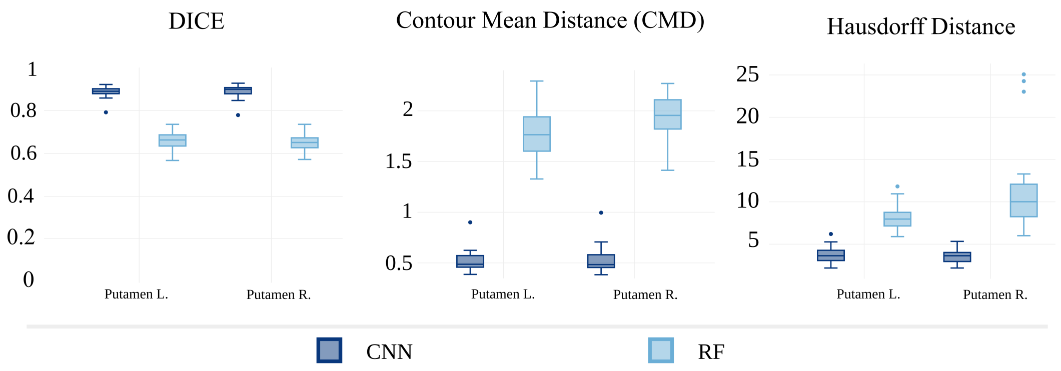


FIGURE 4.11 – The average Dice coefficient, Hausdorff distance, and contour mean distance on left and right putamen structure of RE dataset. The proposed CNN-based method generates more accurate segmentation results compared to the RF-based approach.

MR scans with slice thickness of around 1.3 mm. In this work we use the subset of 8 primarily subcortical labels, including left and right thalamus, caudate, putamen, and pallidum. The second dataset is obtained from a Rolandic Epilepsy (RE) study, including 17 children with epilepsy and 18 matched healthy individuals. For each participant, T1-weighted magnetic resonance images (MRI) were acquired with a 3 T scanner (Philips Acheiva) with an in-plane resolution of 256×256 and slice thickness of 1 mm. The left and right putamen structures were manually annotated by an experienced user. For both datasets, we process volumes slice by slice, after resizing them to 256×256 pixels. We treat these 2D slices as individual grayscale images to train our CNN.

In the first experiment, we compare the performance of our segmentation method using CNN evidence, with an approach based on Random Forest evidence, where the same MRF refinement is applied. The RF-based per-voxel likelihoods are computed in the same way as [Alchatzidis 2014]. Then, the RF probability maps are considered as the unary potentials of a Markov Random Field and alpha-expansion is used to compute the most likely label for each voxel, as explained in Section 4.2.2. Figure 4.10 and Figure 4.11 show the average Dice coefficient, Hausdorff distance, and contour mean distance between output segmentation and the ground truth for dif-

Table 4.7 – The average Dice coefficient of the three methods on different brain structures. Values are reported as the average of the left and right structures.

	Proposed	Freesurfer	FSL
IBSR-Thalamus	**0.87**	0.86	0.85
IBSR-Caudate	0.78	**0.82**	0.68
IBSR-Putamen	**0.83**	0.81	0.81
IBSR-Pallidum	**0.75**	0.71	0.73
RE-Putamen	**0.89**	0.74	0.88

ferent structures. These results show that the CNN-based approach achieves higher Dice compared to RF-based method, while producing lower Hausdorff and contour mean distance.

In the second experiment, we compare the accuracy of our proposed method to two publicly available state-of-the-art automatic segmentation toolboxes, Freesurfer [Fischl 2002], and FSL-FIRST [Patenaude 2011]. In Table 4.7 we report the average Dice coefficient of the left and right structures computed by the three segmentation methods : the proposed CNN-based approach, Freesurfer, and FSL. These results show that our method provides better segmentations compared to the state-of-the-art for three sub-cortical structures in both IBSR and RE dataset. However, Freesurfer results in better segmentation for caudate in the IBSR dataset which could be attributed to the limitation of CNN in capturing thin tail areas of the caudate structures.

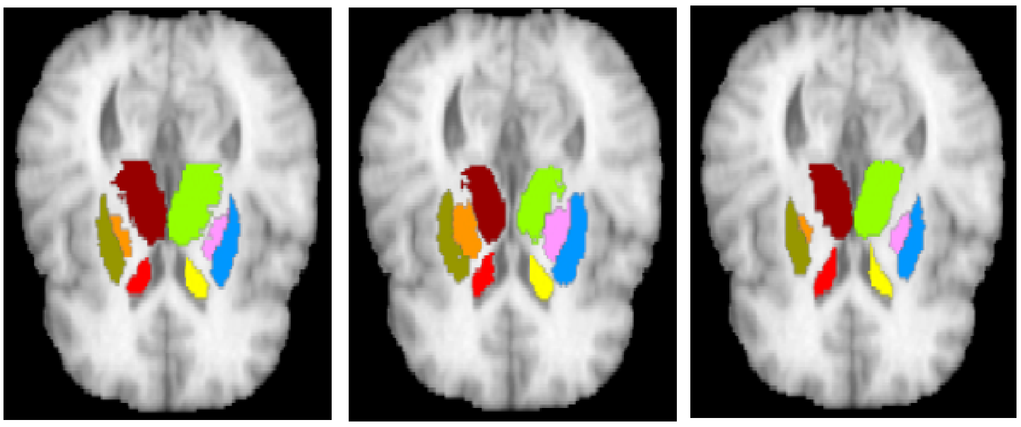


Figure 4.12 – 2D slice segmentation (IBSR). **Left** : Groundtruth. **Middle** : RF-based results. **Right** : CNN-based results.

4.2.4 CNN Training and Evaluation Details

Training

The input to our network is a single 2D slice from a 3D MRI scan, along with the corresponding label map. We apply data augmentation to avoid overfitting : we use horizontally flipped and translated versions of the input images by 5, 10, 15, 20 pixels, across the x/y axes. Other transformations, such as rotation, could be considered as well. The MR image data are centered and the background always takes zero values, so we do not perform mean image subtraction as is usually the case.

In the case of IBSR, we split the available data into three sets. Each time, we use two of the sets as training data (approximately $100K$ training samples) and the third set as test data. One of the training data volumes is left out and used as validation data. Similarly, we split RE into two subsets of equal size, using one for training and one for testing, each time. We train on both datasets for 35 epochs starting with a learning rate of 0.01 and dropping it at a logarithmic rate until 0.0001. For training, we use standard SGD with a momentum of 0.9 and a softmax loss. For all our experiments we used MATLAB and the deep learning library MatConvNet [Vedaldi 2015].

Testing

At test time, we are given a 3D volume containing N slices. We process each slice individually, resizing the slice to the nominal 256×256 dimensions. For a 256×256 input image, the total sub-sampling factor of the network is 4, resulting in a $64 \times 64 \times L$ array of scores, where L is the number of class labels. These scores are turned into probabilities after applying a softmax operation, resulting in a $64 \times 64 \times L$ probability array P. Using MATLAB notation, $P(:, :, l)$ is a 2D map, containing the probability of each pixel belonging to class l.

Comments

We also experimented with CNNs trained on 2D slices from the other two views (sagittal and coronal) but the resulting models performed poorly. The problem is rooted in the inherent symmetry of some brain structures and the fact that the CNN is evaluated on individual slices, ignoring 3D structure. For instance, when processing slices across sagittal view, the right and left putamen appear at roughly the same positions in the image. They are also very similar in terms of shape and appearance, which fools the system into assigning the same label to both regions. This simple example demonstrates the need for richer priors that take into account the full volume structure to assign class labels.

4.3 Summary

We have demonstrated that a simple and generic system for semantic segmentation relying on Deep CNNs and Dense CRFs can provide state-of-the-art results in the task of semantic part segmentation, outperforming highly sophisticated techniques that have been carefully engineered in a category-specific manner. We explored methods of integrating high-level information through a joint discriminative training of the network with a statistical, category-specific shape prior, showing that these can act in a complementary manner to the bottom-up information provided by DCNNs. We also proposed a simple, yet effective multi-scale scheme for segmentation "in the wild", guided by a fast object detector, that is used both to propose possible object boxes, and select the appropriate scale for segmentation in a pyramid of CNN scores.

In the second part of the chapter, we proposed a deep learning framework for segmenting frontostriatal sub-cortical structures in MR images of the human brain. We trained a fully convolutional neural network for segmentation of 2D slices and treated the output probability maps as a proxy for the respective voxel likelihoods. We further improved segmentation results by using the CNN outputs as potentials of a Markov Random Field (MRF) to impose spatial volumetric homogeneity. Our experiments show that the proposed method outperforms approaches based on outputs from other classifiers, as well as state-of-the-art segmentation methods. However, we also note some limitations : the current model is not able to accurately capture thin tail areas of the caudate structures. Second, symmetric structures confound the CNN training process when considering views which are parallel to the plane of symmetry. Third, graph-based methods have to be used to impose volumetric consistency since training is done on 2D slices. Different network layouts, taking account of volumetric structure can possibly help overcome these limitations.

Chapter 5

Conclusions and Future Work

Contents

5.1 Introduction

In this thesis we have focused on the use of mid-level representations for modelling objects of various types. These representations are discriminative, can be shared among different object classes, often carry rich semantic information, and can benefit high-level vision applications, such as object detection, image classification, pose estimation, and fine-grained recognition. Our work can be partitioned according to the three different types of structures we considered : medial axes, object parts, and learned feature representations in convolutional neural networks. We have made contributions, in accurate and efficient medial axis detection and benchmarking, efficient object detection, and the use of CNN features for object detection and dense pixel labelling of object parts in natural and medical images.

Publication List

A large part of the work described in this thesis has been preliminarily presented in several high-quality conference publications [Tsogkas 2012, Vedaldi 2014, Savalle 2014], as well as three conference submissions that are currently under review. Two additional publications [Trulls 2014, Chandra 2015] have been the result of collaborations on topics that are related technique-wise with the methods presented here (DPMs and DCNNs for segmentation) but were not included in this manuscript as they were not stemming from the author's main research focus. A journal paper on symmetry detection is currently under preparation for submission.

Below is a complete list of publications. In Section 5.2 we present our contributions in more details and in Section 5.3 we discuss possible future directions and extensions.

1. Learning-Based Symmetry Detection in Natural Images, ECCV 2012,
   S. Tsogkas, I. Kokkinos.
2. Understanding Objects in Detail with Fine-grained Attributes, CVPR 2014,
   A. Vedaldi, S. Mahendran, **S. Tsogkas**, S. Maji, B. Girshick, J. Kannala, E. Rahtu, I. Kokkinos, M. B. Blaschko, D. Weiss, B. Taskar, K. Simonyan, N. Saphra, S. Mohamed.
3. Segmentation-aware Deformable Part Models, CVPR 2014,
   E.Trulls, **S. Tsogkas**, I. Kokkinos, A. Sanfeliu, F. Moreno.
4. Deformable Part Models with CNN Features, ECCV 2014 Parts and Attributes workshop,
   P.-A. Savalle, **S. Tsogkas**, G. Papandreou, I. Kokkinos.
5. Accurate Human-Limb Segmentation in RGB-D images for Intelligent Mobility Assistance Robots, ICCV 2015 Third Workshop on Assistive Computer Vision and Robotics,
   S. Chandra, **S. Tsogkas**, I. Kokkinos.
6. Deep Learning for Semantic Part Segmentation with High-Level Guidance (under submission to ICLR 2016),
   S. Tsogkas, I. Kokkinos, G. Papandreou, A. Vedaldi.

5.2 Contributions

5.2.1 Medial Axis Detection

Datasets : In Chapter 2 we have focused on improving medial axis detection in natural images. As there were no existing datasets specifically for medial axes in cluttered RGB images at the time, our first step was to create a set of ground truth annotations that could be used for comparison with other methods. Over the course of our research we have constructed two datasets with medial axis annotations for all color images in the Bekeley Segmentation Dataset [Martin 2001]. The first one, SYMMAX300, is a semi-automatically constructed ground truth that is based on automated skeleton extraction of manually selected symmetric segments from BSDS300. In later work we have constructed an upgraded version of SYMMAX300 based on BSDS500, denoted by SYMMAX500. For the latter we manually annotated all 500 images in BSDS500, using a graphical user interface we implemented in MATLAB. These datasets, as well as the tools to expand them, are freely available and they provide a benchmark for training and evaluating relevant algorithms.

Improving detection in RGB images : In terms of algorithms and experiments, we have made several contributions towards accurate and efficient detection of medial axes in natural images. Most existing algorithms for the task of skeleton/ridge extraction operate on binary shapes or grayscale images, and often depend on a well localised and closed boundary. In contrast, we have exploited the two datasets we have constructed to propose a machine learning algorithm that directly localizes medial contours in cluttered RGB images without any direct knowledge about the object boundary. We have used multiple instance learning to deal with the unknown scale and orientation, treating them as latent variables during training. Our algorithm significantly outperforms other methods in experiments evaluating detection accuracy on SYMMAX300.

Speeding up medial axis detection : In further work we we have reduced the running time of our symmetry detector at test time, through a combination of sketch tokens, a mid-level contour representation, and random forest (RF) classifiers. Our RF-based variant runs in less than 1 second for a 321×421 input image, which amounts to a 40-fold speedup compared to the MIL-based version. In experiments on SYMMAX500 we have also demonstrated the beneficial effects of grouping pixel responses to contours, achieving higher accuracy, especially when high precision is demanded.

Applications : Our medial axis detectors can be readily applied to natural images, detecting local symmetries efficiently and with high accuracy. They have already been used for text detection [Zhang 2015] and for constrained foreground-background segmentation in images [Teo 2015]. Another possible application where these class-agnostic representations could be used is object and object part proposal, as symmetry is often associated with salient structures in the image. Our system can also be re-trained for specialized tasks. For instance, given the appropriate datasets, we could learn to delineate local symmetries of roads in aerial photographs or the medial axes of neurons in medical images.

5.2.2 Object Part Modeling

Efficient Object Part Detection : In the second part of the thesis we have developed a coarse-to-fine algorithm to perform efficient detection of object parts using HOG templates. In our experiments we have first shown that we can improve detection accuracy of parts if we have enough training data and use a large number of templates to model part appearance. To mitigate the increased computational complexity incurred by the higher number of filters evaluated at test time, we have proposed to organize HOG templates in a binary tree, merging them iteratively according to their pairwise similarities. As a second step, we have used the Chebyshev inequality to construct probabilistic bounds that approximate the true part scores provided by convolutions in the image domain.

Our method is independent of the training procedure and allows us to process an input image in a hierarchical fashion using the probabilistic bounds to prune large parts of the spatial domain. Focusing only on the (few) image regions that hold high promise of containing the part under question, we have been able to achieve a four-fold or even five-fold speedup, with minimal loss in detection accuracy. Since using a large number of templates to model parts is no longer computationally prohibitive, we can better exploit larger datasets exhibiting high intra-class variation. Our algorithm is also flexible, permitting us to trade accuracy for efficiency at will, depending on the task requirements. For example, in Section 3.2.3 we have shown that we can increase the speedup from four-fold to seven-fold, with only a moderate 2% loss in detection accuracy.

Improving DPM Detection with CNN Features : Continuing our work with deformable part models, we have investigated the effect of replacing HOG-based templates with CNN features and integrating them in a DPM pipeline. In our experiments we have shown that using learned features to model the object and its parts leads to significant improvements over hand-crafted features, increasing mean average precision by 14.5% in the VOC2007 dataset. We have tackled the increased computational complexity by training in a whitened feature space [Girshick 2013], and using a patchwork representation of an image pyramid for efficient computation at test time [Dubout 2012]. Our method requires less than 2 seconds for a single input image.

5.2.3 Fully Convolutional Neural Networks for Semantic Segmentation of Object Parts

Part Segmentation in Natural Images : In the last part of the thesis we have explored the semantic segmentation of object parts using fully-connected deep convolutional neural networks in natural and medical images. In a first series of experiments, we have shown that we can successfully re-purpose a deep network trained for large-scale image classification [Simonyan 2014], for pixel-wise labeling of object parts. Coupling that with a fully-connected CRF, we have obtained state-of-the-art results in a variety of object classes and datasets, indicating that intermediate layers of deep CNNs learn useful representations that can be transferred across tasks and across object types. We have used a Restricted Boltzmann Machine trained on ground truth part segmentations to infuse prior shape knowledge in our model and have shown that in some cases it can improve performance by correcting unrealistic labelings. We have also addressed part segmentation "in the wild", namely when we do not know the precise location of objects in the image. We have proposed a practical approach that consists in using a state-of-the-art object detector that jointly suggests probable object regions and assigns a class score to each one of them. We have used the output of the detector to guide feature selection from a pyramid of CNN responses extracted at varying image resolutions, and create a multi-scale feature "patchwork". This simple method permits us to share computation among overlapping regions and segment the complete image domain without treating each proposed bounding box separately. Exploiting information from multiple scales also helps us recover small object parts that often get lost due to subsampling.

Brain Segmentation in Magnetic Resonance Images : In addition, we have demonstrated that similar approaches can be used to segment frontostriatal sub-cortical structures in magnetic resonance images of the human brain. We have proposed a CNN architecture for the segmentation of the 2D slices composing the MRI volume into different parts and we have treated the output probability maps as a surrogate for the respective voxel likelihoods. Since working on 2D slices ignores structural correlations in the 3D volume, we have used the CNN outputs as unary potentials in a Markov Random Field (MRF) defined on a 3D grid, to impose spatial volumetric homogeneity. Our approach achieves state-of-the-art results in two different datasets compared with previous segmentation methods and a similar approach using random forests. In a paper that is under submission we have used the segmentations provided by our CNN to guide a co-registration and co-segmentation, replacing the ground truth segmentations that are normally registered to the target image. CNN-based results are very close to the performance of an “oracle” using the ground truth masks, indicating that high-quality segmentations can act as an effective surrogate when ground truth annotations are not available, validating the potential impact of our method.

5.3 Future Work

A first possible direction for future work is the extension of the SYMMAX500 dataset introduced in Chapter 2. Our annotations at the moment contain only binary information concerning each image pixel, namely whether a given pixel lies in some medial axis or not. However, as explained in Section 2.2.1 the complete medial representation if formed as a pair of a point in the 2D space, and the distance of the medial point from the object boundary (or alternatively, the two vectors that connect the medial point to two points on the object boundary). Annotating correspondences between medial and boundary points would essentially give us the scale of the object at each location, which in turn could be used to increase supervision during training. Another important extension would be to group pixels into contours and label them as *foreground* or *background*. This type of information could be exploited for example to train object proposal systems based on symmetry contours, or better exploit ridges for constrained foreground/background segmentation [Teo 2015]. Furthermore, CNNs could be used to improve medial axis detection accuracy, in a similar spirit to recent works on boundary detection [Xie 2015].

Another problem on which further research is needed is the semantic segmentation of object parts. We have shown some promising results in Chapter 4 and other researchers have been working on the same problem in parallel with us [Lu 2014, Wang 2015a, Wang 2015b]. Directly parsing objects and their parts on the full image domain (“in the wild”) is of particular interest, as it allows as to share feature computation for all object classes and instances. Moreover it would allow us to better exploit context, in contrast to methods that segment an object to its parts given a tight bounding box around it. The value of having a precise delineation of the object and part masks for other computer vision tasks should also be investigated. For instance, a reasonable assumption is that part segmentation will be beneficial in fine-grained tasks, where minute details in object parts often rise as the discriminative factor between subordinate classes ; in these cases, background information should probably be ignored [Nilsback 2008, Chai 2012, Parkhi 2012].

Finally, we are interested in exploring ways of enforcing volumetric consistency in 3D volumes, *e.g.* Magnetic Resonance Images. There are two avenues that are worth exploring : the first one follows the same principle as the method we described in Section 4.2, and consists in computing dense pixelwise confidence scores for individual *2D slices* and trying to enforce volumetric consistency among different structures post-hoc. A second way of tackling the problem is learning feature representations directly on the 3D space, using for example CNNs with 3D convolutional kernels. Although there have been some works that follow this route for volumes of spatio-temporal data [Ji 2013], 3D CNNs are largely unexplored in the context of medical imaging.

5.4 Closing Remarks

The representations and tools we have used in this thesis span a broad range, from lower level structures such as contours and hand-crafted features, to learned representations in convolutional neural networks and semantically meaningful object parts. Perhaps the most interesting observations are related to the interplay between these structures and the fact that they can be used to describe and synthesize different types of objects ; contour fragments combine to represent different shapes, parts are arranged in a specific layout to model an object (DPMs), and so on. Currently, the way to model this type of object structure and achieve state-of-the-art performance for most high-level tasks is to use hierarchically learned feature representations (*e.g.* using CNNs) that implicitly model spatial relations and abstractions. However, the problem of finding appropriate feature representations has been replaced by the problem of designing an optimal network architecture for a given computer vision task. This in turn amounts to combining a set of pre-defined computational layers in an optimal way in terms of performance, sometimes respecting a certain number-of-parameters budget. In that sense, studying mid-level representations remains relevant, as it can provide useful intuition for improving CNN design. Moreover, methods such as the ones studied in this thesis can find application in settings where the lack of specialized hardware like GPUs renders the use of deep learning computationally prohibitive.

Bibliography

[Achanta 2012] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua and Sabine Süsstrunk. *SLIC Superpixels Compared to State-of-the-art Superpixel Methods*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 34, no. 11, pages 2274–2282, 2012. (Cited on page 2.)

[Alchatzidis 2014] S. Alchatzidis, A. Sotiras and N. Paragios. *Discrete Multi Atlas Segmentation using Agreement Constraints*. In BMVC, 2014. (Cited on page 101.)

[Amit 1999] Yali Amit and Donald Geman. *A computational model for visual selection*. Neural computation, vol. 11, no. 7, pages 1691–1715, 1999. (Cited on page 61.)

[Amit 2004] Yali Amit, Donald Geman and Xiaodong Fan. *A coarse-to-fine strategy for multiclass shape detection*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26, no. 12, pages 1606–1621, 2004. (Cited on page 61.)

[Andreetto 2012] Marco Andreetto, Lihi Zelnik-Manor and Pietro Perona. *Unsupervised learning of categorical segments in image collections*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 34, no. 9, pages 1842–1855, 2012. (Cited on page 61.)

[Apostoloff 2005] Nicholas Apostoloff and Andrew Fitzgibbon. *Learning spatiotemporal T-junctions for occlusion detection*. In IEEE Conference on Computer Vision and Pattern Recognition, volume 2, pages 553–559. IEEE, 2005. (Cited on page 12.)

[Arandjelović 2012] Relja Arandjelović and Andrew Zisserman. *Three things everyone should know to improve object retrieval*. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 2911–2918. IEEE, 2012. (Cited on page 2.)

[Arbelaez 2011] P. Arbelaez, M. Maire, C. Fowlkes and J. Malik. *Contour detection and hierarchical image segmentation*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2011. (Cited on pages 2, 12, 30, 31, 33 and 82.)

[Arbeláez 2012] Pablo Arbeláez, Bharath Hariharan, Chunhui Gu, Saurabh Gupta, Lubomir Bourdev and Jitendra Malik. *Semantic segmentation using regions and parts*. In CVPR, pages 3378–3385. IEEE, 2012. (Cited on page 82.)

[Azizpour 2012] Hossein Azizpour and Ivan Laptev. *Object detection using strongly-supervised deformable part models*. In ECCV, 2012. (Cited on pages 53 and 55.)

[Babenko 2008] B. Babenko, P. Dollár, Z. Tu, S. Belongie*et al. Simultaneous learning and alignment : Multi-instance and multi-pose learning*. 2008. (Cited on page 33.)

[Barlow 1979] HB Barlow and BC Reeves. *The versatility and absolute efficiency of detecting mirror symmetry in random dot displays.* Vision research, vol. 19, no. 7, pages 783–793, 1979. (Cited on page 10.)

[Blum 1967] H. Blum. *A transformation for extracting new descriptors of shape.* Models for the perception of speech and visual form, 1967. (Cited on pages 10 and 13.)

[Blum 1973] H. Blum. *Biological shape and visual science (Part I).* Journal of theoretical Biology, 1973. (Cited on pages 10 and 13.)

[Bo 2011] Yihang Bo and Charless C Fowlkes. *Shape-based pedestrian parsing.* In CVPR, pages 2265–2272. IEEE, 2011. (Cited on pages ix, 81, 88 and 89.)

[Bosch 2007] Anna Bosch, Andrew Zisserman and Xavier Munoz. *Image classification using random forests and ferns.* In IEEE 11th International Conference on Computer Vision, pages 1–8. IEEE, 2007. (Cited on page 38.)

[Bourdev 2005] Lubomir Bourdev and Jonathan Brandt. *Robust object detection via soft cascade.* In Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, volume 2, pages 236–243. IEEE, 2005. (Cited on page 57.)

[Bourdev 2009a] L. Bourdev and J. Malik. *Poselets : Body Part Detectors Trained Using 3D Human Pose Annotations.* In ICCV, 2009. (Cited on page 59.)

[Bourdev 2009b] Lubomir Bourdev and Jitendra Malik. *Poselets : Body part detectors trained using 3d human pose annotations.* In Computer Vision, 2009 IEEE 12th International Conference on, pages 1365–1372. IEEE, 2009. (Cited on pages 53 and 82.)

[Boykov 2001] Yuri Boykov, Olga Veksler and Ramin Zabih. *Fast approximate energy minimization via graph cuts.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 23, no. 11, pages 1222–1239, 2001. (Cited on pages 99 and 100.)

[Brady 1984] M. Brady and H. Asada. *Smoothed local symmetries and their implementation.* IJRR, 1984. (Cited on page 14.)

[Breiman 1984] Leo Breiman, Jerome Friedman, Charles J Stone and Richard A Olshen. Classification and regression trees. CRC press, 1984. (Cited on page 38.)

[Burl 1998] Michael C Burl, Markus Weber and Pietro Perona. *A probabilistic approach to object recognition using local photometry and global geometry.* In Computer Vision—ECCV'98, pages 628–641. Springer, 1998. (Cited on page 54.)

[Canny 1986] J. Canny. *A computational approach to edge detection.* PAMI, 1986. (Cited on page 1.)

[Carreira 2012] Joao Carreira and Cristian Sminchisescu. *Cpmc : Automatic object segmentation using constrained parametric min-cuts.* Pattern Analysis and

Machine Intelligence, IEEE Transactions on, vol. 34, no. 7, pages 1312–1328, 2012. (Cited on page 2.)

[Caruana 2008] Rich Caruana, Nikos Karampatziakis and Ainur Yessenalina. *An empirical evaluation of supervised learning in high dimensions.* In Proceedings of the 25th international conference on Machine learning, pages 96–103. ACM, 2008. (Cited on page 38.)

[Catanzaro 2009] B. Catanzaro, B.Y. Su, N. Sundaram, Y. Lee, M. Murphy and K. Keutzer. *Efficient, high-quality image contour detection.* ICCV, 2009. (Cited on page 28.)

[Chai 2012] Yuning Chai, Esa Rahtu, Victor Lempitsky, Luc Van Gool and Andrew Zisserman. *Tricos : A tri-level class-discriminative co-segmentation method for image classification.* In Computer Vision–ECCV 2012, pages 794–807. Springer, 2012. (Cited on page 110.)

[Chandra 2015] Siddhartha Chandra, Stavros Tsogkas and Iasonas Kokkinos. *Accurate Human-Limb Segmentation in RGB-D images for Intelligent Mobility Assistance Robots.* Third Workshop on Assistive Computer Vision and Robotics (in conjunction with ICCV), 2015. (Cited on page 105.)

[Chatfield 2014] Ken Chatfield, Karen Simonyan, Andrea Vedaldi and Andrew Zisserman. *Return of the devil in the details : Delving deep into convolutional nets.* arXiv preprint arXiv :1405.3531, 2014. (Cited on page 80.)

[Chen 2014a] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy and Alan L Yuille. *Semantic image segmentation with deep convolutional nets and fully connected crfs.* arXiv preprint arXiv :1412.7062, 2014. (Cited on pages 78, 79, 80 and 84.)

[Chen 2014b] Xianjie Chen, Roozbeh Mottaghi, Xiaobai Liu, Sanja Fidler, Raquel Urtasun and Alan Yuille. *Detect what you can : Detecting and representing objects using holistic models and body parts.* In Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on, 2014. (Cited on pages 53, 82, 88 and 91.)

[Chudasama 2006] Y. Chudasama and T.W. Robbins. *Functions of frontostriatal systems in cognition : Comparative neuropsychopharmacological studies in rats, monkeys and humans.* 2006. (Cited on page 98.)

[Cimpoi 2014] Mircea Cimpoi, Subhransu Maji and Andrea Vedaldi. *Deep convolutional filter banks for texture recognition and segmentation.* arXiv preprint arXiv :1411.6836, 2014. (Cited on page 79.)

[Cootes 2001] Timothy F Cootes, Gareth J Edwards and Christopher J Taylor. *Active appearance models.* IEEE Transactions on Pattern Analysis & Machine Intelligence, no. 6, pages 681–685, 2001. (Cited on page 81.)

[Cour 2005] T. Cour, F. Benezit and J. Shi. *Spectral segmentation with multiscale graph decomposition.* CVPR, 2005. (Cited on page 30.)

[Criminisi 2011] Antonio Criminisi, Jamie Shotton, Duncan Robertson and Ender Konukoglu. *Regression forests for efficient anatomy detection and localization in CT studies.* In Medical Computer Vision. Recognition Techniques and Applications in Medical Imaging, pages 106–117. Springer, 2011. (Cited on page 38.)

[Dalal 2005] Navneet Dalal and Bill Triggs. *Histograms of oriented gradients for human detection.* In Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, volume 1, pages 886–893. IEEE, 2005. (Cited on pages 2, 6, 53, 54, 55 and 56.)

[Dean 2013] Thomas Dean, Mark Ruzon, Michael Segal, Jonathon Shlens, Sudheendra Vijayanarasimhan, Jay Yagnik*et al. Fast, accurate detection of 100,000 object classes on a single machine.* In Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on, pages 1814–1821. IEEE, 2013. (Cited on page 58.)

[Demirci 2009] M.F. Demirci, A. Shokoufandeh and S.J. Dickinson. *Skeletal shape abstraction from examples.* PAMI, 2009. (Cited on page 14.)

[Deng 2009] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li and Li Fei-Fei. *Imagenet : A large-scale hierarchical image database.* In Proc. CVPR, pages 248–255. IEEE, 2009. (Cited on pages 1 and 79.)

[Deng 2011] Jia Deng, Sanjeev Satheesh, Alexander C Berg and Fei Li. *Fast and balanced : Efficient label tree learning for large scale object recognition.* In Advances in Neural Information Processing Systems, pages 567–575, 2011. (Cited on page 61.)

[Dietterich 1997] Thomas G Dietterich, Richard H Lathrop and Tomás Lozano-Pérez. *Solving the multiple instance problem with axis-parallel rectangles.* Artificial intelligence, vol. 89, no. 1, pages 31–71, 1997. (Cited on page 12.)

[Dollár 2009] Piotr Dollár, Zhuowen Tu, Pietro Perona and Serge Belongie. *Integral Channel Features.* In BMVC, volume 2, page 5, 2009. (Cited on page 40.)

[Dollár 2013] Piotr Dollár and C Lawrence Zitnick. *Structured forests for fast edge detection.* In Computer Vision (ICCV), 2013 IEEE International Conference on, pages 1841–1848. IEEE, 2013. (Cited on pages 33, 38, 40 and 45.)

[Dubout 2012] Charles Dubout and François Fleuret. *Exact acceleration of linear object detectors.* In Proc. ECCV, pages 301–311. Springer, 2012. (Cited on pages 6, 54, 71 and 108.)

[Eberly 1994] D. Eberly, R. Gardner, B. Morse, S. Pizer and C. Scharlach. *Ridges for image analysis.* JMIV, 1994. (Cited on page 15.)

[Eslami 2014] SM Ali Eslami, Nicolas Heess, Christopher KI Williams and John Winn. *The shape boltzmann machine : a strong model of object shape.* ijcv, vol. 107, no. 2, pages 155–176, 2014. (Cited on page 83.)

[Everingham 2010] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn and Andrew Zisserman. *The pascal visual object classes (voc)* *challenge.* International journal of computer vision, vol. 88, no. 2, pages 303–338, 2010. (Cited on pages 1 and 57.)

[Farabet 2013] Clement Farabet, Camille Couprie, Laurent Najman and Yann LeCun. *Learning hierarchical features for scene labeling.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 35, no. 8, pages 1915–1929, 2013. (Cited on page 79.)

[Felzenszwalb 2004] Pedro F Felzenszwalb and Daniel P Huttenlocher. *Efficient graph-based image segmentation.* International Journal of Computer Vision, vol. 59, no. 2, pages 167–181, 2004. (Cited on page 2.)

[Felzenszwalb 2005] Pedro F Felzenszwalb and Daniel P Huttenlocher. *Pictorial structures for object recognition.* IJCV, vol. 61, no. 1, pages 55–79, 2005. (Cited on page 54.)

[Felzenszwalb 2006] Pedro Felzenszwalb and David McAllester. *A min-cover approach for finding salient curves.* In Computer Vision and Pattern Recognition Workshop, 2006. CVPRW'06. Conference on, pages 185–185. IEEE, 2006. (Cited on page 45.)

[Felzenszwalb 2008] Pedro Felzenszwalb, David McAllester and Deva Ramanan. *A discriminatively trained, multiscale, deformable part model.* In Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on, pages 1–8. IEEE, 2008. (Cited on pages 2 and 53.)

[Felzenszwalb 2010a] Pedro F Felzenszwalb, Ross B Girshick and David McAllester. *Cascade object detection with deformable part models.* In Proc. CVPR, pages 2241–2248. IEEE, 2010. (Cited on pages 54, 57 and 64.)

[Felzenszwalb 2010b] Pedro F Felzenszwalb, Ross B Girshick, David McAllester and Deva Ramanan. *Object detection with discriminatively trained part-based models.* IEEE Trans. PAMI, vol. 32, no. 9, pages 1627–1645, 2010. (Cited on pages 2, 53, 54, 55, 56, 73 and 81.)

[Fergus 2001] R Fergus, M Weber and P Perona. *Efficient methods for object recognition using the constellation model.* California Inst. Technol., Tech. Rep, 2001. (Cited on page 54.)

[Fidler 2010] Sanja Fidler, Marko Boben and Aleš Leonardis. *A coarse-to-fine taxonomy of constellations for fast multi-class object detection.* In Computer Vision–ECCV 2010, pages 687–700. Springer, 2010. (Cited on page 4.)

[Fischl 2002] B. Fischl, D. H. Salat, E. Busa, M. Albert, M. Dieterich, C. Haselgrove, A. Van Der Kouwe, R. Killiany, D. Kennedy, S. Klavenesset *et al.* *Whole brain segmentation : automated labeling of neuroanatomical structures in the human brain.* Neuron, vol. 33, no. 3, pages 341–355, 2002. (Cited on page 102.)

[Fischler 1973] Martin A Fischler and Robert A Elschlager. *The representation and matching of pictorial structures.* IEEE Transactions on Computers, vol. 100, no. 1, pages 67–92, 1973. (Cited on page 54.)

[Fleuret 2001] Francois Fleuret and Donald Geman. *Coarse-to-fine face detection*. International Journal of computer vision, vol. 41, no. 1-2, pages 85–107, 2001. (Cited on page 57.)

[Florack 1992] Luc MJ Florack, Bart M ter Haar Romeny, Jan J Koenderink and Max A Viergever. *Scale and the differential structure of images*. Image and Vision Computing, vol. 10, no. 6, pages 376–388, 1992. (Cited on page 15.)

[Foulkes 2003] C Foulkes, D Martin and J Malik. *Learning affinity functions for image segmentation*. In Proceedings IEEE Conference CVPR, volume 32, pages 8092–8096, 2003. (Cited on page 30.)

[Fowlkes 2004] Charless Christopher Fowlkes and Jitendra Malik. How much does globalization help segmentation ? Computer Science Division, University of California, 2004. (Cited on page 30.)

[Freund 1997] Yoav Freund and Robert E Schapire. *A decision-theoretic generalization of on-line learning and an application to boosting*. Journal of computer and system sciences, vol. 55, no. 1, pages 119–139, 1997. (Cited on page 38.)

[Gall 2013] Juergen Gall and Victor Lempitsky. *Class-specific hough forests for object detection*. In Decision forests for computer vision and medical image analysis, pages 143–157. Springer, 2013. (Cited on page 38.)

[Gangaputra 2006] Sachin Gangaputra and Donald Geman. *A design principle for coarse-to-fine classification*. In Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, volume 2, pages 1877–1884. IEEE, 2006. (Cited on page 57.)

[Ghodrati 2015] Amir Ghodrati, Ali Diba, Marco Pedersoli, Tinne Tuytelaars and Luc Van Gool. *DeepProposal : Hunting Objects by Cascading Deep Convolutional Layers*. International Conference on Computer Vision (ICCV), 2015. (Cited on page 4.)

[Gilbert 1992] Jean Charles Gilbert and Jorge Nocedal. *Global convergence properties of conjugate gradient methods for optimization*. SIAM Journal on optimization, vol. 2, no. 1, pages 21–42, 1992. (Cited on page 33.)

[Girshick] R. B. Girshick, P. F. Felzenszwalb and D. McAllester. *Discriminatively Trained Deformable Part Models, Release 5*. http ://people.cs.uchicago.edu/~rbg/latent-release5/. (Cited on pages vii, 53, 58 and 60.)

[Girshick 2011a] Ross Girshick, Jamie Shotton, Pushmeet Kohli, Antonio Criminisi and Andrew Fitzgibbon. *Efficient regression of general-activity human poses from depth images*. In Computer Vision (ICCV), 2011 IEEE International Conference on, pages 415–422. IEEE, 2011. (Cited on page 38.)

[Girshick 2011b] Ross B Girshick, Pedro F Felzenszwalb and David A Mcallester. *Object detection with grammar models*. In Advances in Neural Information Processing Systems, pages 442–450, 2011. (Cited on page 54.)

[Girshick 2013] Ross Girshick and Jitendra Malik. *Training Deformable Part Models with Decorrelated Features.* In Proceedings of the International Conference on Computer Vision (ICCV), 2013. (Cited on pages 6, 54, 58, 71 and 108.)

[Girshick 2014a] Ross Girshick, Jeff Donahue, Trevor Darrell and Jagannath Malik. *Rich feature hierarchies for accurate object detection and semantic segmentation.* In Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on, pages 580–587. IEEE, 2014. (Cited on pages 1, 53, 70, 71 and 79.)

[Girshick 2014b] Ross Girshick, Forrest Iandola, Trevor Darrell and Jitendra Malik. *Deformable part models are convolutional neural networks.* arXiv preprint arXiv :1409.5403, 2014. (Cited on pages 51, 71 and 75.)

[Girshick 2015] Ross Girshick. *Fast R-CNN.* arXiv preprint arXiv :1504.08083, 2015. (Cited on pages 70 and 79.)

[Gray 2003] Alexander G Gray and Andrew W Moore. *Nonparametric Density Estimation : Toward Computational Tractability.* In SDM, pages 203–211. SIAM, 2003. (Cited on page 58.)

[Gu 2009] Chunhui Gu, Joseph J Lim, Pablo Arbeláez and Jitendra Malik. *Recognition using regions.* In CVPR, pages 1030–1037, 2009. (Cited on page 2.)

[Hallman 2015] Sam Hallman and Charless C Fowlkes. *Oriented Edge Forests for Boundary Detection.* In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1732–1740, 2015. (Cited on page 38.)

[Haralick 1983] R.M. Haralick. *Ridges and valleys on digital images.* CVGIP, 1983. (Cited on page 15.)

[Hariharan 2011] Bharath Hariharan, Pablo Arbeláez, Lubomir Bourdev, Subhransu Maji and Jitendra Malik. *Semantic contours from inverse detectors.* In Computer Vision (ICCV), 2011 IEEE International Conference on, pages 991–998. IEEE, 2011. (Cited on page 82.)

[He 2004] Xuming He, Richard S Zemel and Miguel Á Carreira-Perpiñán. *Multiscale conditional random fields for image labeling.* In Computer vision and pattern recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE computer society conference on, volume 2, pages II–695. IEEE, 2004. (Cited on page 85.)

[He 2014] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun. *Spatial pyramid pooling in deep convolutional networks for visual recognition.* In ECCV, pages 346–361. Springer, 2014. (Cited on page 70.)

[He 2015] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun. *Delving deep into rectifiers : Surpassing human-level performance on imagenet classification.* 2015. (Cited on page 1.)

[Hinton 1984] Geoffrey E Hinton. *Distributed representations.* 1984. (Cited on page 83.)

[Hinton 2006] Geoffrey E Hinton and Ruslan R Salakhutdinov. *Reducing the dimensionality of data with neural networks.* Science, vol. 313, no. 5786, pages 504–507, 2006. (Cited on page 83.)

[Hinton 2010] Geoffrey Hinton. *A practical guide to training restricted Boltzmann machines.* Momentum, vol. 9, no. 1, page 926, 2010. (Cited on page 87.)

[Hoiem 2007] Derek Hoiem, Carsten Rother and John Winn. *3d layoutcrf for multi-view object class recognition and segmentation.* In CVPR, pages 1–8. IEEE, 2007. (Cited on page 82.)

[Huang 2007a] Gary B Huang, Vidit Jain and Erik Learned-Miller. *Unsupervised joint alignment of complex images.* In Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on, 2007. (Cited on page 88.)

[Huang 2007b] Gary B Huang, Manu Ramesh, Tamara Berg and Erik Learned-Miller. *Labeled faces in the wild : A database for studying face recognition in unconstrained environments.* Rapport technique, Technical Report 07-49, University of Massachusetts, Amherst, 2007. (Cited on page 90.)

[Iandola 2014] Forrest Iandola, Matt Moskewicz, Sergey Karayev, Ross Girshick, Trevor Darrell and Kurt Keutzer. *Densenet : Implementing efficient convnet descriptor pyramids.* arXiv preprint arXiv :1404.1869, 2014. (Cited on page 71.)

[Jaakkola 1999] Tommi Jaakkola, David Hausssler*et al.* *Exploiting generative models in discriminative classifiers.* Advances in neural information processing systems, pages 487–493, 1999. (Cited on page 2.)

[Jegou 2008] Herve Jegou, Matthijs Douze and Cordelia Schmid. *Hamming embedding and weak geometric consistency for large scale image search.* In Computer Vision–ECCV 2008, pages 304–317. Springer, 2008. (Cited on page 2.)

[Jégou 2010] Hervé Jégou, Matthijs Douze, Cordelia Schmid and Patrick Pérez. *Aggregating local descriptors into a compact image representation.* In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 3304–3311. IEEE, 2010. (Cited on page 2.)

[Ji 2013] Shuiwang Ji, Wei Xu, Ming Yang and Kai Yu. *3D convolutional neural networks for human action recognition.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 35, no. 1, pages 221–231, 2013. (Cited on page 110.)

[Jin 2006] Ya Jin and Stuart Geman. *Context and hierarchy in a probabilistic image model.* In Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, volume 2, pages 2145–2152. IEEE, 2006. (Cited on page 82.)

[Jurie 2005] Frederic Jurie and Bill Triggs. *Creating efficient codebooks for visual recognition.* In Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on, volume 1, pages 604–610. IEEE, 2005. (Cited on page 2.)

[Kae 2013] Andrew Kae, Kihyuk Sohn, Honglak Lee and Erik Learned-Miller. *Augmenting CRFs with Boltzmann machine shape priors for image labeling.* In Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on, pages 2019–2026. IEEE, 2013. (Cited on pages 83, 84, 86, 88, 89, 90 and 94.)

[Keeler 1990] J.D. Keeler, D.E. Rumelhart and W.K. Leow. *Integrated segmentation and recognition of hand-printed numerals.* In NIPS, 1990. (Cited on page 12.)

[Kirk 2007] David Kirk*et al. NVIDIA CUDA software and GPU parallel computing architecture.* In ISMM, volume 7, pages 103–104, 2007. (Cited on page 1.)

[Koenderink 1984] Jan J Koenderink. *The structure of images.* Biological cybernetics, vol. 50, no. 5, pages 363–370, 1984. (Cited on page 15.)

[Kokkinos 2006] I. Kokkinos, P. Maragos and A. Yuille. *Bottom-up & top-down object detection using primal sketch features and graphical models.* In CVPR, volume 2, pages 1893–1900. IEEE, 2006. (Cited on pages vi, 11, 35 and 36.)

[Kokkinos 2010] I. Kokkinos. *Highly accurate boundary detection and grouping.* CVPR, 2010. (Cited on page 45.)

[Kokkinos 2011a] Iasonas Kokkinos. *Rapid deformable object detection using dual-tree branch-and-bound.* In Proc. NIPS, pages 2681–2689, 2011. (Cited on page 58.)

[Kokkinos 2011b] Iasonas Kokkinos and Alan Yuille. *Inference and learning with hierarchical shape models.* International Journal of Computer Vision, vol. 93, no. 2, pages 201–225, 2011. (Cited on page 82.)

[Kokkinos 2012] Iasonas Kokkinos. *Bounding part scores for rapid detection with deformable part models.* In Computer Vision–ECCV 2012. Workshops and Demonstrations, pages 41–50. Springer, 2012. (Cited on page 58.)

[Kokkinos 2013] Iasonas Kokkinos. *Shufflets : shared mid-level parts for fast object detection.* In Proc. ICCV, pages 1393–1400. IEEE, 2013. (Cited on pages 2, 57, 61, 62 and 63.)

[Konishi 2003] Scott Konishi, Alan L Yuille, James M Coughlan and Song Chun Zhu. *Statistical edge detection : Learning and evaluating edge cues.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 25, no. 1, pages 57–74, 2003. (Cited on page 12.)

[Kootstra 2008] Gert Kootstra, Arco Nederveen and Bart De Boer. *Paying attention to symmetry.* In British Machine Vision Conference (BMVC2008), pages 1115–1125. The British Machine Vision Association and Society for Pattern Recognition, 2008. (Cited on page 10.)

[Krähenbühl 2011] Philipp Krähenbühl and Vladlen Koltun. *Efficient Inference in Fully Connected CRFs with Gaussian Edge Potentials.* In Advances in Neural Information Processing Systems, pages 109–117, 2011. (Cited on page 84.)

[Krizhevsky 2012] Alex Krizhevsky, Ilya Sutskever and Geoffrey E Hinton. *Imagenet classification with deep convolutional neural networks.* In Advances in neural information processing systems, pages 1097–1105, 2012. (Cited on pages 1, 6, 53, 68, 69, 79, 80 and 99.)

[Kumar 2009] M Pawan Kumar, Andrew Zisserman and Philip HS Torr. *Efficient discriminative learning of parts-based models.* In Computer Vision, 2009 IEEE 12th International Conference on, pages 552–559. IEEE, 2009. (Cited on page 53.)

[Lafferty 2001] John Lafferty, Andrew McCallum and Fernando CN Pereira. *Conditional random fields : Probabilistic models for segmenting and labeling sequence data.* 2001. (Cited on page 85.)

[Larochelle 2008] Hugo Larochelle and Yoshua Bengio. *Classification using discriminative restricted Boltzmann machines.* In Proceedings of the 25th international conference on Machine learning, pages 536–543. ACM, 2008. (Cited on page 83.)

[Lazebnik 2006] Svetlana Lazebnik, Cordelia Schmid and Jean Ponce. *Beyond bags of features : Spatial pyramid matching for recognizing natural scene categories.* In Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, volume 2, pages 2169–2178. IEEE, 2006. (Cited on pages 2 and 82.)

[LeCun 1998] Yann LeCun, Léon Bottou, Yoshua Bengio and Patrick Haffner. *Gradient-based learning applied to document recognition.* Proceedings of the IEEE, vol. 86, no. 11, pages 2278–2324, 1998. (Cited on pages 1, 4, 68 and 78.)

[Lee 2001] Ann B Lee, David Mumford and Jinggang Huang. *Occlusion models for natural images : A statistical study of a scale-invariant dead leaves model.* International Journal of Computer Vision, vol. 41, no. 1-2, pages 35–59, 2001. (Cited on page 41.)

[Lepetit 2006] Vincent Lepetit and Pascal Fua. *Keypoint recognition using randomized trees.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 28, no. 9, pages 1465–1479, 2006. (Cited on page 38.)

[Leung 1998] Thomas Leung and Jitendra Malik. *Contour continuity in region based image segmentation.* In Computer Vision—ECCV'98, pages 544–559. Springer, 1998. (Cited on page 30.)

[Levinshtein 2009] A. Levinshtein, S. Dickinson and C. Sminchisescu. *Multiscale symmetric part detection and grouping.* ICCV, 2009. (Cited on pages vi, 12, 15, 35, 36 and 37.)

[Li 2010] Li-Jia Li, Hao Su, Li Fei-Fei and Eric P Xing. *Object bank : A high-level image representation for scene classification & semantic feature sparsification.* In Advances in neural information processing systems, pages 1378–1386, 2010. (Cited on page 2.)

[Lim 2013] Jasmine J Lim, C Lawrence Zitnick and Piotr Dollár. *Sketch tokens : A learned mid-level representation for contour and object detection.* In Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on, pages 3158–3165. IEEE, 2013. (Cited on pages 12, 33, 38 and 40.)

[Lin 2014] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár and C Lawrence Zitnick. *Microsoft COCO : Common objects in context.* In European Conference on Computer Vision (ECCV), pages 740–755. Springer, 2014. (Cited on page 1.)

[Lindeberg 1990] Tony Lindeberg. *Scale-space for discrete signals.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 12, no. 3, pages 234–254, 1990. (Cited on page 15.)

[Lindeberg 1998] T. Lindeberg. *Edge detection and ridge detection with automatic scale selection.* IJCV, 1998. (Cited on pages v, 12, 15, 16, 37 and 44.)

[Lindholm 2008] Erik Lindholm, John Nickolls, Stuart Oberman and John Montrym. *NVIDIA Tesla : A unified graphics and computing architecture.* IEEE micro, no. 2, pages 39–55, 2008. (Cited on page 1.)

[Liu 2009] Y. Liu. Computational symmetry in computer vision and computer graphics. Now publishers Inc, 2009. (Cited on pages 10 and 13.)

[Locher 1989] P Locher and C Nodine. *The perceptual value of symmetry.* Computers & mathematics with applications, vol. 17, no. 4, pages 475–484, 1989. (Cited on page 10.)

[Long 2014] Jonathan Long, Evan Shelhamer and Trevor Darrell. *Fully convolutional networks for semantic segmentation.* arXiv preprint arXiv :1411.4038, 2014. (Cited on pages 77, 79, 80 and 84.)

[Lôpez 1999] A.M. Lôpez, F. Lumbreras, J. Serrat and J.J. Villanueva. *Evaluation of methods for ridge and valley detection.* PAMI, 1999. (Cited on pages 12 and 15.)

[Lowe 2004] D.G. Lowe. *Distinctive image features from scale-invariant keypoints.* IJCV, 2004. (Cited on page 2.)

[Lu 2014] Wenhao Lu, Xiaochen Lian and Alan Yuille. *Parsing Semantic Parts of Cars Using Graphical Models and Segment Appearance Consistency.* BMVC, 2014. (Cited on pages 82, 91 and 109.)

[Luo 2013] Ping Luo, Xiaogang Wang and Xiaoou Tang. *Pedestrian parsing via deep decompositional network.* In Computer Vision (ICCV), 2013 IEEE International Conference on, 2013. (Cited on page 89.)

[Maire 2008] Michael Maire, Pablo Arbeláez, Charless Fowlkes and Jitendra Malik. *Using contours to detect and localize junctions in natural images.* In Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on, pages 1–8. IEEE, 2008. (Cited on page 12.)

[Maire 2011] Michael Maire, Stella X Yu and Pietro Perona. *Object detection and segmentation from joint embedding of parts and pixels.* In ICCV, pages 2142–2149, 2011. (Cited on page 82.)

[Maji 2009] Subhransu Maji and Alexander C Berg. *Max-margin additive classifiers for detection.* In Computer Vision, 2009 IEEE 12th International Conference on, pages 40–47. IEEE, 2009. (Cited on page 2.)

[Maji 2011] S. Maji, N.K. Vishnoi and J. Malik. *Biased normalized cuts.* CVPR, 2011. (Cited on page 30.)

[Mallat 1999] Stéphane Mallat. A wavelet tour of signal processing. Academic press, 1999. (Cited on page 80.)

[Martin 2001] D. Martin, C. Fowlkes, D. Tal and J. Malik. *A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics.* ICCV, 2001. (Cited on pages 12, 17, 42 and 106.)

[Martin 2004] D.R. Martin, C.C. Fowlkes and J. Malik. *Learning to detect natural image boundaries using local brightness, color, and texture cues.* PAMI, 2004. (Cited on pages 2, 12, 28, 29 and 33.)

[Matzen 2013] Kevin Matzen and Noah Snavely. *NYC3DCars : A Dataset of 3D Vehicles in Geographic Context.* In ICCV, 2013. (Cited on page 59.)

[Mitzenmacher 2005] Michael Mitzenmacher and Eli Upfal. Probability and computing : Randomized algorithms and probabilistic analysis. Cambridge University Press, 2005. (Cited on page 62.)

[Mnih 2012] Volodymyr Mnih, Hugo Larochelle and Geoffrey E Hinton. *Conditional restricted boltzmann machines for structured output prediction.* arXiv preprint arXiv :1202.3748, 2012. (Cited on pages 83 and 86.)

[Moosmann 2007] Frank Moosmann, Bill Triggs and Frederic Jurie. *Fast discriminative visual codebooks using randomized clustering forests.* In Twentieth Annual Conference on Neural Information Processing Systems (NIPS'06), pages 985–992. MIT Press, 2007. (Cited on page 2.)

[Murphy 2012] Kevin P Murphy. Machine learning : a probabilistic perspective. MIT press, 2012. (Cited on pages 85 and 87.)

[Ng 2002] A.Y. Ng, M.I. Jordan and Y. Weiss. *On spectral clustering : Analysis and an algorithm.* NIPS, 2002. (Cited on page 30.)

[Nickolls 2010] John Nickolls and William J Dally. *The GPU computing era.* IEEE micro, no. 2, pages 56–69, 2010. (Cited on page 1.)

[Nilsback 2008] Maria-Elena Nilsback and Andrew Zisserman. *Automated flower classification over a large number of classes.* In Computer Vision, Graphics & Image Processing, 2008. ICVGIP'08. Sixth Indian Conference on, pages 722–729. IEEE, 2008. (Cited on page 110.)

[Oliva 2001] Aude Oliva and Antonio Torralba. *Modeling the shape of the scene : A holistic representation of the spatial envelope.* International journal of computer vision, vol. 42, no. 3, pages 145–175, 2001. (Cited on page 2.)

[Oliva 2006] Aude Oliva and Antonio Torralba. *Building the gist of a scene : The role of global image features in recognition.* Progress in brain research, vol. 155, pages 23–36, 2006. (Cited on page 2.)

[Oquab 2014] Maxime Oquab, Léon Bottou, Ivan Laptev and Josef Sivic. *Weakly supervised object recognition with convolutional neural networks.* 2014. (Cited on page 84.)

[Ott 2011] Patrick Ott and Mark Everingham. *Shared parts for deformable part-based models.* In CVPR, pages 1513–1520, 2011. (Cited on page 57.)

[Papandreou 2014] George Papandreou, Iasonas Kokkinos and Pierre-André Savalle. *Untangling local and global deformations in deep convolutional networks for image classification and sliding window detection.* arXiv preprint arXiv :1412.0296, 2014. (Cited on page 70.)

[Parkhi 2012] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman and CV Jawahar. *Cats and dogs.* In Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, pages 3498–3505. IEEE, 2012. (Cited on page 110.)

[Patenaude 2011] B. Patenaude, S. M. Smith, Kennedy D. N. and M. Jenkinson. *A Bayesian model of shape and appearance for subcortical brain segmentation.* NeuroImage, 2011. (Cited on page 102.)

[Perronnin 2007] Florent Perronnin and Christopher Dance. *Fisher kernels on visual vocabularies for image categorization.* In Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on, pages 1–8. IEEE, 2007. (Cited on page 2.)

[Perronnin 2010] Florent Perronnin, Jorge Sánchez and Thomas Mensink. *Improving the fisher kernel for large-scale image classification.* In Computer Vision–ECCV 2010, pages 143–156. Springer, 2010. (Cited on page 2.)

[Philbin 2007] James Philbin, Ondřej Chum, Michael Isard, Josef Sivic and Andrew Zisserman. *Object retrieval with large vocabularies and fast spatial matching.* In IEEE Conference on Computer Vision and Pattern Recognition, pages 1–8. IEEE, 2007. (Cited on page 2.)

[Pizer 1994] Stephen M Pizer, Christina A Burbeck, James M Coggins, Daniel S Fritsch and Bryan S Morse. *Object shape before boundary shape : Scale-space medial axes.* Journal of Mathematical Imaging and Vision, vol. 4, no. 3, pages 303–313, 1994. (Cited on pages 12 and 15.)

[Pizer 1998] S.M. Pizer, D. Eberly, D.S. Fritsch and B.S. Morse. *Zoom-Invariant Vision of Figural Shape : The Mathematics of Cores* 1.* CVIU, 1998. (Cited on page 15.)

[Prasoon 2013] Adhish Prasoon, Kersten Petersen, Christian Igel, François Lauze, Erik Dam and Mads Nielsen. *Deep feature learning for knee cartilage segmentation using a triplanar convolutional neural network.* In Medical Image Computing and Computer-Assisted Intervention–MICCAI 2013, pages 246–253. Springer, 2013. (Cited on page 98.)

[Quattoni 2007] Ariadna Quattoni, Sybor Wang, Louis-Philippe Morency, Michael Collins and Trevor Darrell. *Hidden conditional random fields.* IEEE Transactions on Pattern Analysis & Machine Intelligence, no. 10, pages 1848–1852, 2007. (Cited on page 85.)

[Quinlan 1993] J Ross Quinlan. *C4. 5 : Programming for machine learning*. Morgan Kauffmann, 1993. (Cited on page 38.)

[Ren 2015] Shaoqing Ren, Kaiming He, Ross Girshick and Jian Sun. *Faster r-cnn : Towards real-time object detection with region proposal networks*. arXiv preprint arXiv :1506.01497, 2015. (Cited on pages ix, 4, 70, 79, 95 and 96.)

[Rohlfing 2012] T. Rohlfing. *Image Similarity and Tissue Overlaps as Surrogates for Image Registration Accuracy : Widely Used but Unreliable*. IEEE Transactions on Medical Imaging, 2012. (Cited on page 100.)

[Rosten 2006] Edward Rosten and Tom Drummond. *Machine learning for high-speed corner detection*. In Computer Vision–ECCV 2006, pages 430–443. Springer, 2006. (Cited on page 12.)

[Royer 1981] Fred L Royer. *Detection of symmetry*. Journal of Experimental Psychology : Human Perception and Performance, 1981. (Cited on page 10.)

[Rubner 2001] Y. Rubner, J. Puzicha, C. Tomasi and J.M. Buhmann. *Empirical evaluation of dissimilarity measures for color and texture*. CVIU, 2001. (Cited on page 28.)

[Rumelhart 1988] David E Rumelhart, Geoffrey E Hinton and Ronald J Williams. *Learning representations by back-propagating errors*. Cognitive modeling, vol. 5, page 3, 1988. (Cited on page 1.)

[Sadeghi 2014] Mohammad Amin Sadeghi and David Forsyth. *30hz object detection with dpm v5*. In Computer Vision–ECCV 2014, pages 65–79. Springer, 2014. (Cited on page 58.)

[Salakhutdinov 2007] Ruslan Salakhutdinov, Andriy Mnih and Geoffrey Hinton. *Restricted Boltzmann machines for collaborative filtering*. In Proceedings of the 24th international conference on Machine learning, pages 791–798. ACM, 2007. (Cited on page 84.)

[Salakhutdinov 2009] Ruslan Salakhutdinov and Geoffrey E Hinton. *Deep boltzmann machines*. In International Conference on Artificial Intelligence and Statistics, pages 448–455, 2009. (Cited on page 83.)

[Savalle 2014] Pierre-André Savalle, Stavros Tsogkas, George Papandreou and Iasonas Kokkinos. *Deformable part models with cnn features*. In European Conference on Computer Vision, Parts and Attributes Workshop, 2014. (Cited on pages 51, 71 and 105.)

[Schapire 1990] Robert E Schapire. *The strength of weak learnability*. Machine learning, vol. 5, no. 2, pages 197–227, 1990. (Cited on page 38.)

[Schwing 2015] Alexander G Schwing and Raquel Urtasun. *Fully connected deep structured networks*. arXiv preprint arXiv :1503.02351, 2015. (Cited on page 78.)

[Sebastian 2001] Thomas Sebastian, Philip Klein and Benjamin Kimia. *Recognition of shapes by editing shock graphs.* In null, page 755. IEEE, 2001. (Cited on page 14.)

[Sermanet 2014] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus and Yann LeCun. *Overfeat : Integrated recognition, localization and detection using convolutional networks.* In International Conference on Learning Representations (ICLR 2014), 2014. (Cited on pages 70 and 84.)

[Sharon 2006] Eitan Sharon, Meirav Galun, Dahlia Sharon, Ronen Basri and Achi Brandt. *Hierarchy and adaptivity in segmenting visual scenes.* Nature, vol. 442, no. 7104, pages 810–813, 2006. (Cited on page 82.)

[Shi 2000] J. Shi and J. Malik. *Normalized cuts and image segmentation.* PAMI, 2000. (Cited on page 30.)

[Shotton 2013] Jamie Shotton, Toby Sharp, Alex Kipman, Andrew Fitzgibbon, Mark Finocchio, Andrew Blake, Mat Cook and Richard Moore. *Real-time human pose recognition in parts from single depth images.* Communications of the ACM, vol. 56, no. 1, pages 116–124, 2013. (Cited on page 38.)

[Siddiqi 1999] Kaleem Siddiqi, Ali Shokoufandeh, Sven J Dickinson and Steven W Zucker. *Shock graphs and shape matching.* International Journal of Computer Vision, vol. 35, no. 1, pages 13–32, 1999. (Cited on page 14.)

[Siddiqi 2002] K. Siddiqi, S. Bouix, A. Tannenbaum and S.W. Zucker. *Hamilton-jacobi skeletons.* IJCV, 2002. (Cited on page 14.)

[Siddiqi 2008] Kaleem Siddiqi and Stephen Pizer. Medial representations : mathematics, algorithms and applications, volume 37. Springer Science & Business Media, 2008. (Cited on pages 10, 12 and 13.)

[Sie Ho Lee 2013] Tom Sie Ho Lee, Sanja Fidler and Sven Dickinson. *Detecting curved symmetric parts using a deformable disc model.* In Computer Vision (ICCV), 2013 IEEE International Conference on, pages 1753–1760. IEEE, 2013. (Cited on page 15.)

[Simonyan 2014] Karen Simonyan and Andrew Zisserman. *Very deep convolutional networks for large-scale image recognition.* arXiv preprint arXiv :1409.1556, 2014. (Cited on pages 70, 79, 80, 84, 99 and 108.)

[Singh 2012] Saurabh Singh, Abhinav Gupta and Alexei Efros. *Unsupervised discovery of mid-level discriminative patches.* Computer Vision–ECCV 2012, pages 73–86, 2012. (Cited on page 2.)

[Sivic 2003] Josef Sivic and Andrew Zisserman. *Video Google : A text retrieval approach to object matching in videos.* In IEEE Conference on Computer Vision and Pattern Recognition, pages 1470–1477. IEEE, 2003. (Cited on page 2.)

[Smolensky 1986] Paul Smolensky. *Information processing in dynamical systems : Foundations of harmony theory.* 1986. (Cited on page 83.)

[Šochman 2005] Jan Šochman and Jiří Matas. *Waldboost-learning for time constrained sequential detection.* In Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, volume 2, pages 150–156. IEEE, 2005. (Cited on page 57.)

[Soegaard 2010] Mads Soegaard. *Gestalt principles of form perception.* Interaction-Design. org, page 8, 2010. (Cited on page 10.)

[Stark 2009] M. Stark, M. Goesele and B. Schiele. *A shape-based object class model for knowledge transfer.* ICCV, 2009. (Cited on pages 2, 4 and 11.)

[Steger 1998] C. Steger. *An unbiased detector of curvilinear structures.* PAMI, 1998. (Cited on page 15.)

[Sun 2011] Min Sun and Silvio Savarese. *Articulated part-based model for joint object detection and pose estimation.* In ICCV, pages 723–730. IEEE, 2011. (Cited on page 57.)

[Szegedy 2014] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke and Andrew Rabinovich. *Going deeper with convolutions.* arXiv preprint arXiv :1409.4842, 2014. (Cited on pages 70 and 80.)

[Telea 2002] A. Telea and J.J. Van Wijk. *An augmented fast marching method for computing skeletons and centerlines.* Eurographics 2002, 2002. (Cited on pages 14 and 17.)

[Telea 2004] A. Telea, C. Sminchisescu and S. Dickinson. *Optimal inference for hierarchical skeleton abstraction.* Pattern Recognition, 2004. (Cited on page 14.)

[Teo 2015] Ching L. Teo, Cornelia Fermueller and Yannis Aloimonos. *Detection and segmentation of 2D curved reflection symmetric structures.* In Computer Vision (ICCV), 2011 IEEE International Conference on. IEEE, 2015. (Cited on pages 4, 10, 107 and 109.)

[Tompson 2014] Jonathan J Tompson, Arjun Jain, Yann LeCun and Christoph Bregler. *Joint training of a convolutional network and a graphical model for human pose estimation.* In Advances in Neural Information Processing Systems, pages 1799–1807, 2014. (Cited on page 78.)

[Trulls 2014] Eduard Trulls, Stavros Tsogkas, Iasonas Kokkinos, Alberto Sanfeliu and Francesc Moreno-Noguer. *Segmentation-aware deformable part models.* In Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on, pages 168–175. IEEE, 2014. (Cited on page 105.)

[Tsogkas 2012] Stavros Tsogkas and Iasonas Kokkinos. *Learning-Based symmetry detection in natural images.* In ECCV, pages 41–54. 2012. (Cited on pages 10 and 105.)

[Uijlings 2013] Jasper RR Uijlings, Koen EA van de Sande, Theo Gevers and Arnold WM Smeulders. *Selective search for object recognition.* IJCV, vol. 104, no. 2, pages 154–171, 2013. (Cited on pages 70 and 90.)

[Van De Sande 2010] Koen EA Van De Sande, Theo Gevers and Cees GM Snoek. *Evaluating color descriptors for object and scene recognition.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 32, no. 9, pages 1582–1596, 2010. (Cited on page 82.)

[van Eede 2006] M. van Eede, D. Macrini, A. Telea, C. Sminchisescu and SS Dickinson. *Canonical skeletons for shape matching.* ICPR, 2006. (Cited on page 14.)

[Vedaldi 2009] Andrea Vedaldi, Varun Gulshan, Manik Varma and Andrew Zisserman. *Multiple kernels for object detection.* In Computer Vision, 2009 IEEE 12th International Conference on, pages 606–613. IEEE, 2009. (Cited on page 53.)

[Vedaldi 2012] Andrea Vedaldi and Andrew Zisserman. *Sparse kernel approximations for efficient classification and detection.* In Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, pages 2320–2327. IEEE, 2012. (Cited on page 58.)

[Vedaldi 2014] Andrea Vedaldi, Siddarth Mahendran, Stavros Tsogkas, Subhrajyoti Maji, Ross Girshick, Juho Kannala, Esa Rahtu, Iasonas Kokkinos, Matthew B Blaschko, Daniel Weiss*et al. Understanding objects in detail with fine-grained attributes.* In Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on, pages 3622–3629. IEEE, 2014. (Cited on pages vii, 51, 52, 53, 75 and 105.)

[Vedaldi 2015] A. Vedaldi and K. Lenc. *MatConvNet – Convolutional Neural Networks for MATLAB.* 2015. (Cited on page 103.)

[Viola 2001] P. Viola and M. Jones. *Rapid object detection using a boosted cascade of simple features.* CVPR, 2001. (Cited on pages 2, 28 and 57.)

[Viola 2006] P. Viola, J. Platt and C. Zhang. *Multiple instance boosting for object detection.* NIPS, 2006. (Cited on page 12.)

[Wagemans 1998] Johan Wagemans. *Parallel visual processes in symmetry perception : Normality and pathology.* Documenta ophthalmologica, 1998. (Cited on page 10.)

[Wah 2011] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona and Serge Belongie. *The caltech-ucsd birds-200-2011 dataset.* 2011. (Cited on page 90.)

[Wan 2014] Li Wan, David Eigen and Rob Fergus. *End-to-End Integration of a Convolutional Network, Deformable Parts Model and Non-Maximum Suppression.* arXiv preprint arXiv :1411.5309, 2014. (Cited on page 78.)

[Wang 2007] Liming Wang, Jianbo Shi, Gang Song and I-fan Shen. *Object detection combining recognition and segmentation.* In ACCV. 2007. (Cited on pages 81, 88 and 89.)

[Wang 2015a] Jianyu Wang and Alan Yuille. *Semantic Part Segmentation using Compositional Model combining Shape and Appearance.* In Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on, 2015. (Cited on pages 82 and 109.)

[Wang 2015b] Peng Wang, Xiaohui Shen, Zhe Lin, Scott Cohen, Brian Price and Alan Yuille. *Joint Object and Part Segmentation using Deep Learned Potentials.* International Conference on Computer Computer Vision (ICCV), 2015. (Cited on pages 4, 83 and 109.)

[Weyl 1989] Hermann Weyl. Symmetry. 1952. 1989. (Cited on pages 10 and 17.)

[Winn 2006] John Winn and Jamie Shotton. *The layout consistent random field for recognizing and segmenting partially occluded objects.* In CVPR, volume 1, pages 37–44. IEEE, 2006. (Cited on page 82.)

[Witkin 1984] Andrew P Witkin. *Scale-space filtering : A new approach to multi-scale description.* In Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'84., volume 9, pages 150–153. IEEE, 1984. (Cited on page 15.)

[Xie 2015] Saining Xie and Zhuowen Tu. *Holistically-Nested Edge Detection.* arXiv preprint arXiv :1504.06375, 2015. (Cited on pages 47 and 109.)

[Yang 2013] Yi Yang and Deva Ramanan. *Articulated human detection with flexible mixtures of parts.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 35, no. 12, pages 2878–2890, 2013. (Cited on pages 53, 57 and 71.)

[Yang 2014] Jimei Yang, Simon Safar and Ming-Hsuan Yang. *Max-margin boltzmann machines for object segmentation.* In Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on, pages 320–327. IEEE, 2014. (Cited on page 83.)

[Yu 2015] Qian Yu, Yongxin Yang, Yi-Zhe Song, Tao Xiang and Timothy M Hospedales. *Sketch-a-net that beats humans.* In Proceedings of the British Machine Vision Conference (BMVC), pages 7–1, 2015. (Cited on pages 1, 79, 80 and 81.)

[Yuille 1986] Alan L Yuille and Tomaso A Poggio. *Scaling theorems for zero crossings.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, no. 1, pages 15–25, 1986. (Cited on page 15.)

[Zeiler 2010] Matthew D Zeiler, Dilip Krishnan, Graham W Taylor and Rob Fergus. *Deconvolutional networks.* In Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on, pages 2528–2535. IEEE, 2010. (Cited on page 4.)

[Zeiler 2014] Matthew D Zeiler and Rob Fergus. *Visualizing and understanding convolutional networks.* In Computer Vision–ECCV 2014, pages 818–833. Springer, 2014. (Cited on pages 4 and 68.)

[Zhang 2014] Ning Zhang, Jeff Donahue, Ross Girshick and Trevor Darrell. *Part-Based R-CNNs for Fine-Grained Category Detection.* In ECCV, pages 834–849. Springer, 2014. (Cited on pages 78 and 90.)

[Zhang 2015] Zheng Zhang, Wei Shen, Cong Yao and Xiang Bai. *Symmetry-Based Text Line Detection in Natural Scenes.* In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2558–2567, 2015. (Cited on pages 10 and 107.)

[Zhu 2007a] Qihui Zhu, Gang Song and Jianbo Shi. *Untangling cycles for contour grouping.* In Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on, pages 1–8. IEEE, 2007. (Cited on page 45.)

[Zhu 2007b] Song-Chun Zhu and David Mumford. A stochastic grammar of images. Now Publishers Inc, 2007. (Cited on page 82.)

[Zhu 2010a] Long Zhu, Yuanhao Chen and Alan Yuille. *Learning a hierarchical deformable template for rapid deformable object parsing.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 32, no. 6, pages 1029–1043, 2010. (Cited on page 82.)

[Zhu 2010b] Long Leo Zhu, Yuanhao Chen, Antonio Torralba, William Freeman and Alan Yuille. *Part and appearance sharing : Recursive Compositional Models for multi-view.* 2010. (Cited on page 4.)

[Zhu 2012a] Xiangxin Zhu and Deva Ramanan. *Face detection, pose estimation, and landmark localization in the wild.* In Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, pages 2879–2886. IEEE, 2012. (Cited on page 71.)

[Zhu 2012b] Xiangxin Zhu, Carl Vondrick, Deva Ramanan and Charless Fowlkes. *Do We Need More Training Data or Better Models for Object Detection ?* BMVA Press, 2012. (Cited on pages 53, 58 and 59.)

[Zitnick 2014] C Lawrence Zitnick and Piotr Dollár. *Edge boxes : Locating object proposals from edges.* In Computer Vision–ECCV 2014, pages 391–405. Springer, 2014. (Cited on pages 13, 38 and 47.)