Unsupervised Learning of Categorical Segments in Image Collections

Thesis by

Marco Andreetto

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

California Institute of Technology

Pasadena, California

2011

(Defended January 12, 2011)

To my parents, Sante and Emilia, and my brother Paolo

Acknowledgements

There are many people I want to to thank for this dissertation and for completing my doctorate here at Caltech. First of all, I would like to thank my advisor, Prof. Pietro Perona, for his constant help and support during all these years at Caltech. I have learned a lot from him, both as a researcher and as a person. I also would like to thank Prof. Yaser Abu-Mostafa, Prof. Serge Belongie, Prof. Babak Hassibi, and Prof. Max Welling, for agreeing to be part of my defense committee, for their useful comments on my dissertation, and for the words of appreciation they had for my work.

The help of my collaborators has been fundamental in completing my graduate studies. I had the great privilege of collaborating with Prof. Lihi Zelnik-Manor on most of what I present in this dissertation. Her sharp observation and constant positive attitude have been essential for completing this work. I also had the privilege of working with Prof. Silvio Savarese on my project on 3D shape reconstruction from ray tracing constraints.

A special thank goes to my first academic advisor and personal friend, Prof. Guido Maria Cortelazzo. It is because of his advice and encouragement that I started this endeavor, and for that and the way my whole life is now, I will be forever grateful to him.

I would like to thank my fellow members at the Computational Vision Lab. They were great coworkers and wonderful friends and I have learned a lot from all of them. Here

they are (in the random order of cut and paste): Greg Griffin, Mohamed Aly, Ron Appel, Xavier Burgos-Artizzu, Piotr Dollar, Eyrun Eyjolfsdottir, Ali Lashgari, Michael Maire, Merrielle Spain, Peter Welinder, Ryan Gomes, Andrea Boyle, Pierre Moreels, Anelia Angelova, Christophe Basset, Alex Holub, Fei Fei Li, Marc'Aurelio Ranzato, Kristin Branson, Evgeniy Bart, Takeshi Mita, Seigo Watanabe and Greg Griffin (again!).

I want to thank my closest friend Claudio Fanti and his wife Ting. They have been my best friends and constant companions during all these years. I treasure the time I spent with them and all the meals we shared at Din Tai Fung. I also want to thank Mandy, from Din Tai Fung, whose smile was so often in my mind.

Finally, I want to thank the most important people in my life: my parents, Sante and Emilia, and my brother, Paolo, for their endless love and support. They have contributing more than anybody else in shaping every aspect of myself, and I hope they will b proud of this accomplishment, which as much mine as it is theirs.

Abstract

Which one comes first: segmentation or recognition? We propose a unified framework for carrying out the two simultaneously and without supervision. The framework combines a flexible probabilistic model for representing the shape and appearance of each segment, with the popular "bag of visual words" model for recognition. If applied to a collection of images, our framework can simultaneously discover the segments of each image, and the correspondence between such segments, without supervision. Such recurring segments may be thought of as the "parts" of corresponding objects that appear multiple times in the image collection. Thus, the model may be used for learning new categories, detecting/classifying objects, and segmenting images, without using expensive human annotation.

Contents

Acknowledgements **v**

Abstract **vii**

List of Figures **xiii**

1 Introduction **1**

2 Previous Work **5**

2.1 Top-down segmentation 5

2.2 Bottom-up segmentation 7

2.3 Joint segmentation and recognition methods 10

3 A Probabilistic Model for Single-Image Segmentation **11**

3.1 Basic probabilistic model 12

3.2 Modeling segment distributions 15

3.2.1 Non-parametric segment model 15

3.2.2 Parametric segment model 16

3.2.3 Semi-parametric segment model 17

3.3 Modeling the mixing coefficients 18

3.4 Inference 22
3.4.1 MCMC inference algorithm 22
3.4.2 Variational inference 24
3.4.3 Kernel density estimation of f_k 27
3.4.3.1 Connection to spectral clustering 29
3.4.3.2 Connection to kernel k-means 30
3.5 Experiments 31
3.6 Partial labeling 35

4 Video Segmentation **39**
4.1 Temporal coherence in videos 39
4.2 Video segmentation algorithm 40
4.3 Experimental results 43

5 Segmenting Image Collections **45**
5.1 Semi-parametric LDA model (SP-LDA) 45
5.2 Experiments 47

6 Learning Categorical Segments in Image Collections **51**
6.1 Modeling recurring segments 51
6.2 Inference algorithms 55
6.2.1 Sampling-based inference 55
6.2.1.1 Gibbs sampling 55
6.2.1.2 Block sampler 57

6.2.2 Variational inference 61

7 Experimental Results **67**

7.1 Evaluation metrics 69

7.2 Comparing different types of visual words 71

7.3 Comparing different inference algorithms 78

7.4 Comparison with other probabilistic models 84

7.5 Accuracy vs. category sample size 90

8 Conclusions **93**

Bibliography **95**

List of Figures

1.1 Segmentation and recognition task. (a) An input image for a generic Computer Vision algorithm. (b) The segmented image with different segments labeled with different category: cow (orange), grass (green), and sea (light blue). 2

2.1 In top-down segmentation, a model for a particular category, in the example a cow, is constructed using a human-annotated training set. This model can represent the appearance of the elements of the category as well as the shape of the segment. Given a test image containing an object from the category, the segmentation algorithm uses the model to identify the object and collect the segment. 6

2.2 Bottom-up segmentation. Given an input image different local are extracted for each pixel. These cues are based on properties of the patch centered in each pixel or the contours between a pair of pixels. Using the cues dissimilarity measures are computed and used as input of a clustering algorithm that returns the final segmentation. 7

2.3 Segmentation results from different segmentation algorithms. (a) Normalized cut [SM00]. (b) Felzenszwalb and Huttenlocher segmentation algorithm [FH04]. (c) The gPb-ucm-owt algorithm [AMFM09]. (d) Meanshift clustering algorithm [CM02]. 9

3.1 Left: plate diagram [Jor04] of our generative model for image segmentation. The gray node x_n represents the observations (pixel features). The node c_n represents the segment assignment for the observation x_n. The node θ represents the mixing coefficients for each segment. The two rounded boxes α and f_k represent the hyperparameters for the Dirichlet distributions over θ and the density function for each segment k. Finally N is the total number of pixels in the image and K is the number of segments in the image. Right: image formation process as described by the graphical model. An image is composed of two segments: ground (45% of the image) and sky (55% of the image). An observation x_n is obtained by first sampling the assignment variable c_n. Assuming $c_n = 1$, the corresponding density f_1 is used to sample x_n as member of the ground segment. Similarly, a second observation, x_m, in the sky segment is sampled from the corresponding density f_2 when $c_m = 2$. 12

3.2 Modeling outliers with a uniform distribution (garbage collector cluster). (a) Input data. (b) Segmentation by spectral clustering using 3 clusters: the outliers are arbitrarily assigned to the 3 clusters. (c) Segmentation by spectral clustering into 4 clusters: even with an additional cluster the outliers are assigned to the main clusters and one of the three clusters is randomly split. (d) Our segmentation: using a parametric (uniform) distribution for one cluster results in correctly identifying the three clusters and the outliers (crosses). . . 17

3.3 Effect of Prior. (a) Cluster size probability with Dirichlet prior $\alpha = [100, 100]$. (b) Clustering result with $K = 2$ and the prior in (a) preferring clusters of equal size. (c) Cluster size probability with Dirichlet prior $\alpha = [200, 25]$. (d) Clustering result with $K = 2$ and the prior in (c) preferring one large cluster and one small cluster. 19

3.4 Zipf's law for relative segment size in images (a) The blue line is Zipf's law with the power $s = 1.2$ (in loglog representation). Each of the other curves represents the segment sizes in a human segmentation of an image in the Berkeley dataset [MFTM01]. (b) The blue curve represents the constants α_k used as prior for the segmentation in Figs. 3.3.c,f. The red curve represents the obtained segment sizes. 20

3.5 Unsupervised image segmentation. Example results from the two data sets we experimented on. Columns 2, 4, and 5 show segmentations of three images (column 1) using a Gaussian mixture model (GMM), normalized cuts (Ncut) and our semi-parametric mixture model (SPMM), respectively. The images shown in rows 1 and 2 come from a collection of 16 general pictures; the bottom image was selected from the 100 Egret images (the same experiment was carried out on all images in both collections, see supplemental material). The number of segments was set to 8 for general images, and to 4 for the Egrets. Columns 3 and 6 show assignment probabilities, where the color of a pixel is a convex combination of the segment markers according to segment assignment probabilities. 21

3.6 Human Ratings. Six people rated the unsupervised segmentation results of all the images in our data sets (Section 3.5) as good, OK, or bad. The plots show the rating statistics for each experiment and each method. Each bar is split into three parts whose sizes correspond to the fraction of images assigned to the corresponding rating. Better overall performance corresponds to less red and more blue. Our method outperforms other methods in both experiments. 21

3.7 Comparison between the Gaussian mixture model (GMM) and the semi-parametric mixture model (SPMM) of Section 3.2.3. The colors of the sky segment are not well modeled by a unimodal distribution: the left part has a more uniform color than the right part, where some clouds are present. The GMM segmentation (center) splits the sky into two components, while the semi-parametric segmentation (right) correctly assigns the sky to a single segment. Fig. 3.8 shows the observations in each segment projected on different coordinate planes of the xy-RGB feature space. The bottom row shows a sample image from the estimated segmentations from the GMM model (center) and from the semi-parametric model (right). 32

3.8 Comparison between the different segmentations in Fig. 3.7. Each plot shows different coordinate planes of the xy-RGB feature space. The left column refers to the GMM segmentation the right column to the SPMM one. The points correspond to the projections of the image pixels. The ellipses represent Gaussian distributions (the parametric term for the SPMM). The colors of points and ellipses correspond to the segments in Fig. 3.7. 33

3.9 Comparison of the Gibbs-sampler and variational inference methods for the image segmentation problem. The first column shows the original images, the second one shows the segmentation results of the Gibbs sampler used in [AZMP07], and the third one shows the segmentation results of our variational method. 36

3.10 Partial labeling. A typical result of intensity-based image segmentation into 2 clusters (out of 100 images in the Egret set of [LSP05]). (a) Original image, (b) GMM-EM clustering, (c) normalized cuts, (d) our result with partial labeling. Boundary pixels were constrained to the background cluster. 37

3.11 Partial labeling, comparison with GrabCut. Left: input image. Right: our segmentation result, obtained by manually labeling part of the image as background. Refer to [RKB04] for the corresponding GrabCut segmentation. 38

4.1 Video sequence segmentation. Left column: Frames 218, 280, 282, 284, 286, and 329 out of a 343-frame-long video. Middle column: normalized cut segmentation results. Right column: SPMM result while enforcing spatiotemporal coherence across frames is significantly better. See Fig. 4.3 for human rating of the segmentation results. The complete video as well as results on a different video are provided in the supplemental material. 42

4.2 Another video sequence segmentation. First column: Frames 61, 136, and 154 out of a 193-frame-long video. Second column: GMM segmentation results. Third column: normalized cut segmentation results. Right column: SPMM result while enforcing spatiotemporal coherence across frames is significantly better. The complete video as well as results on a different video are provided in the supplemental material. 43

4.3 Human Ratings. Six people rated the video segmentation results of a subset of all the frames in the "ballet" sequence. As for the results in Section 3.5) the possible rates were: good, OK, or bad. The plots show the rating statistics for the SPMM with video coherence (top bar) and for the normalized cut (bottom bar). Each bar is split into three parts whose sizes correspond to the fraction of images assigned to the corresponding rating. Better overall performance corresponds to less red and more blue. Our method outperforms clearly outperforms normalized cut. 44

5.1 Semi-parametric Latent Dirichlet Allocation model (SP-LDA) for joint segmentation of image collections (see Section 5.1). As in Fig. 3.1, the gray node x_{mn} represents the observed quantities (features vector n for image m in the collection). The node c_{mn} represents the segment assignment for the observation x_{mn}. The node θ_m represents the mixing coefficients for each segment in image m. The rounded box α is the hyperparameter of the Dirichlet distribution of θ_m. The inner plate represents the N_m pixels in image m, while the outer plate represents all the M images in the collection. The K distributions f_k^s model the recurring objects in the collection and are shared across all the images. The H distributions $f_{h,m}^{ns}$ are local to each image, i.e., independent of the rest of the collection, and represent the image-specific segments. 46

5.2 Segmenting an image collection. First row: six examples out of a collection of 30 images of faces on different backgrounds. Second row: corresponding ground truth segmentation of the face. Rows three to five: binary segmentations with different numbers of shared segments. Rows six to eight: segmentation in three segments with different number of shared segments. K is the number of shared segments and H is the number of image-specific ones. . 48

5.3 Precision/recall for the face collection. Different markers correspond to the performance of the SP-LDA model (Fig. 5.1) for different settings of the parameters K (number of shared segments) and H (number of image-specific segments). The green curves correspond to precision/recall values with the same harmonic mean (F measure [Rij79]). 49

6.1 The affinity-based LDA model (A-LDA) for learning categorical segments (see Section 6). The two gray nodes x_{mn} and w_{mn} represent the observed quantities in the model: the feature vector (position and color) and the visual word associated with each pixel, respectively. The nodes c_{mn}, $f_{k,m}$, ϕ_k, and θ_m are hidden quantities that represent the segment assignment for x_{mn} and w_{mn}, the probability density of the feature vectors in segment k of image I_m, the visual words distribution for segment k, and the sizes of the segments in image m, respectively. The two squares with rounded corners α and ε represent the hyperparameters of the Dirichlet distributions over θ_m and ϕ_k, respectively. Finally, K is the number of segments, N_m is the number of pixels in image m, and M is the number of images in the collection. 52

6.2 Block sampler. (a) A starting segmentation which assigns part of one object (the legs of the cow) to the wrong segment (grass). (b) The block sampler selects a set of pixels that are likely to have the same label (red region). (c) The sampler reassigns all the pixels in the proposed region to a new segment (the same the cow). 59

7.1 Filter banks visual words. The schema shows how visual words are computed using a filter bank. The different color channels in the images are filtered with different Gaussian (low-pass filters for capturing color information) and gradient (high pass filter for capturing edges and texture information) filters. After the filtering each pixel is represented by an 18 dimensional vector. The visual words are obtained by running kmeans over all the pixels, and assigning the discrete label of the cluster to which a pixel is assigned. . . 68

7.2 Error measures. For each image used in the experiments the ground truth segmentation (GT) is available (orange region). The result segment for the cow category obtained from the A-LDA is displayed in dark green. The intersection of the two regions is the set of correctly identified pixels in the image (magenta). 69

7.3 Visual words dictionaries. Left: 256 visual words when the pixel color is used as descriptor. Right: average of the patches associated to 256 visual words when the filter bank is used as descriptor. 71

7.4 Unsupervised segmentation and recognition results when only RGB information is used to construct the visual words. Three panels are presented. In each of the three panels we present the original image, the segmentation using the A-LDA model, and the segmentation using the LDA model. The three panels show the different types of images: cows, trees, and faces. For a specific model the same color in different images identifies the same topic segment. 72

7.5 Precision/recall plots for the MSRC dataset when using visual word based on RGB color. The dictionary size is of 1024 visual words. The number of topics K is set to 20. The F-measure isolines are defined as in Fig. 5.3. . . . 74

7.6 Unsupervised segmentation and recognition when filter responses are used to construct the visual words. Similarly to Fig. 7.4, we present three panels.. In each of the three panels we present the original image, the segmentation using the A-LDA models and the segmentation using the LDA model. The three panels show different types of images: cows, trees, and faces. For a specific model the same color in different images identifies the same topic segment. 75

7.7 Precision/recall results for the MSRC dataset when using visual words based on filter bank responses (red crosses). The dictionary size is 1024 visual words. The number of topics K is set to 20. The precision/recall results for the spatial latent Dirichlet allocation (S-LDA) [WG07] are also reported (black diamonds). 76

7.8 Comparison of the segmentation accuracy of the A-LDA model for different types of visual words: color (RGB) visual words (horizontal axis) and the filter bank visual words (vertical axis). 77

7.9 Four topics/segments learned from the LabelMe database. Each panel contains 8 segments from the same topic. The four topics represent four different elements of a possible street scene: “tree/foliage”, “buildings”, “street pavement”, and “sky”. These topic panels show the consistency we obtain across the images of the collection. 78

7.10 Six topics/segments learned from the Scene database. Each panel contains 8 segments from the same topic. Our visual words representation incorporates color information, therefore skies were assigned to two topics, light blue and dark blue. 79

7.11 Categorical segments from MSRCv1. The top panel shows 12 segments from the category “cows”. The bottom panel shows 12 segments from the category “faces”. These two categories are often confused by the A-LDA model because of the color similarity. 80

7.12 Categorical segments from MSRCv1. The top panel shows 12 segments from the category “tree”. The bottom panel shows 12 segments from the category “grass”. 81

7.13 Categorical segments from MSRCv1. The top panel shows 12 segments from the category “bicycles”. The bottom panel shows 12 segments from the category “sky”. 82

7.14 Precision/recall plots showing the segmentation/recognition performance of the A-LDA on seven categories: airplanes, bikes, buildings, cars, cows, faces, and trees (foliage). The red crosses refer to the unsupervised case (see Fig. 7.7). The blue circles refer to the weakly-supervised case, where the category label of the objects in an image is known. Even this limited amount of supervision, a single label for the whole image, greatly improves performance of the segmentation. 83

7.15 Comparison of sampling algorithms. (a) Precision/recall results showing the segmentation/recognition performance of the Gibbs sampler inference algorithm (blue circle), and the Block sampler together with the Gibbs sampler (red crosses). (b) Scatter plot of the accuracy for the two sampling algorithms. 84

7.16 Comparison of sampling and variational algorithms. (a) Precision/recall plots showing the segmentation/recognition performance of the Gibbs sampler inference algorithm (blue circle) and the variational inference (red crosses). (b) Scatter plot of the accuracy for the Gibbs sampler and variational approximation. 85

7.17 Left: scatter plot comparing the A-LDA model with a Gaussian mixture model (GMM)). We can see that the A-LDA model always outperforms the GMM. Right: scatter plot comparing the A-LDA model with an LDA model. In this case the A-LDA model has better accuracy for almost all categories. All of the three models use 20 segments and are unsupervised. 86

7.18 LDA results from MSRCv1. The top panel shows 12 segments from the category “faces”. The bottom pannel shows 12 segments from the category “cow”. See Fig. 7.11 for the corresponding results from the A-LDA model. . 87

7.19 Accuracies of different classes as the size of the faces category in the collection increases. The accuracy for the faces category (solid orange) keeps improving as the size of this class increases. The accuracies of other categories like grass, foliage, and buildings are fairly constant. The accuracy for the cow category decreases as the number of pixels in the faces category increases, suggesting that it is more difficult to discriminate between these two categories given our visual words. We confirmed this effect by exploring segmentation results for individual images: the reddish cows are sometimes confused with pink-brown faces. 91

List of Tables

7.1 Comparison of our model (A-LDA) with the probabilistic model of Wang and Grimson (S-LDA) [WG07]. 88

7.2 Segmentation accuracy (in percent) for the MSRCv2 dataset. The results are divided in two tables. The first row of each table reports the accuracy for the MSRCv1 dataset, a subset of the MSRCv2 dataset 88

Chapter 1

Introduction

Given an image, like the one presented in Fig. 1.1a, a possible computer vision task is to recognize the content of the image: for example the image in Fig. 1.1a contains a cow in the foreground and grass and sea in the background (bottom and top part, respectively). At a finer scale we may want to label each pixel in the image with the name of the object in the real world that generated the pixel in image. The resulting partition, shown in Fig. 1.1b, of the image is called segmentation and the set of pixels with the same label are called segments.

Image segmentation and recognition have long been associated in the vision literature. Three views have been entertained on their relationship: (a) segmentation is a preprocessing step for recognition: first you divide up the image into homogeneous regions, then recognition proceeds by classifying and combining these regions [Mar82, MBLS01, RES+06, CFF07]; (b) segmentation is a by-product of recognition: once we know that there is an object in a given position, we may posit the components of the object and this may help segmentation [LLS04, BU02]; (c) segmentation and recognition may be performed independently: in particular, recognition does not require segmentation nor grouping [WWP00, VJ04, Low04, FPZ03, FFP05]. These views are not mutually exclusive,

(a) (b)

Figure 1.1: Segmentation and recognition task. (a) An input image for a generic Computer Vision algorithm. (b) The segmented image with different segments labeled with different category: cow (orange), grass (green), and sea (light blue).

while segmentation and recognition are not necessary for each other; both benefit from each other. It is therefore intuitive that recognition and segmentation might have to be carried out together, rather than in sequence, in order to obtain the best results. We explore here the idea of carrying out category learning for recognition and segmentation jointly – we propose and study a simple probabilistic model that allows a unified view of both tasks. Our model represents each image as a composition of segments, where a segment could correspond to a whole object (e.g., a cow) or to a part of an object (e.g., a leg), to a patch of a distinctive texture, or to a "nonsense" homogeneous region in the background. The inference process divides each image into segments, and discovers segments that are similar across multiple images, thus discovering new visual categories.

We build upon recent work on recognition and segmentation. First, we choose to represent image segments using simple statistics of "visual words" as features. Using "bags of visual words" to characterize the appearance of an image segment combines an idea coming from the literature on texture, where Leung and Malik [LM01] proposed vector-quantizing

image patches to produce a small dictionary of "textons", and an idea from the literature on document retrieval, where statistics of words are used to classify documents [BNJ03]. Early visual recognition papers using "bags of visual words" considered the image as a single bag [VNU03, DS03, FFP05], while recently we have seen efforts either to classify independently multiple regions per image, after image segmentation [RES+06, CFF07, RVG+07], or to force nearby visual words to have the same statistics [WG07]. Recent literature on image segmentation successfully combines the notion that images are "piecewise smooth" with the notion that segments shapes are more often than not "simple". These insights have been pursued with parametric probabilistic models [TZ02, OB07], with non-parametric deterministic models [SM00], and with nonparametric probabilistic models [AZMP07]. The latter is a very simple probabilistic formulation which, as we shall see, combines gracefully with the popular LDA model for visual recognition.

Our work most closely builds upon two papers. Russell et al. [RES+06] first proposed to model image segments, rather than the whole image, with "bag of visual words" point of view to image segments, rather than to the entire image, in the hope of discovering multiple objects in each image. Our work combines segmentation and category model learning in one step, rather than first carrying out segmentation and then categorizing the segments. Furthermore, while Russell et al.'s segmentation is independent for each image, in our work segmentation is carried out simultaneously and each segment's definition benefits from related segments being simultaneously discovered in other images. Conversely, Andreetto et al. [AZMP07] segment an entire collection of images simultaneously, while discovering the correspondence between homologous segments. However, the features that pair segments

are restricted to size, shape, and average color of the segments. Associating bags of visual words to each segment allows us to discover more interesting visual connections between corresponding segments, and thus discover visual categories.

We develop the simultaneous segmentation/recognition scheme step by step. We start (Chapter 3) by proposing a probabilistic model for segmenting individual images. We then generalize the model so that information is shared across images, and entire image collections may be segmented simultaneously (Chapter 5). Finally we further extend the model to incorporate a richer set of visual features (Chapter 6). This provides a model for automatic inference of categorical segments.

Chapter 2

Previous Work

Image segmentation has long been studied in Computer Vision and a large number of solutions have been proposed. Rather than an extensive review, we concentrate on the two classes of solutions that are most relevant to the problem we are addressing. For a more complete review please refer to [AMFM10, UPH07]. The two classes are *top-down segmentation* and the *bottom-up segmentation*. In more recent years, many new methods that try to combine both the top-down and the bottom-up have been proposed. These joint segmentation methods are the ones closer to the algorithms proposed in this thesis.

2.1 Top-down segmentation

In top-down approaches an object from a specific category is identified in an image and the segment containing that object is extracted. An early example of these approaches is given by Borestein and Ullman [BU04]. In this class of algorithms the segmentation is a consequence of the recognition task.

To identify the objects the segmentation algorithm needs a model that represents the visual properties of the category we want to recognize. This model can describe the ap-

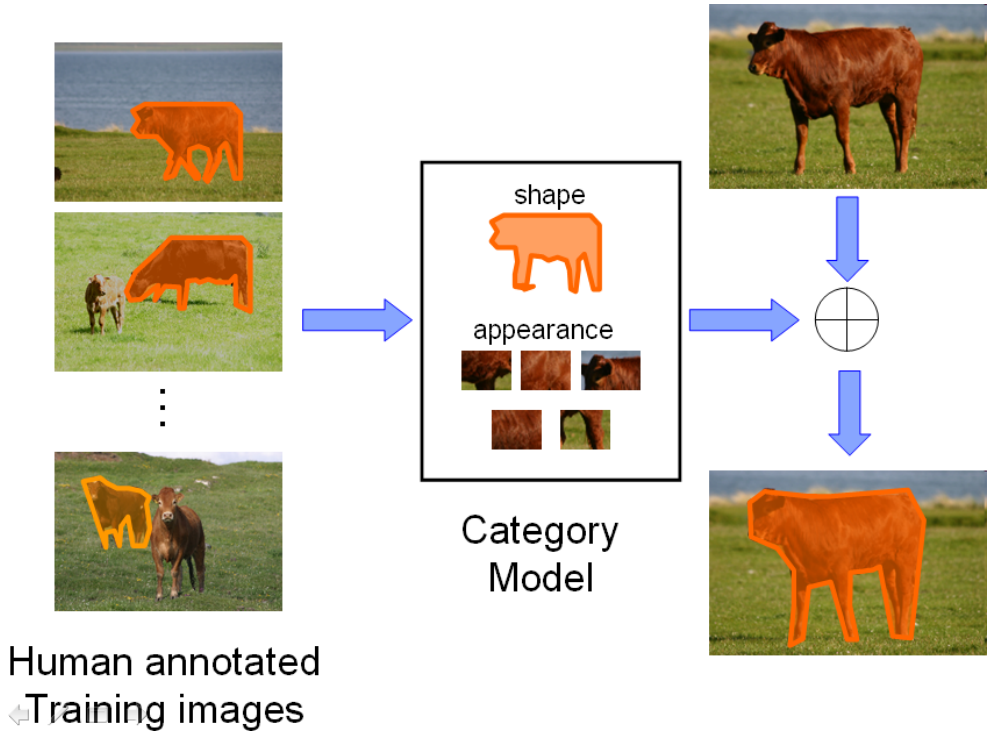


Figure 2.1: In top-down segmentation, a model for a particular category, in the example a cow, is constructed using a human-annotated training set. This model can represent the appearance of the elements of the category as well as the shape of the segment. Given a test image containing an object from the category, the segmentation algorithm uses the model to identify the object and collect the segment.

pearance of the objects in the category and the shape of the segments as depicted in Fig. 2.1. In these algorithms, low-level segmentation cues such as texture and contours are used to obtain uniformly labeled regions by means of Markov random fields [VT07], conditional random fields [SWRC09], or indirectly by training a classifier that consider the segmentation cues over a large region of the image [SJC08]. Alternatively a superpixel representation of the image can be first obtained, with the superpixels classified using the category model [FVS09].

Top-down segmentation algorithms give good segmentation and recognition results, but require an elevated level of human annotation which can be quite expensive to obtain

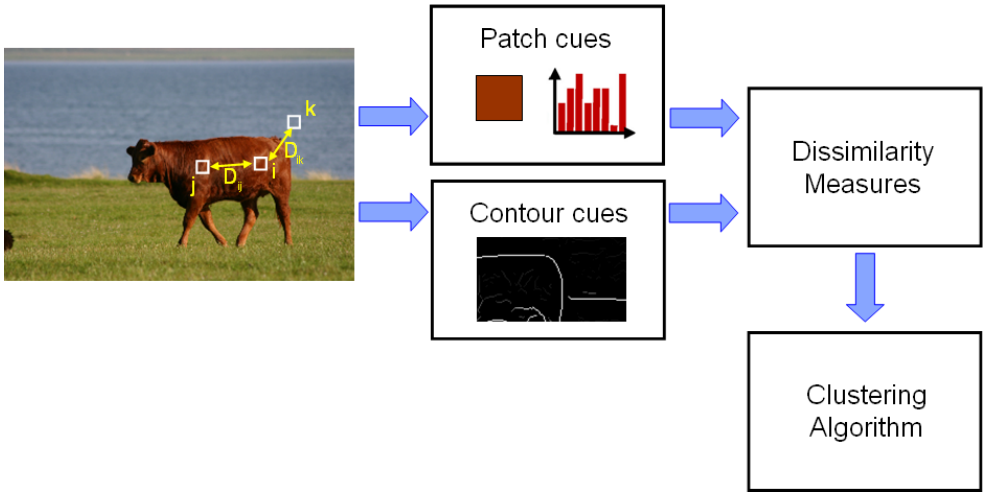


Figure 2.2: Bottom-up segmentation. Given an input image different local are extracted for each pixel. These cues are based on properties of the patch centered in each pixel or the contours between a pair of pixels. Using the cues dissimilarity measures are computed and used as input of a clustering algorithm that returns the final segmentation.

[MFTM01, WBBP10]. Also, these algorithms can detect only the objects specified during the training phase, when the category models are constructed. Therefore new objects can't be detected if they start appearing in the testing set, even if there is sufficient evidence to separate them from the "background clutter".

2.2 Bottom-up segmentation

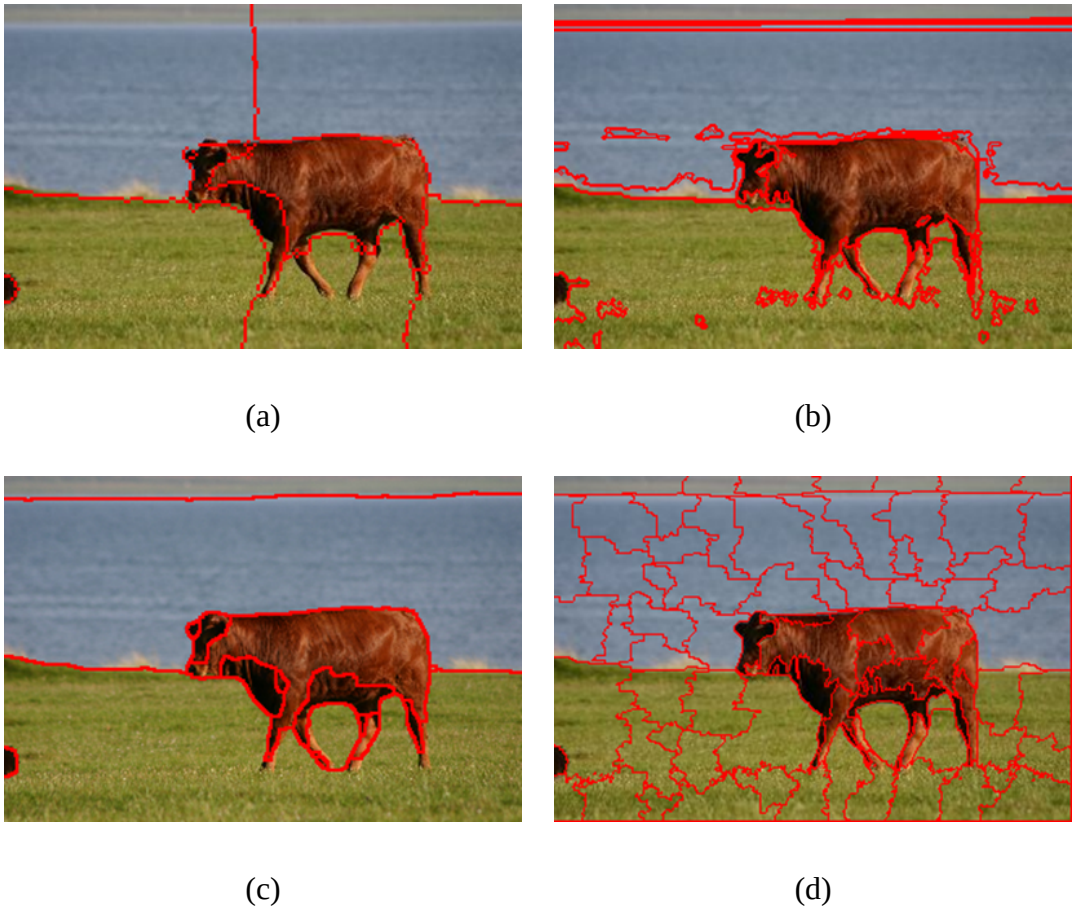
Bottom-up segmentation algorithms are agnostic about the content of the image. Rather than segmenting a specific category or a set of categories of objects they try to group pixels in the image according to the similarity (or the dissimilarity) of the properties of the single pixels.

Fig. 2.2 shows the conceptual structure of a bottom-up segmentation algorithm. Given an input image, several cues are computed for each pixels. These cues may describe the

color of the pixel, its texture (as histogram of textons), and the response of a contour operator, such as canny [Can86] or Pb [FMM03]. Using these cues it is possible to compute the dissimilarity between pairs (or its "inverse" what we call the affinity). For example, the dissimilarity D_{ij} between pixels i and j highlighted in Fig. 2.2 should be very small given the similar color and texture of the patches centered on those two pixels and the lack of contours between them. On the other hand, the dissimilarity between pixels i and k should be larger because of the contour between the two pixels and the different texture and color statistics of the two corresponding patches. Given the dissimilarity measures, or the affinities, between all pairs of pixels, the final segmentation is obtained using a clustering algorithm. This clustering algorithm can be a generic one, such as spectral clustering and Gaussian mixture model, or a specific one like the gPb-ucm-owt.

While the bottom-up approach can be used for segmenting any natural image[1], the end result is in general different from the desired segmentation presented in Fig. 1.1a. This can be seen considering the segmentation results by four popular bottom-up algorithms for the same input image presented in Fig. 2.3a. We can see that the foreground object, the cow, is divided into three different segments corresponding to regions of the object with different colors. Also the background elements, the grass and the sky, are also subdivided into smaller segments, instead of a single segment as desired. These artifacts are a consequence of the implicit bias of normalized cut toward equal size segments. Given this segmentation it is necessary to perform some additional process to merge segments from the same object. Fig. 2.3b shows the segmentation results for another popular algorithm based on graph par-

[1]Other types of images, such as tissue samples from microscopy, can require a different set of local cues and dissimilarity measure, because of their different visual properties.



(a) (b)

(c) (d)

Figure 2.3: Segmentation results from different segmentation algorithms. (a) Normalized cut [SM00]. (b) Felzenszwalb and Huttenlocher segmentation algorithm [FH04]. (c) The gPb-ucm-owt algorithm [AMFM09]. (d) Meanshift clustering algorithm [CM02].

tition by Felzenszwalb and Huttenlocher [FH04]. This algorithm is very fast to run, but also in this case several artifacts are present and an additional step is necessary to merge redundant segments. Fig. 2.3c presents the current state of the art for bottom-up segmentation methods: the gPb-ucm-owt algorithm [AMFM09]. Although the best-performing algorithm compared against human annotators, even this method is not capable alone of returning the desired segmentation. Finally, Fig. 2.3d shows the results from the popular Meanshift clustering algorithm [CM02]. In this case the image is over segmented in a large number of regions with the property that pixels in the same segment (superpixel) are

extremely likely to belong to the same object. The main purpose of this last algorithm is to provide a more compact representation for an image than the pixel level. A following stage can then group these more descriptive superpixels as purposed in [FVS09].

The limitation of the bottom-up algorithms are in a way expected, since it is unlikely that perfectly segmented objects can be obtained using only low-level information; even a simple image like the one presented in Fig. 1.1a. For this reason some higher notion of object class should be used possibly without the need of training a model with annotated data.

2.3 Joint segmentation and recognition methods

To overcome the limitations of the bottom-up methods several authors have explored new segmentation methods that return multiple segmentation hypotheses for a given image. A subsequent stage can be used to collect the more useful hypothesis for the specific vision task. Among these methods, the more interesting for this work is the one proposed by Russel et al [RES+06] that collect a large set of segmentations for the same image by varying the parameters of the normalized segmentation algorithm. The segments that contain the objects in the image are then retrieved by means of a topic-based probabilistic model.

A very different approach is the one developed by Todorovic and Ahuja [TA06, AT07], where a segmentation tree is computed for a given image. This tree encodes important properties of containment and sub-parts that can be used to match different segments (subtrees) across a collection of images, thus identifying segments containing the same object.

Chapter 3

A Probabilistic Model for Single-Image Segmentation

In order to address the main problem of unsupervised recognition and segmentation in image collection we introduce in this chapter a simple probabilistic generative model for single-image segmentation. Like other probabilistic algorithms (such as expectation-maximization on a mixture of Gaussians) the proposed model is principled, provides both hard and probabilistic cluster assignments, as well as the ability to naturally incorporate prior knowledge. While previous probabilistic approaches are restricted to parametric models of clusters (e.g., Gaussians) we eliminate this limitation. The suggested approach does not make heavy assumptions on the shape of the clusters and can thus handle complex structures. We developed different inference algorithms for this probabilistic model based on sampling and variational approximation. We also discuss how it is possible to extend this basic model to address several complex computer vision problems such as video segmentation and semi-supervised image segmentation Finally we report experimental results that suggest our approach outperforms previous work on a variety of image segmentation tasks.

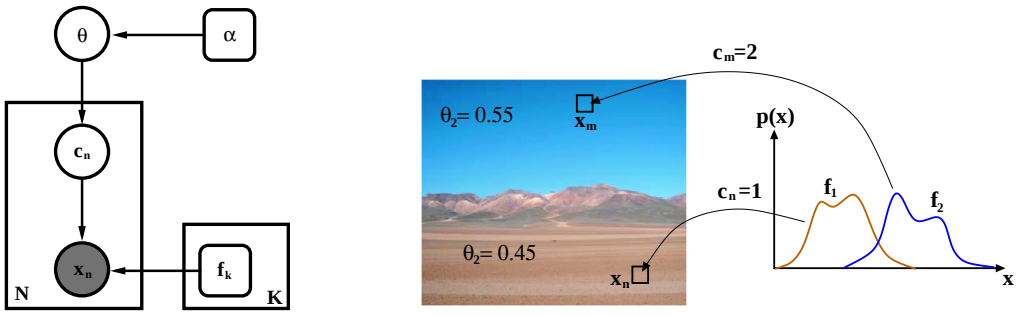


Figure 3.1: Left: plate diagram [Jor04] of our generative model for image segmentation. The gray node x_n represents the observations (pixel features). The node c_n represents the segment assignment for the observation x_n. The node θ represents the mixing coefficients for each segment. The two rounded boxes α and f_k represent the hyperparameters for the Dirichlet distributions over θ and the density function for each segment k. Finally N is the total number of pixels in the image and K is the number of segments in the image. Right: image formation process as described by the graphical model. An image is composed of two segments: ground (45% of the image) and sky (55% of the image). An observation x_n is obtained by first sampling the assignment variable c_n. Assuming $c_n = 1$, the corresponding density f_1 is used to sample x_n as member of the ground segment. Similarly, a second observation, x_m, in the sky segment is sampled from the corresponding density f_2 when $c_m = 2$.

3.1 Basic probabilistic model

Image segmentation techniques may be categorized into three broad classes. The first class consists of deterministic heuristic methods, such as k-means, mean-shift [CM02], and agglomerative methods [DHS00]. When the heuristic captures the statistics of the data the segmentation algoriths perform well. For example, k-means provides good results when the data is blob-like and the agglomerative approach succeeds when clusters are dense and there is little noise. However, these methods often fail with more complex data [NJW01].

The second class consists of probabilistic methods that explicitly estimate parametric models of the data, such as expectation maximization for fitting Gaussian mixture models

(GMM) [CBGM02]. The GMM method is principled and can easily be used as a building block of a larger model that addresses a more general task. However, when the data is arranged in complex and unknown shapes, as is the case for images, it tends to fail, as in GMM each class is represented by a Gaussian (see Fig. 3.7).

Complex data are handled well by the third class of methods, consisting of the many variants of spectral factorization [KVV04, NJW01, SM00, ZS05, MS00]. These techniques do not make strong assumptions on the shape of clusters, and thus generally perform well on images. Unfortunately, spectral factorization lacks a probabilistic interpretation, which makes its use in more general problems, such as recognition and segmentation or segmentation with prior knowledge, somewhat convoluted [YS04], if not impossible.

We propose a generative probabilistic model that can describe segments of complex shape and appearance and can easily be used as a building block for a more complex probabilistic model. Unlike previous probabilistic models, it contains a non-parametric component allowing complex-shaped groups to be modeled faithfully. Unlike factorization methods, it is probabilistic in nature, allowing easy extensions to situations where prior information is available, and integration into larger probabilistic models that address more complex problems such as recognition and motion segmentation [AZMP07].

Let $x_1, x_2, \ldots, x_N$ be a set of observations in $\mathbb{R}^D$ generated from K independent processes $\{C_1, \ldots, C_K\}$. Each process C_k is described by a density function $f_k(x)$. These density functions are not restricted to any specific parametric family, such as Gaussian densities; we only assume that they are smooth functions (see Section 3.2.1). The observations $x_1, x_2, \ldots, x_N$ are generated as follows (see Fig. 3.1):

1. Select a set of K mixing coefficients $\theta_1, \theta_2, \ldots, \theta_K$, drawing them from a probability distribution $p(\theta)$ (see Section 2.2). Each θ_k will correspond to a process C_k.

2. For n equal 1 to N:

    3. Select one of the K processes C_k by sampling the hidden variable c_n according to a multinomial distribution with parameters $\theta_1, \theta_2, \ldots, \theta_K$.

    4. Draw the observation x_n according to the process-specific probability density function $f_k(x)$.

Rather than obtaining samples from the model of Fig. 3.1, we are interested in the inverse problem: computing the posterior distribution of the hidden variables $\boldsymbol{c} = \{c_1, c_2, ...c_N\}$ given the observed variables $\boldsymbol{x} = \{x_1, x_2, ...x_N\}$. Using Bayes' theorem we have:

$$p(\boldsymbol{c}|\boldsymbol{x}) \propto p(\boldsymbol{x}|\boldsymbol{c})p(\boldsymbol{c}) \tag{3.1}$$

where the mixing coefficients θ_k have been marginalized out from the joint distribution $p(\boldsymbol{c}, \theta)$ leaving just the prior term $p(\boldsymbol{c})$. If we assume that the x_i are independent given the c_i, then the likelihood term is defined as:

$$p(\boldsymbol{x}|\boldsymbol{c}) = \prod_{n=1}^{N} p(x_n, c_n) = \prod_{n=1}^{N} f_{c_n}(x_n). \tag{3.2}$$

So far we have not made any assumptions on the structure of the segments, i.e., on $f_k(x)$. In the following sections we describe how the segments densities $f_k(x)$ and the prior $p(\boldsymbol{c})$ are modeled.

3.2 Modeling segment distributions

3.2.1 Non-parametric segment model

If the $f_k(x)$ are Gaussians, then the model is a Gaussian mixture model (GMM). To handle segments of complex shapes and irregular appearances it is best to avoid parametric representations (which may not fit the shape of the segment) and use non-parametric approximation of the densities $f_k(x)$.

Given a kernel function $K(x_i, x_j)$ [Was06] representing the affinity A_{ij} between observations x_i and x_j (i.e., how much we believe the two observations originated from the same process when all we know is their coordinates x_i and x_j), and a set of N_k observations drawn from the unknown distribution $f_k(x)$, a non-parametric density estimator for $f_k(x)$ is defined as:

$$\hat{f}_k(x) = \frac{1}{N_k} \sum_{n=1}^{N_k} K(x, x_n). \tag{3.3}$$

This is equivalent to placing a little probability "bump", the kernel $K(x_i, x_j)$, around each observation x_n sampled from the segment density f_k and approximating the segment distribution as the normalized "sum" of all the "bumps". If the density function $f_k(x)$ is sufficiently smooth, and if a sufficient number of samples x_n are available, $\hat{f}_k(x)$ is a good estimate. A typical choice for the kernel function is the Gaussian:

$$K_{\sigma_j}(x, x_j) = \frac{1}{\sqrt{(2\pi)^D |\Sigma_j|}} \exp\left(-\frac{1}{2}(x - x_j)^T \Sigma_j^{-1} (x - x_j)\right)$$

where Σ_j is a local covariance matrix that can be set according to local analysis as suggested

in [ZMP05, BRCS07]. Other kernel functions may be used as well [CBS05]. For example, in image segmentation we may wish to set to zero the connectivity between far away pixels to enforce a locality of the segmentation or to obtain a sparse problem. The kernel in this case will be a product of a Gaussian kernel and two "box kernels":

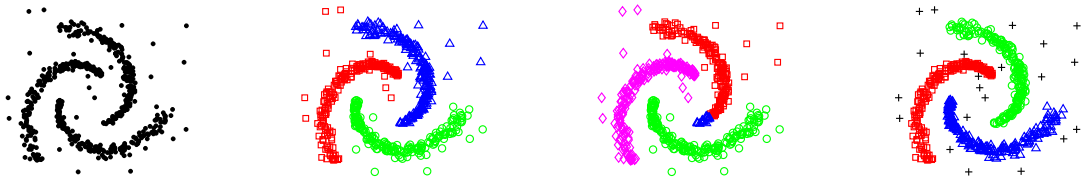
$$K(x, x_j) = K_L(r, r_j)K_L(s, s_j)K_{\sigma_j}(l, l_j) \tag{3.4}$$

where r_j, s_j are the image coordinates of the j'th pixel and l_j is its intensity. The box kernel is defined as: $K_L(r, r_j) = \frac{I((y-y_j)/2L)}{2L}$ and $I(a) = 1$ for $|a| \leq 1$ and 0 otherwise. L is the radius of the box kernel and K_{σ_j} is as defined above.

3.2.2 Parametric segment model

When it is known apriori that some segments are distributed according to some parametric form one should incorporate this information. This is easily done within the proposed framework by using parametric models for the segment densities $f_k(x)$. For example, when it is believed the data generated by one segment is "lumpy", it may be described by a Gaussian density: $f_k(x) = \mathcal{G}(x; \mu_k, \Sigma_k)$. Uniformly distributed outlier points can be represented as a segment with uniform density: $f_k(x) = \frac{1}{Vol(B)}$ if $x \in B$ and 0 otherwise, where B is the data bounding box. We assume that the densities of different segments are independent, thus different types of models can be used for each one (i.e., we can have a mixture of non-parametric and parametric clusters and a variety of parametric models).

Fig. 3.2 presents an example where this becomes useful. The data contains three spiral clusters and random outlier points. Clearly, fitting a mixture of Gaussians will not work on



(a) (b) (c) (d)

Figure 3.2: Modeling outliers with a uniform distribution (garbage collector cluster). (a) Input data. (b) Segmentation by spectral clustering using 3 clusters: the outliers are arbitrarily assigned to the 3 clusters. (c) Segmentation by spectral clustering into 4 clusters: even with an additional cluster the outliers are assigned to the main clusters and one of the three clusters is randomly split. (d) Our segmentation: using a parametric (uniform) distribution for one cluster results in correctly identifying the three clusters and the outliers (crosses).

such data. Spectral clustering into three clusters discovers the dense spiral clusters but the outliers are arbitrarily assigned to the closest spiral. Spectral factorization into 4 clusters splits one of the spirals. Applying the suggested probabilistic approach with three non-parametric clusters and one parametric with a uniform distribution results in discovering the three spirals and collecting all the outliers into the uniform distribution cluster.

3.2.3 Semi-parametric segment model

While parametric models provide a good representation in many cases, when dealing with image segments their modeling assumptions on the structure of the data are too often strong. This explains why spectral clustering (which does not assume any structure) outperforms the parametric methods in most image segmentation tasks. However, in many cases assuming a specific parametric model is too restrictive. For example, the overall distribution of a segment can be well represented by a Gaussian distribution (*global* behavior of the segment); yet, this description could be too crude and inaccurate when considering the finer details of the distribution (*local* behavior of the segment), e.g., if it has a jagged boundary.

It is interesting to consider a hybrid representation combining a parametric and a non-parametric component. Intuitively the parametric component captures a coarse blob-like description of the global structure, while the non-parametric component captures the local deviation from it. The simplest such representation is a convex combination:

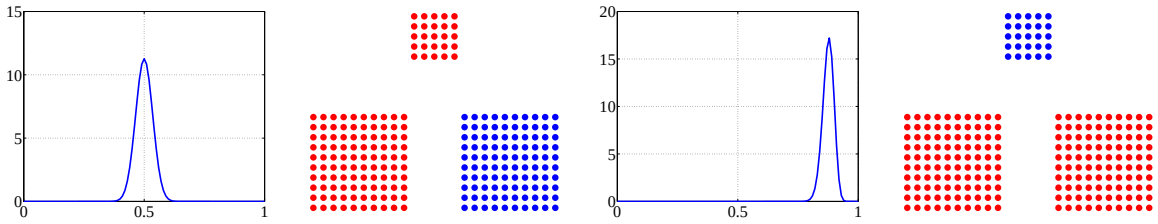
$$\hat{f}_k(x) = (1-\lambda)\frac{1}{N_k}\sum_{j=1}^{N_k} K(x, x_j) + \lambda g_k(x) \tag{3.5}$$

where $g_k(x)$ is a parametric density, e.g., a Gaussian or a uniform density, and $\lambda \in [0, 1]$ represents the relative influence between the two terms (recall that both terms are normalized and sum to 1). We experimented with this representation of the segment distribution and found that it does indeed present numerous advantages with respect to the simpler parametric and non-parametric models (see Section 3.5 and Section 5). In all of our experiments we used $\lambda = 0.1$. An interesting question, which we do not address in this paper, is whether λ could be estimated automatically for each segment. When a semi-parametric representation is used for f_k in the graphical model of Fig. 3.1 we call the overall model a semi-parametric mixture model (SPMM).

3.3 Modeling the mixing coefficients

We assume the mixing coefficients $\theta_1, \theta_2, \ldots, \theta_k$ are distributed as a Dirichlet random variable [BNJ03]:

$$(\theta_1, \theta_2, \ldots, \theta_k) \sim \text{Dir}(\alpha_1, \alpha_2, \ldots, \alpha_K). \tag{3.6}$$

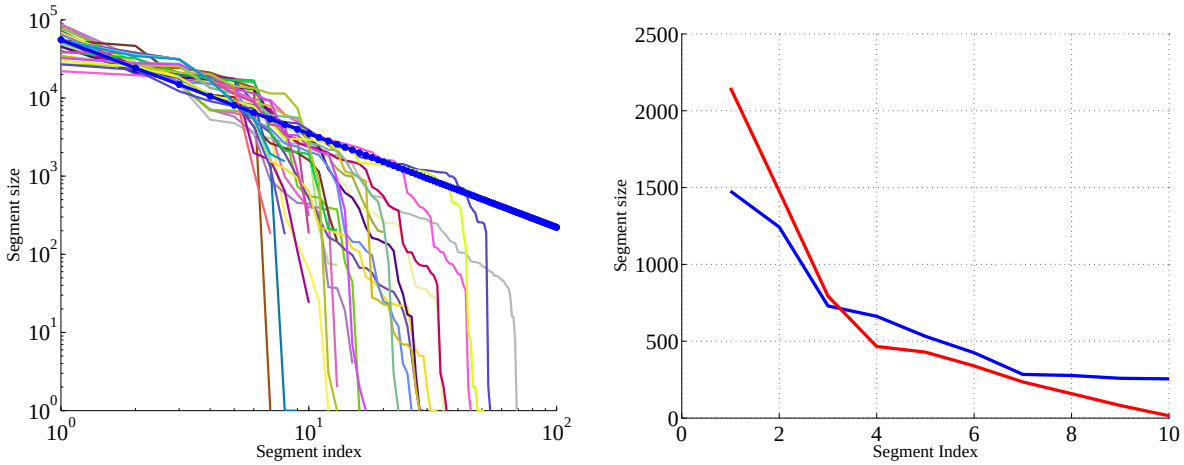


(a) (b) (c) (d)

Figure 3.3: Effect of Prior. (a) Cluster size probability with Dirichlet prior $\alpha = [100, 100]$. (b) Clustering result with $K = 2$ and the prior in (a) preferring clusters of equal size. (c) Cluster size probability with Dirichlet prior $\alpha = [200, 25]$. (d) Clustering result with $K = 2$ and the prior in (c) preferring one large cluster and one small cluster.

Under this assumption the ratio $\alpha_k / \sum_k \alpha_k$ represents the a priori knowledge of the mixing coefficient θ_k, while $\sum_k \alpha_k$ represents the level of confidence in this a priori knowledge. The larger $\sum_k \alpha_k$ is, the stronger is the belief in the mixing coefficients and the corresponding segment sizes. Setting all α_k to the same value suggests that all segments, a priori, have equal size, while if prior knowledge suggests that some segments are larger, e.g., following a power law, this may be incorporated in the model by setting α_k accordingly.

A simple synthetic example showing the effect of the prior is presented in Figure 3.3. By changing the Dirichlet prior parameter α we can "choose" between a segmentation into two similar size segments and a segmentation into one large and one small segment. This can become useful in image segmentation. The highly popular normalized-cut approach to image segmentation [SM00, NJW01] implicitly assumes clusters of equal size. This frequently results in erroneous segmentations. To examine the correctness of the equal segment size assumption we collected statistics of cluster sizes from the manually segmented images in the Berkeley Image Segmentation Dataset [MFTM01]. Fig. 3.3a) shows that a typical distribution of image segment sizes is not uniform but rather similar to Zipf's law [Zip49]. Fig. 3.3 shows that using a Dirichlet prior with α_k set according to Zipf's law can



(a) (b)

Figure 3.4: Zipf's law for relative segment size in images (a) The blue line is Zipf's law with the power $s = 1.2$ (in loglog representation). Each of the other curves represents the segment sizes in a human segmentation of an image in the Berkeley dataset [MFTM01]. (b) The blue curve represents the constants α_k used as prior for the segmentation in Figs. 3.3.c,f. The red curve represents the obtained segment sizes.

improve image segmentation results.

The choice of a Dirichlet distribution for the hidden variable θ is a convenient one, since it allows closed-form derivation of many useful quantities during inference. For example, it is possible to derive the expression for the conditional prior term (see Appendix 3.4.1): $p(c_i = k|\boldsymbol{c}_{-i}) = \frac{N_k+\alpha_k}{N-1+\sum_k \alpha_k}$, where N_k is the size of segments k excluding observation i, N is the total number of observations, and the α_k's are the hyperparameters of the Dirichlet distribution for θ.

Other choices for the distribution of the random variable θ are possible. Of particular interest are non-parametric priors such as the Dirichlet Process [TJBB03], in which the number of segments is automatically discovered during inference, and priors that capture the empirical distribution of segments in natural images [LMH01], such as the one in [SJ08].

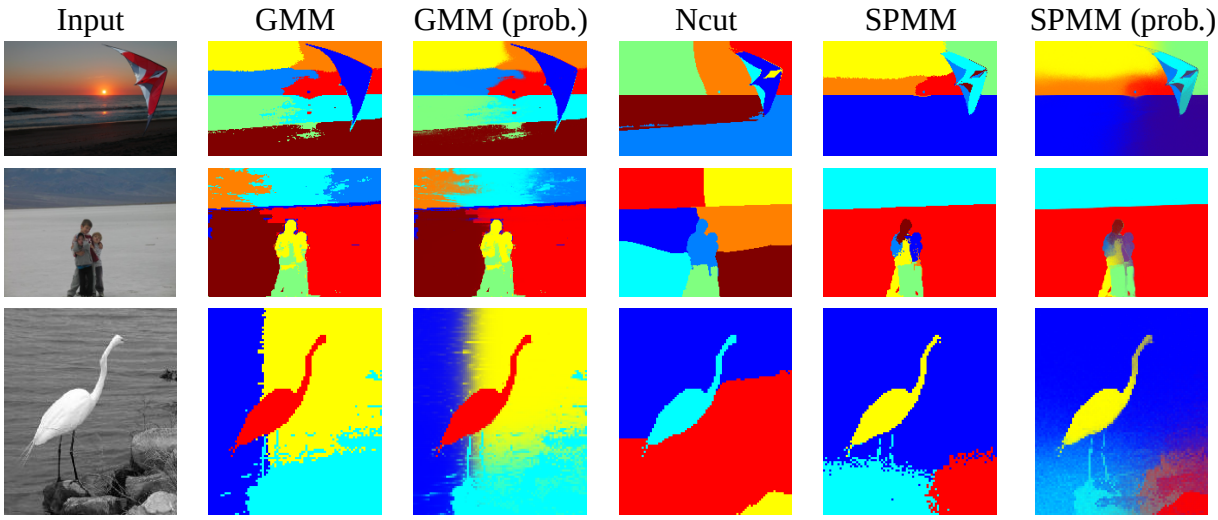


Figure 3.5: Unsupervised image segmentation. Example results from the two data sets we experimented on. Columns 2, 4, and 5 show segmentations of three images (column 1) using a Gaussian mixture model (GMM), normalized cuts (Ncut) and our semi-parametric mixture model (SPMM), respectively. The images shown in rows 1 and 2 come from a collection of 16 general pictures; the bottom image was selected from the 100 Egret images (the same experiment was carried out on all images in both collections, see supplemental material). The number of segments was set to 8 for general images, and to 4 for the Egrets. Columns 3 and 6 show assignment probabilities, where the color of a pixel is a convex combination of the segment markers according to segment assignment probabilities.

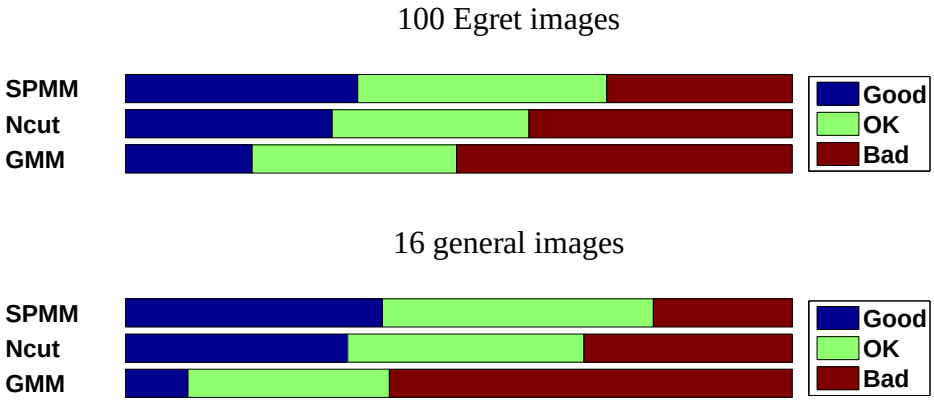


Figure 3.6: Human Ratings. Six people rated the unsupervised segmentation results of all the images in our data sets (Section 3.5) as good, OK, or bad. The plots show the rating statistics for each experiment and each method. Each bar is split into three parts whose sizes correspond to the fraction of images assigned to the corresponding rating. Better overall performance corresponds to less red and more blue. Our method outperforms other methods in both experiments.

3.4 Inference

Since it is not computationally feasible to perform exact inference for the model of Fig. 3.1, we have to use approximate inference. In particular, we developed two inference algorithms: one based on a Markov chain Monte Carlo (MCMC) method [Cas99] and one based variational approximation [Bis06].

3.4.1 MCMC inference algorithm

We first present the inference algorithm for segmenting a single image (model in Fig. 3.1). Let $p(c_n|\mathbf{c}_{-n}, \mathbf{x})$ be the posterior distribution of the segment label c_n for the n'th pixel given the segment labels $\mathbf{c}_{-n}$ of all the other pixels in the image and all the feature vectors $\mathbf{x}$ of all the pixels in the image. Using Bayes' rule we obtain:

$$p(c_n = k|\mathbf{c}_{-n}, \mathbf{x}) \propto p(x_n|c_n = k, \mathbf{x}_{-n}, \mathbf{c}_{-n})p(c_n|\mathbf{c}_{-n}). \tag{3.7}$$

The first term of of Eq. 3.7 is the likelihood of the feature vector x_n to be in the k-th segment. The expression for this term depends on the model used to represent the segment. For example using the non-parametric approximation of Eq. 3.3 we have:

$$p(x_n|c_n = k, \mathbf{x}_{-n}, \mathbf{c}_n) = \hat{f}_k(x_n) = \frac{1}{N_k} \sum_{j \in S_k} K(x_n, x_j) \tag{3.8}$$

where the kernel values $K(x_n, x_j) = A_{nj}$ represent the affinity between x_n, and x_j[1], S_k is the set of observations in segment k, excluding the observation n, and N_k is the cardinality of segment S_k. Similarly if we are using the semi-parametric model of Section 3.2.3 the likelihood terms become:

$$p(x_n|c_n = k, \mathbf{x}_{-n}, \mathbf{c}_n) = (1-\lambda)\frac{1}{N_k}\sum_{j=1}^{N_k} K(x_n, x_j) + \lambda \mathcal{G}_k(x_;\mu_k, \Sigma_k), \qquad (3.9)$$

where $\mathcal{G}_k(x_n; \mu_k, \Sigma_k)$ is a multivariate Gaussian distribution and μ_k and Σ_k are the mean and covariance matrix of segment k. These two quantities could be modeled as additional random variables with suitable prior distribution, for example a normal inverse-Wishart distribution. These random variables could also be sampled from their posterior distribution given the observations $\mathbf{x}$ and the segment labeling $\mathbf{c}$. However, in our experiments we treated them as parameters and computed their value as sample mean and covariance of the observations in each segment (which corresponds to a maximum likelihood estimator for them). This algorithm can be seen as a version of Monte Carlo EM [WT90] with the E-step implemented using the a single Gibbs sampling round for the segment labels $\mathbf{c}$ and the M-step implemented by the maximization of likelihood of the observation and labels over the parameter μ_k and Σ_k of each semi-parametric distribution.

The second term of Eq. 3.7 is the a priori probability for observation n to be in segment k, given the segment labels of all the other observations. Since we are assuming a Dirichlet distribution for the mixing coefficients θ we can marginalize this hidden random and obtain

[1]The A_{nj} are the entries of the affinity matrix used by the normalized cut segmentation algorithm. They can be precomputed before the inference step.

the closed form expression:

$$p(c_n = k|\mathbf{c}_{-n}) = \frac{N_k + \alpha_k}{(N-1) + \sum_k \alpha_k}, \tag{3.10}$$

where N_k is the cardinality of segment S_k, and α_k are the hyperparameters of the Dirichlet distribution of θ.

Using Eq. 3.8, or Eq. 3.9 for the semi-parametric model, and Eq. 3.10 we can compute the posterior distribution in Eq. 3.7. We can therefore run a Gibbs sampling algorithm to obtain samples of $\mathbf{c}$ from $p(\mathbf{c}|\mathbf{x})$. All the quantities used to compute the posterior can either be precomputed, like the affinities $K(x_i, x_j) = A_{ij}$, or updated efficiently like the counts N_k.

Given the samples from $p(\mathbf{c}|\mathbf{x})$ obtained by Gibbs sampling, it is possible to estimate at each pixel the segment assignment probabilities. To obtain a segmentation of the image the MAP estimator at each pixel can be used.

3.4.2 Variational inference

In order to formulate the variational inference on the model of Fig. 3.1 we write down the joint distribution of all the random variables:

$$p(\boldsymbol{x}, \boldsymbol{c}, \theta) = p(\boldsymbol{x}|\boldsymbol{c})p(\boldsymbol{c}|\theta)p(\theta), \tag{3.11}$$

where, given our assumptions on the distributions of the model, the expressions for each of the three factors are given by:

$$\begin{aligned} p(\boldsymbol{x}|\boldsymbol{c}) &= \prod_{n=1}^{N}\prod_{k=1}^{K}\left[f_k(x_n)\right]^{c_n(k)} \\ p(\boldsymbol{c}|\theta) &= \prod_{n=1}^{N}\prod_{k=1}^{K}\theta^{c_n(k)} \\ p(\theta) &= C(\alpha)\prod_{k=1}^{K}\theta^{(\alpha_k-1)} \end{aligned} \tag{3.12}$$

with $C(\alpha)$ the normalization constant of a Dirichlet distribution of parameter α (see [Bis06], p. 687).

We then consider a variational distribution $q(\boldsymbol{c},\theta)$ for the hidden variables $\boldsymbol{c}$ and θ that factorizes, i.e., assumes independence, as:

$$q(\boldsymbol{c},\theta) = q(\boldsymbol{c})q(\theta). \tag{3.13}$$

Following [Bis06], we derive the update equation for $q(\mathbf{c})$:

$$\begin{aligned} \log q^*(\boldsymbol{c}) &= E_{q(\theta)}[\log p(\boldsymbol{x},\boldsymbol{c},\theta)] + \text{const} \\ &= E_{q(\theta)}[\log p(\boldsymbol{x}|\boldsymbol{c})] + E_{q(\theta)}[\log p(\boldsymbol{c}|\theta)] + E_{q(\theta)}[\log p(\theta)] + \text{const} \\ &= \sum_{n=1}^{N}\sum_{k=1}^{K} c_n(k)\log\Big(f_k(x_n)\exp(E_{q(\theta)}[\log\theta_k])\Big) + \text{const}, \end{aligned} \tag{3.14}$$

with $E_q[x]$ the expectation of random variable x under the probability distribution $q(x)$. In Eq. 3.14 we have absorbed the $E_{q(\theta)}[\log p(\theta)]$ into the constant since it is independent of $\boldsymbol{c}$. Taking the exponent of both sides of Eq. 3.14 and normalizing provides:

$$q(\boldsymbol{c}) = \prod_{n=1}^{N} \prod_{k=1}^{K} r_{nk}^{c_n(k)}, \tag{3.15}$$

where we defined the responsibilities:

$$r_{nk} = \frac{f_k(x_n) \exp(E_{q(\theta)}[\log \theta_k])}{\sum_k f_k(x_n) \exp(E_{q(\theta)}[\log \theta_k])}. \tag{3.16}$$

Eq. 3.15 shows that the variational density factorizes into N independent multinomial distributions, one for each term $\boldsymbol{c_n}$. The parameters of each multinomial $q(\boldsymbol{c_n})$ are the responsibilities $(r_{n1}, r_{n2}, \ldots, r_{nK})$ in Eq. 3.16.

Similarly, for the variational distribution $q(\theta)$, we have the update equation:

$$\begin{aligned}
\log q^*(\theta) &= E_{q(\boldsymbol{c})}[\log p(\boldsymbol{x}|\boldsymbol{c})] + E_{q(\boldsymbol{c})}[\log p(\boldsymbol{c}|\theta)] + E_{q(\boldsymbol{c})}[\log p(\theta)] + \text{const} \\
&= \textstyle\sum_{n=1}^{N} \sum_{k=1}^{K} E_{q(\boldsymbol{c})}[c_n(k)] \log \theta_k + \log p(\theta) + \text{const} \\
&= \textstyle\sum_{k=1}^{K} \left(\alpha_k + \sum_{n=1}^{N} r_{nk} - 1\right) \log \theta_k + \text{const},
\end{aligned} \tag{3.17}$$

taking the exponent and normalizing yields:

$$q(\theta) = C(\gamma) \prod_{k=1}^{K} \theta^{(\gamma_k - 1)} \tag{3.18}$$

which implies that $q(\theta)$ is a Dirichlet distribution with parameters

$$\gamma_k = \alpha_k + \sum_n r_{nk} = \alpha_k + R_k, \tag{3.19}$$

where R_k represents the total responsibility for cluster k.

Note that we did not assume any particular functional form for $q(\boldsymbol{c})$ and $q(\theta)$. Instead, Eq. 3.15 and Eq. 3.18 follow from the graphical structure and the distributions used in the model, as well as from the factorized form $q(\theta, \boldsymbol{c}) = q(\theta)q(\boldsymbol{c})$. Finally, since $q(\theta)$ is a Dirichlet distribution, we can derive a closed-form solution for $E_{q(\theta)}[\log p(\theta_k)] = \Psi(\gamma_k) - \Psi(\sum_k \gamma_k)$, where $\Psi(a)$ is the first derivative of $\log \Gamma(a)$ (see [BNJ03] for the details of the derivation). This expression is then used to compute the responsibilities in Eq. 3.16.

3.4.3 Kernel density estimation of f_k

The above derivation requires knowing the density functions $f_k(x)$, however this information is not actually available when performing clustering. Following the previous approach we can use Kernel Density Estimation (KDE) to obtain an approximation $\hat{f}_k(x)$ of each unknown $f_k(x)$ if they are sufficiently smooth [Was06].

Given a kernel function $K_\sigma(x_i, x_j)$ which measures the affinity A_{ij} between a pair of points (i.e., how much we believe the two points originated from the same process when all we know is their coordinates x_i and x_j) and a set of N_k points drawn from the unknown

distribution $f_k(x)$, the kernel density estimator of $f_k(x)$ is defined as:

$$\hat{f}_k(x) = \frac{1}{N_k}\sum_{j=1}^{N_k} K_{\sigma_j}(x, x_j) = \frac{1}{N_k}\sum_{j=1}^{N_k} \frac{1}{(2\pi\sigma_j^2)^{D/2}} e^{-\frac{||x-x_j||^2}{2\sigma_j^2}} \tag{3.20}$$

where for the sake of concreteness the kernel function K_σ is defined here as the exponent with local scale σ_j set according to analysis of local statistics as suggested in [ZMP05].

Since we have the variational distribution $q(\boldsymbol{c}) = \prod_n q(c_n)$, rather than an assignment $\boldsymbol{c}$ of observations to clusters, we can redefine the kernel density estimator as the expected value with respect $q(\boldsymbol{c})$:

$$\hat{f}_k(x) = E_{q(\boldsymbol{c})}\left[\frac{1}{N_k}\sum_{n=1}^{N} c_n(k) K_{\sigma_n}(x, x_n)\right] = \frac{1}{R_k}\sum_{n=1}^{N} r_{nk} K_{\sigma_j}(x, x_n), \tag{3.21}$$

where we used the expected value of a multinomial density $E_{q(c_n)}[c_n(k)] = r_{nk}$.

Alternatively, we can obtain an assignment $\boldsymbol{c_{MAP}}$ by imposing $c_n(k) = \text{argmax}_k q(c_n)$, and compute the usual kernel density estimator of Eq. 3.20. In Sections 3.4.3.1 and 3.4.3.2 we show how these two different approximations of $f_k(x)$ relate to spectral clustering [SM00, NJW01] and kernel k-means [SSM98, DGK04], respectively.

Using Eq. 3.21 together with Eq. 3.15 and Eq. 3.18 we obtain a system of coupled equations that can be iteratively solved as described in the algorithm of Fig. 3.4.3.

Note that using a kernel density approximation is not coherent with the Bayesian framework used in deriving the variational distributions $q(\boldsymbol{c})$ and $q(\theta)$. Additionally, while the updates of Eq. 3.16 and Eq. 3.18 converge due to convexity of the variational problem (see [Bis06]), changing the approximated densities $\hat{f}_k(x)$ at each step might result in a non-

Algorithm

1. Randomly initialize the responsibilities r_{nk}
2. For $i = 1, \ldots, N$:
   - **a.** For each k, compute $\hat{f}_k$ using Eq. 3.21
   - **b.** For each k, compute r_{nk} using Eq. 3.16
   - **c.** Compute the parameters γ of Eq. 3.18
3. Repeat Step 2 until convergence or until some stopping criteria has been reached.
4. For each observation assign the cluster label c_n using the MAP of $q(\boldsymbol{c})$.

convex problem. Nevertheless, we observed that this did not seem to affect the results much, and convergence of the inference algorithm has been empirically verified. It would be interesting to study a theoretical analysis of its convergence, and possibly a Bayesian derivation of the approximation $\hat{f}_k(x)$.

3.4.3.1 Connection to spectral clustering

Let A be an $N \times N$ affinity matrix such that $A_{ij} = K_\sigma(x_i, x_j)$, and R be a $N \times K$ matrix such that its elements are the responsibilities $R_{nk} = r_{nk}$ defined in Eq. 3.16. Finally, let B be a diagonal matrix of dimension K with $B_{kk} = \exp(\Psi(\gamma_k))/\sum_n R_{nk}$.

Plugging the approximated densities $\hat{f}_k(x)$ of Eq. 3.21 into Eq. 3.16 provides:

$$R_{nk} = \frac{f_k(n)\exp(\Psi(\gamma_k))}{\sum_k f_k(n)\exp(\Psi(\gamma_k))} = \frac{(\sum_j A_{nj}R_{jk})B_{kk}}{d_n} \tag{3.22}$$

where $d_n = \sum_k R_{nk}$. If we impose $\alpha_k = 0\ \forall k$, we get $B_{kk} = \exp(\Psi(\sum_n R_{nk})/\sum_n R_{nk}$, which for large values of N is almost one[2] and can be removed from Eq. 3.22. In this

[2]This is a good approximation for $N > 100$.

case we have that $d_n = \sum_j A_{nj}$, and we can write a recursive matrix equation for the the responsibilities: $R^{t+1} = D^{-1}AR^t$, with D a diagonal matrix of dimension N and diagonal elements $D_{nn} = d_n$.

This recursive equation is similar to the power method for computing the eigenvectors of the matrix A [GL91] with the difference that in the case of the power method the *columns* of R are forced to be orthonormal, while in our case we force the *rows* of R to be normalized to 1.

3.4.3.2 Connection to kernel k-means

The variational inference method is also similar to kernel k-means where, instead of the expected KDE of Eq. 3.21, we consider the MAP assignment $\boldsymbol{c_{MAP}}$ and compute the usual kernel density estimator. Following [DGK04], let $\phi(x)$ be a function that maps observations in a feature space $\mathcal{F}$, such that inner product in $\mathcal{F}$ is defined as $\phi(x_i)^T\phi(x_j) = K(x_i, x_j)$. In feature space $\mathcal{F}$ each iteration of the k-means algorithm consists of two steps:

1 For each point $\phi(x_n)$ select a new cluster label $c_n = \text{argmin}_k||\phi(x_n) - \boldsymbol{\mu}_k||^2$.

2 Compute the new cluster center $\boldsymbol{\mu}_k = \frac{1}{N_k}\sum_{n\in C_k}\phi(x_n)$, where N_k is the number of points in cluster C_k.

Substituting the inner product in $\mathcal{F}$ with kernel operations we get:

$$\begin{aligned} ||\phi(x_n) - \boldsymbol{\mu}_k||^2 &= \phi(x_n)^T\phi(x_n) - \tfrac{2}{N_k}\textstyle\sum_{m\in C_k}\phi(x_n)^T\phi(x_m) + \tfrac{1}{N_k^2}\sum_{m,l\in C_k}\phi(x_m)^T\phi(x_l) \\ &= K(x_n, x_n) - \tfrac{2}{N_k}\textstyle\sum_{m\in C_k}K(x_n, x_m) + \tfrac{1}{N_k^2}\sum_{m,l\in C_k}K(x_l, x_m). \end{aligned}$$

The first term $K(x_n, x_n)$ can be omitted from the computation because it does not depend on k. The second term equals minus the approximated densities $\hat{f}_k(x_n)$, and the third term is a cluster-specific quantity (independent of k). We have observed experimentally that this cluster-specific term was irrelevant when computing $c_n = \text{argmin}_k ||\phi(x_n) - \boldsymbol{\mu}_k||^2$. Therefore, we can omit it as well. This implies that step 1 of the k-means algorithm above can be written as:

$$c_n = \text{argmin}_k ||\phi(x_n) - \boldsymbol{\mu}_k||^2 = \text{argmax}_k \frac{1}{N_k} \sum_{m \in C_k} K(x_n, x_m). \tag{3.23}$$

Whenever the number of observations is roughly the same in each cluster, then the term $E_{q(\theta)}[\log p(\theta_k)] = \Psi(\gamma_k) - \Psi(\sum_k \gamma_k)$ is independent of k and the responsibilities r_{nk} in Eq. 3.16 are just proportional to $\hat{f}_k(x_n)$. We conclude that selecting the MAP assignment of $q(\boldsymbol{c})$, as explained in 3.4.3, is equivalent to computing the first step of the kernel k-means[3].

3.5 Experiments

Experiments for the image segmentation model of Fig. 3.1 were performed on two image datasets. The first is a set of 100 images of Egrets [LSP05] where only gray level values and pixel coordinates were used to compute affinities $A_{ij} = K(x_i, x_j)$ (see Section 3.2.1). The second is a set of 16 general color images, where the RGB values and the pixel coordinates were used to compute affinities. Fig. 3.5 shows a few representative image segmentation results. Unless otherwise stated, in all the following experiments the sampling algorithm

[3]The second step of the algorithm is redundant since the term $\frac{1}{N_k^2} \sum_{m,l:c_{m,l}=k} K(x_l, x_m)$ is not important in deciding the cluster assignment.

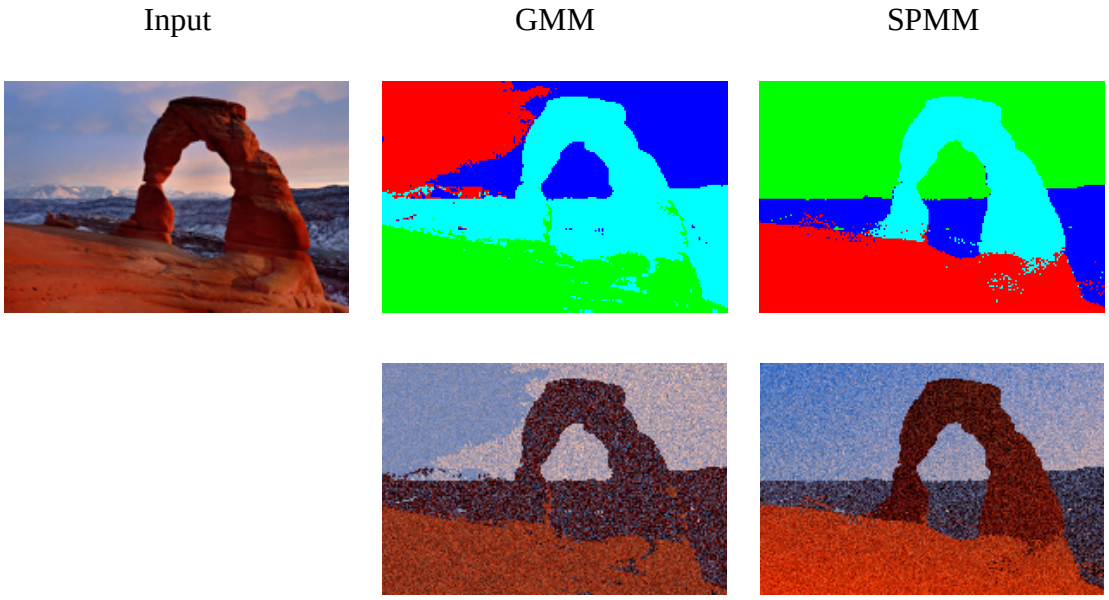


Figure 3.7: Comparison between the Gaussian mixture model (GMM) and the semi-parametric mixture model (SPMM) of Section 3.2.3. The colors of the sky segment are not well modeled by a unimodal distribution: the left part has a more uniform color than the right part, where some clouds are present. The GMM segmentation (center) splits the sky into two components, while the semi-parametric segmentation (right) correctly assigns the sky to a single segment. Fig. 3.8 shows the observations in each segment projected on different coordinate planes of the xy-RGB feature space. The bottom row shows a sample image from the estimated segmentations from the GMM model (center) and from the semi-parametric model (right).

has been used to perform inference.

Fig. 3.6 compares the quality of our results with the state-of-the-art on both datasets. The performance of fitting a Gaussian mixture model (GMM) is of the lowest quality, because Gaussian "blobs" poorly approximate the image segments in xy-RGB space. The results for normalized cut and our semi-parametric mixture model (SPMM) are comparable, with slight preference to our method. The SPMM, as well as GMM, naturally provides soft assignment of pixels to segments (see Fig. 3.5 columns 3 and 6). Such soft assignments often make more sense, e.g., in ambiguous cases where the transition between segments is gradual. Furthermore, they provide more information than hard decisions do. An attempt at

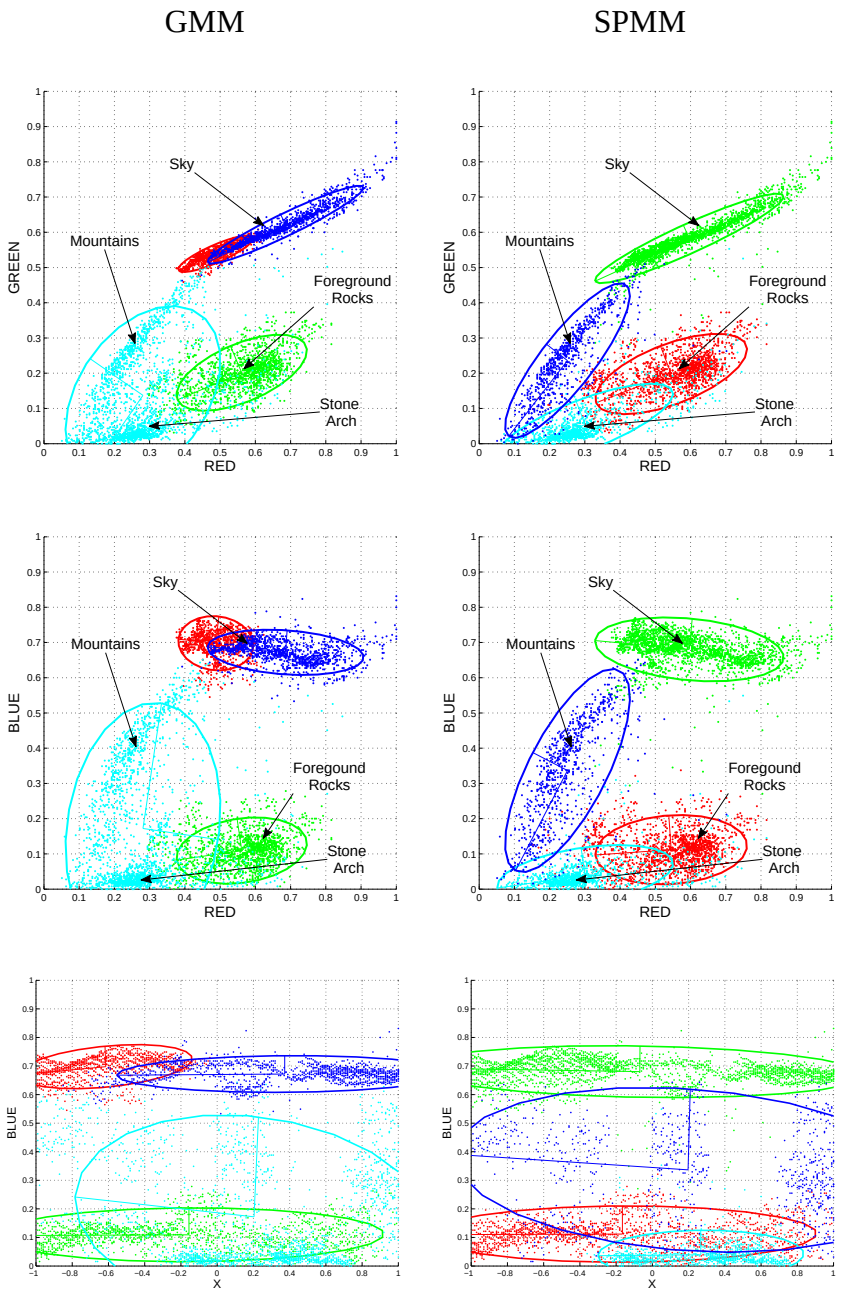


Figure 3.8: Comparison between the different segmentations in Fig. 3.7. Each plot shows different coordinate planes of the xy-RGB feature space. The left column refers to the GMM segmentation the right column to the SPMM one. The points correspond to the projections of the image pixels. The ellipses represent Gaussian distributions (the parametric term for the SPMM). The colors of points and ellipses correspond to the segments in Fig. 3.7.

obtaining soft assignments from normalized cuts was proposed in [JDK05]. This approach however, lacks a complete probabilistic interpretation.

Fig. 3.7 and Fig. 3.8 show an experimental comparison between the two probabilistic models we are considering. To better understand the properties of the semi-parametric mixture model (SPMM) presented in Section 3.2.3, as well as its potential advantages over the Gaussian mixture model (GMM), we analyze a specific example in detail. The image we chose, on the left of Fig. 3.7, presents a number of challenges for any segmentation algorithm: it has an object of complex shape (the stone arch), a sky partially covered with clouds with color changing quickly from deep blue (left part of the image) to veiled whitish blue (right part of the image), and complex texture regions (the mountains in the background).

Examining the segmentation results, we see that the GMM model (center) failed to identify the sky as a single segment, but rather divided it in two parts. The left part without clouds is assigned to the red segment, while the right part where clouds are present is assigned to the blue segment. In the left column of Fig. 3.8 we can see the projections on different coordinate planes of the observations in each segment of the GMM segmentation. We see that pixels in the red segment (in red) and pixels in the blue segment (in blue) fall in two different but contiguous elliptic clusters (see RED/BLUE and X/BLUE projections on the second and third rows). This is a consequence of the multimodal shape of the distribution of the sky segment in the xy-RGB space. Finally, since only four segments are used, the mountains on the background and the stone arch are grouped into a single segment (cyan).

On the other hand, considering the segmentation results of the semi-parametric mixture

model (SPMM) (Fig. 3.7 right), we see that it identifies the sky region as a single segment (green). This is due to the non-parametric term in Eq. 3.5 which allows the model to take advantage of the local proximity of the two modes of the sky distribution (see right column of Fig. 3.8). It is also interesting to observe how the parametric term captured the global color of the sky resulting in assigning the sky label (green) also to the portion of sky under the stone arch. The SPMM method also correctly segments the arch as a single object (cyan).

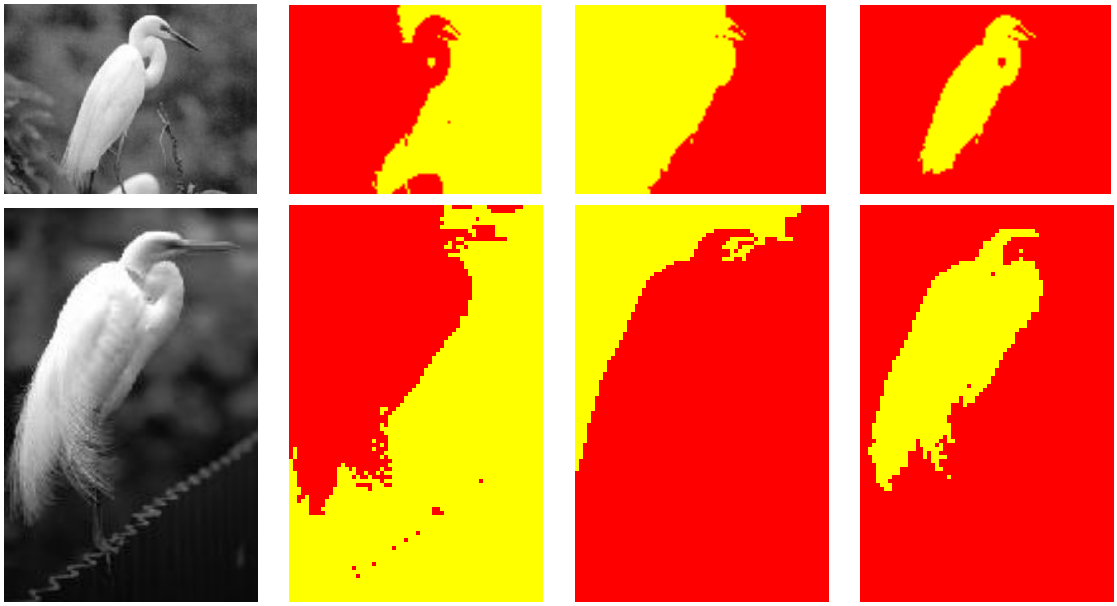
Finally, Fig. 3.9 shows a qualitative comparison between the sampling inference algorithms and the variational approximation method on images from the bird dataset. We observe that both algorithms are capable of extracting the bird in the images, with the variational approximation faster by a factor 5 than the Gibbs sampler.

3.6 Partial labeling

While our general framework is unsupervised, some partial information on the assignment of points to clusters is often available. Such information can be provided in one of three forms: partial labeling, "must-link" constraints, and "cannot-link" constraints. We next explore all three.

Partial assignment of points to clusters is equivalent to having observed the labels of some of the (usually hidden) random variables c_i of the model. Such type of constraints are thus incorporated by fixing the corresponding observed labels c_i during the inference process on the model (described in Section 3.4). This leads to a more stable solution and faster convergence. Figure 3.10 shows how minimal partial labeling can significantly im-

Figure 3.9: Comparison of the Gibbs-sampler and variational inference methods for the image segmentation problem. The first column shows the original images, the second one shows the segmentation results of the Gibbs sampler used in [AZMP07], and the third one



(a) (b) (c) (d)

Figure 3.10: Partial labeling. A typical result of intensity-based image segmentation into 2 clusters (out of 100 images in the Egret set of [LSP05]). (a) Original image, (b) GMM-EM clustering, (c) normalized cuts, (d) our result with partial labeling. Boundary pixels were constrained to the background cluster.

prove image segmentation results. The segmentations obtained by our method are of higher quality than those of GMM-EM (using the same constraints). Comparison to spectral factorization is impossible since labels cannot be fixed. We thus compare our results to those of graph-cuts methods. Graph-cuts [RKB04] are somewhat similar in spirit to spectral factorization but require significant user interaction and are thus generally of less interest to us. Fig. 3.11 shows our approach provides comparable results to those of Rother et al. [RKB04] when the same amount of user intervention is utilized.

Constraints which force points to reside in the same cluster ("must-link") can be incorporated by estimating the labels of those points jointly. This corresponds to a modification of the model of Fig. 3.1 where an edge (conditional dependence) is added between the constrained points. The "cannot-link" constraints can (in theory) be incorporated, in a similar

Figure 3.11: Partial labeling, comparison with GrabCut. Left: input image. Right: our segmentation result, obtained by manually labeling part of the image as background. Refer to [RKB04] for the corresponding GrabCut segmentation.

manner, by estimating the labels for these points jointly while enforcing exclusion. While in our inference method this is easily achievable if the "cannot-link" constraints involve only pairs of separated points, it is difficult to consider exclusion dependencies over a larger number of points, since the number of possible assignments would grow exponentially.

Incorporating labeling constraints (of any type) is not trivial in non-probabilistic methods such as spectral clustering. Yu and Shi [YS04] showed how "must-link" constraints on pairs of points can be incorporated, albeit with some additional computational cost. It has not been shown how to incorporate "cannot-link" constraints or partial labeling in spectral clustering.

Chapter 4

Video Segmentation

4.1 Temporal coherence in videos

In the previous sections we evaluated performance in the unsupervised and partially supervised cases. But other types of prior information are often available. In this section we examine segmentation of video frames. Adjacent video frames are known to be highly correlated regardless of their content. In this section we show how this can be incorporated into our segmentation framework and improve segmentation quality. A related idea was proposed by Jojic and Frey [JF01] who separated video frames into layered sprites. Their underlying assumption was that all layers are shared among the video frames and each layer can undergo only limited transformations such as translation and occlusion. This does not apply to general videos where the camera moves significantly, resulting in large changes in background, as well as complex motion of articulated objects, such as human bodies, which imply large changes in appearance and shape across video frames. We thus propose an approach that assumes coherence only across consecutive frames and not throughout the sequence.

Pixel-level segmentation of video sequences is a high-dimensional problem, since the

data-set size equals the overall number of pixels. Therefore, one has to resort to segmenting separately small portions of the video. We will assume here the video portions are individual frames. This can result in a set of independent segmentations even for consecutive frames which are highly correlated. To obtain a globally consistent segmentation one needs to enforce spatiotemporal coherence across frames. This can be done by first segmenting each frame independently and afterwards matching segments across frames. Alternatively, coherence could be enforced directly during the segmentation task. The latter is impossible for methods like spectral clustering, which do not allow incorporating prior information.

On the contrary, our framework is particularly suitable for this purpose. We segment videos frame-by-frame while propagating information from one frame to the next. We initialize the segmentation of each frame with the segmentation result of the previous frame. Since consecutive frames are highly correlated, this on its own speeds up the computation (by reducing the number of iterations of the sampler) and promotes more consistent results. Furthermore, since our clustering provides cluster assignment probabilities for each pixel, we detect high confidence pixels and fix their labels for some iterations. This constrains the segmentation of each frame to be highly similar to that of its predecessor. We then release the labels of all pixels and collect samples. This procedure localizes slowly changing parts of the video, such as the background, and reduces the computational cost by speeding up convergence.

4.2 Video segmentation algorithm

Following is a short summary of the proposed video segmentation approach:

1. Segment the first frame of the sequence and obtain cluster assignment probabilities for each pixel.

2. For all the remaining frames $f = 2, \ldots, F$:

   a. For frame $f-1$, compute the confidence R_i of segment assignment of the ith pixel as: $R_i = (p(c_i = k_v|\boldsymbol{x}) - p(c_i = k_w|\boldsymbol{x})/p(c_i = k_v|\boldsymbol{x})$, where $p(c_i = k_v|\boldsymbol{x})$ and $p(c_i = k_w|\boldsymbol{x})$ are the highest and the second-highest cluster assignment probabilities for pixel i.

   b. Initialize the sampler for frame f with cluster assignment and confidence weights of frame $f - 1$.

   c. Run the sampler for N_1 iterations while fixing the labels of the high confidence pixels, $R_i > 0.9$.

   d. Run the sampler for further N_1 iterations with all labels free to change, and collect samples.

   e. Set cluster assignment of frame f as MAP estimator and keep cluster assignment probabilities.

Even though this is a very simple way to impose temporal coherence, the previous algorithm still shows that higher-level information can significantly improve the quality of segmentation. Using more complex (possibly probabilistic) models for the motion of the object in the video is likely to further improve the segmentation results.

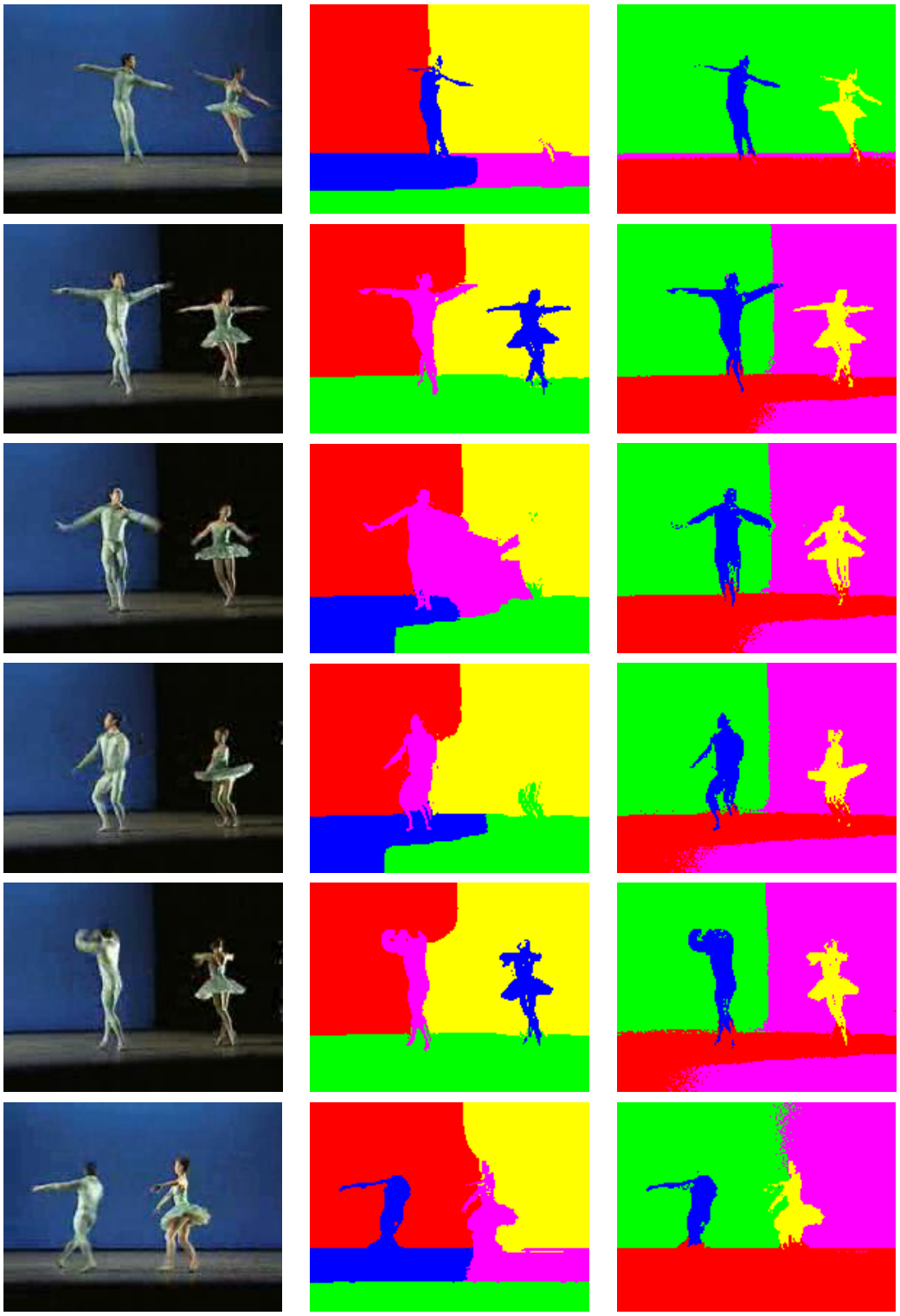


Figure 4.1: Video sequence segmentation. Left column: Frames 218, 280, 282, 284, 286, and 329 out of a 343-frame-long video. Middle column: normalized cut segmentation results. Right column: SPMM result while enforcing spatiotemporal coherence across frames is significantly better. See Fig. 4.3 for human rating of the segmentation results. The complete video as well as results on a different video are provided in the supplemental material.

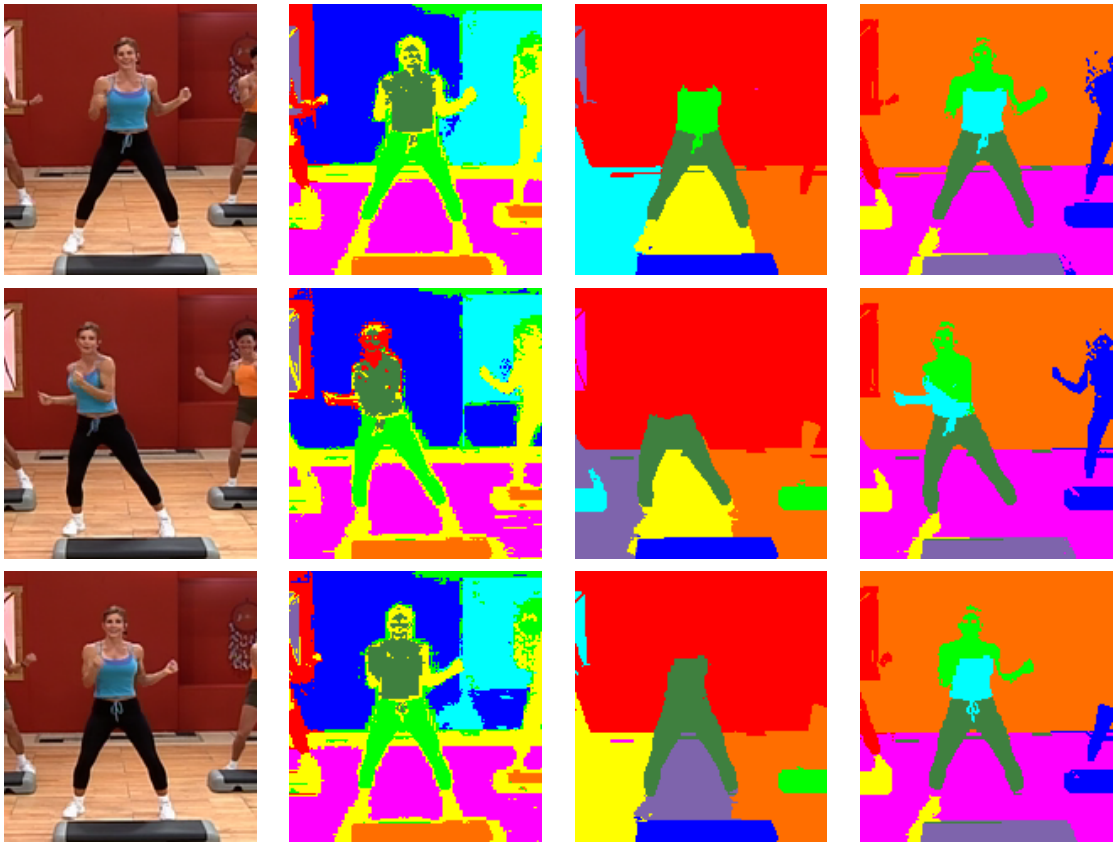


Figure 4.2: Another video sequence segmentation. First column: Frames 61, 136, and 154 out of a 193-frame-long video. Second column: GMM segmentation results. Third column: normalized cut segmentation results. Right column: SPMM result while enforcing spatiotemporal coherence across frames is significantly better. The complete video as well as results on a different video are provided in the supplemental material.

4.3 Experimental results

Fig. 4.1 and Fig. 4.2 compare the results of the proposed approach with those of normalized cuts with post-segmentation segment matching. The segmentation obtained by normalized cuts is inconsistent across frames. Our method significantly outperforms both normalized cuts and GMM-EM[1] and returns video segmentations that are both of high quality and

[1]The GMM-EM model we use for comparison in our experiment is closely related to the model of Khan and Shah [KS01], with the main difference that no information of local velocity is used in the clustering. The segmentation obtained by GMM-EM in our comparisons is consistent across frames but is of poor quality due to the complex shapes of the segments.

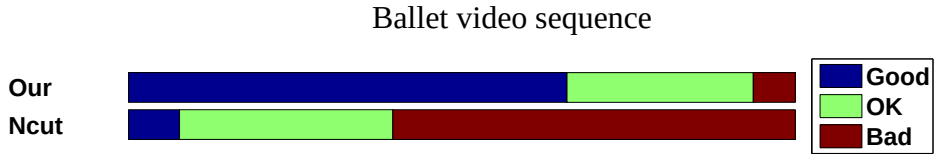


Figure 4.3: Human Ratings. Six people rated the video segmentation results of a subset of all the frames in the "ballet" sequence. As for the results in Section 3.5) the possible rates were: good, OK, or bad. The plots show the rating statistics for the SPMM with video coherence (top bar) and for the normalized cut (bottom bar). Each bar is split into three parts whose sizes correspond to the fraction of images assigned to the corresponding rating. Better overall performance corresponds to less red and more blue. Our method outperforms clearly outperforms normalized cut.

consistent across frames (i.e. the same object is consistently assigned to the same cluster, denoted by same color, throughout the whole video sequences). Fig. 4.3 shows the human ratings for the ballet sequence (see Fig. 4.1). For this quantitative assessment of segmentation quality, the SPMM greatly outperform the normalized cut method[2].

For sanity check, we also compared segmentation results of our method with and without temporal coherence. Using temporal coherence significantly improved the segmentation quality. Please refer to supplemental material of [AZMP07] for the complete video sequence as well as other videos.

[2]For the video sequence of Fig. 4.1 the GMM-EM method fails to converge. Therefore, no human ratings is available

Chapter 5

Segmenting Image Collections

We can extend the probabilistic model of Chapter 3 for the simultaneous segmentation of an image collection. When all the images in the collection share objects that have similar characteristics (see Fig. 5.2, top row) we can improve the segmentation by sharing information across images. For example, in Fig. 5.2, since all the pictures show a person's head (and shoulders), it is possible to use the consistency of these elements' appearance (color, shape, position) across images to improve segmentation quality, as well as provide coherent segment labels across images.

5.1 Semi-parametric LDA model (SP-LDA)

Hence, we propose the new probabilistic model of Fig. 5.1, where K segments are shared across a collection of M images. These shared segments are described by the distributions f_k^s, with k the segment label and the superscript s indicating the distribution is "shared". We also assume that each image has H additional segments that are not shared across the collection. These image-specific segments are described by the distributions $f_{h,m}^{ns}$ where h indicates the segment label in its image and m is the image identifier in the collection.

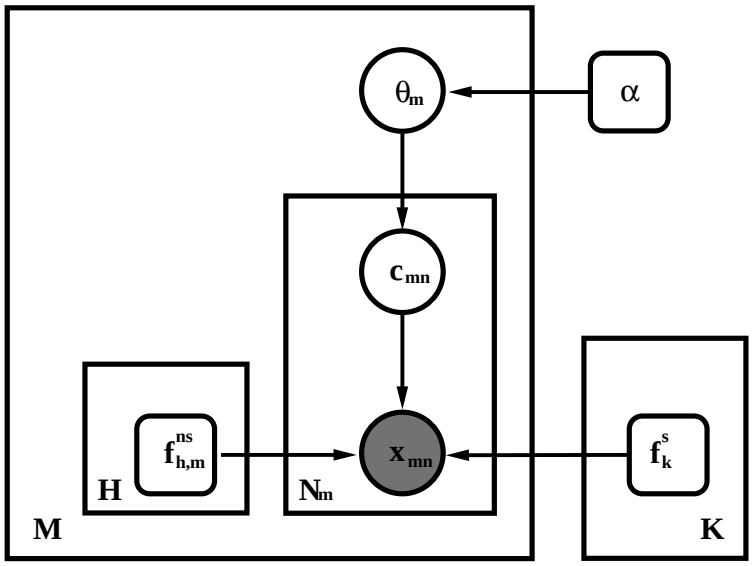


Figure 5.1: Semi-parametric Latent Dirichlet Allocation model (SP-LDA) for joint segmentation of image collections (see Section 5.1). As in Fig. 3.1, the gray node x_{mn} represents the observed quantities (features vector n for image m in the collection). The node c_{mn} represents the segment assignment for the observation x_{mn}. The node θ_m represents the mixing coefficients for each segment in image m. The rounded box α is the hyperparameter of the Dirichlet distribution of θ_m. The inner plate represents the N_m pixels in image m, while the outer plate represents all the M images in the collection. The K distributions f_k^s model the recurring objects in the collection and are shared across all the images. The H distributions $f_{h,m}^{ns}$ are local to each image, i.e., independent of the rest of the collection, and represent the image-specific segments.

Since these distributions are not shared across images we use the the superscript ns for them. Given K and H the total number of segment in each image is $K + H$. If we set the number of shared segments K to zero we obtain the single image case, while if H is set to zero then we are enforcing all the segments in an image to be shared in the collection; in Section 5.2 we will explore the effect of different choices.

We represent both the shared distributions f_k^s and the image-specific ones $f_{l,m}^{ns}$ using the semi-parametric representation described in Section 3.2.3. We call the probabilistic model of Fig. 5.1 with the semi-parametric representation *semi-parametric latent Dirichlet allocation (SP-LDA)*. For the shared distributions f_k^s, the parametric term captures the

information that is consistent across the image collection, such as the shape and position of the recurring object and its color. The non-parametric term of the the distributions f_k^s is still image-specific. As discussed in Section 3.2.3 we can think of the parametric term as providing a prior or bias toward a particular region of the feature space (the position and color of pixels segments). This bias represents appearance and shape properties of the common objects in all the images.

To perform inference, we use the sampling method developed for the single-image case (see Section 3.4.1), with the exception that the parameters of the Gaussian terms of shared segments are computed using observations from all the images. The non-parametric terms of the shared segments are computed independently for each image as for the single-image algorithm.

5.2 Experiments

To study the performance of the SP-LDA model of Fig. 5.1 we consider a collection of 30 images, all showing the face (and the shoulders) of different people in different indoor scenes (varying background). To determine which parts of the image are assigned to a shared segment and which parts to a not-shared segment, we test different values of K (number of shared segments) and H (number of image-specific segments).

Fig. 5.2 shows six images from the collection (first row), their ground truth segmentation (second row)[1] of the face (blue segment), and several segmentation results for different values of H and K. When no information is shared among the images (third and

[1]The ground truth considers only the face and disregards other parts of the person like the neck and the shoulders.

Figure 5.2: Segmenting an image collection. First row: six examples out of a collection of 30 images of faces on different backgrounds. Second row: corresponding ground truth segmentation of the face. Rows three to five: binary segmentations with different numbers of shared segments. Rows six to eight: segmentation in three segments with different number of shared segments. K is the number of shared segments and H is the number of image-specific ones.

sixth rows) the resulting segmentation is not precise in selecting the face. Often it merges the face with part of the scene background, particularly when only 2 segments are used (third row). Moreover, the segment containing the face is not consistently labeled across the image (see sixth row). When one or more segments are shared across the images, they are assigned to the recurring elements of the collection: the face and the shoulders. This results in both an improvement in the segmentation of the face and a consistent labeling of

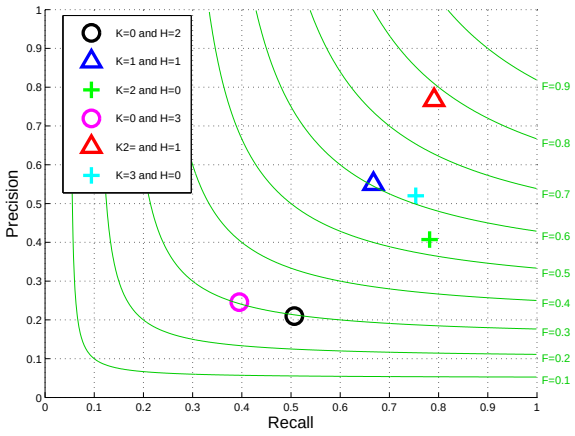


Figure 5.3: Precision/recall for the face collection. Different markers correspond to the performance of the SP-LDA model (Fig. 5.1) for different settings of the parameters K (number of shared segments) and H (number of image-specific segments). The green curves correspond to precision/recall values with the same harmonic mean (F measure [Rij79]).

the segment of a recurring object across different images. In particular, when one segment is shared and one is image-specific ($K = 1$, $H = 1$) the face and the shoulders are almost always assigned to the shared segment (yellow), while the remaining part of the scene is assigned to the image-specific segment (red) as shown in the fourth row. When there are two shared segments and an image-specific one ($K = 2$, $H = 1$) the segmentation of the face improves further. One of the shared segments captures the faces (red) and the other the shoulders (yellow), which are no longer grouped together with the face (seventh row). Again the rest of the scene is assigned to the image-specific segment (green). Finally, we observe that forcing all the segments to be shared (fifth and eighth rows) results in worse segmentation than the case with image-specific segments. This is most likely a result of the mismatch between the model, which assumes all segments are recurring, and the dataset which shows faces (a recurring object) on varying backgrounds.

The qualitative observations for Fig. 5.2 are confirmed by the precision/recall results presented in Fig. 5.3. Without sharing (i.e., setting $K = 0$) we have the lowest perfor-

mance[2] (black and magenta circles). These results are almost equivalent to a random guess, since the face will have random labels across the images. Performance improves when we share information for some segments, and one segment is image-specific. In particular the $K = 2$, $H = 1$ case gives the best results (red triangle). Finally, for a fixed number of total segments, sharing all the segments (green and cyan crosses), i.e., setting $H = 0$, always results in worse performance than keeping one segment image-specific, i.e., $H = 1$. This can be seen by comparing the positions of crosses and triangles.

The computational cost of performing inference on the model of Fig. 5.1 is linear in the number of images and in the total number of segments $K + H$ in each image. Using our C++ implementation of the sampler it takes about 185 sec. per image per segment on a 2.50GHz Intel Xeon machine.

The SP-LDA model can to handle images like the ones in Fig. 5.2. For more complex situations, with many more recurring objects that might not appear in all the images of the collection, the inference algorithm for the SP-LDA fails to converge. For this more general problem we present a new model in Section 6, that can handle variable content in images and is capable of modeling the appearance of more general categories.

[2]To decide which segment label corresponds to the face segment, we select the segment with the largest overlap with the ground truth. However, when a single segment is shared we assume that segment to correspond to the face segment.

Chapter 6

Learning Categorical Segments in Image Collections

In the SP-LDA model of Section 5 we used mean and covariance of the semi-parametric distributions as shared statistics for the position/RGB value across images. For the collection of faces we considered in our experiments this is a good modeling choice since the recurring object (the face) has similar shape and color in all the images. However, for recurring objects with textured appearance and varying position and shape, a more complex representation is required.

6.1 Modeling recurring segments

Inspired by the "bag-of-words" approach [FFP05, SRE+05] we extend the model in Fig. 3.1 by adding new observed variables w_{mn} that represent the visual words associated with an observation. These new discrete random variables are sampled from K different multinomial distributions ϕ_k (topic distributions) which model the visual words' statistics for each of the K segments. Fig. 6.1 shows the graphical representation of the extended model. The model represents a collection of M images. An image is represented by N_m regularly

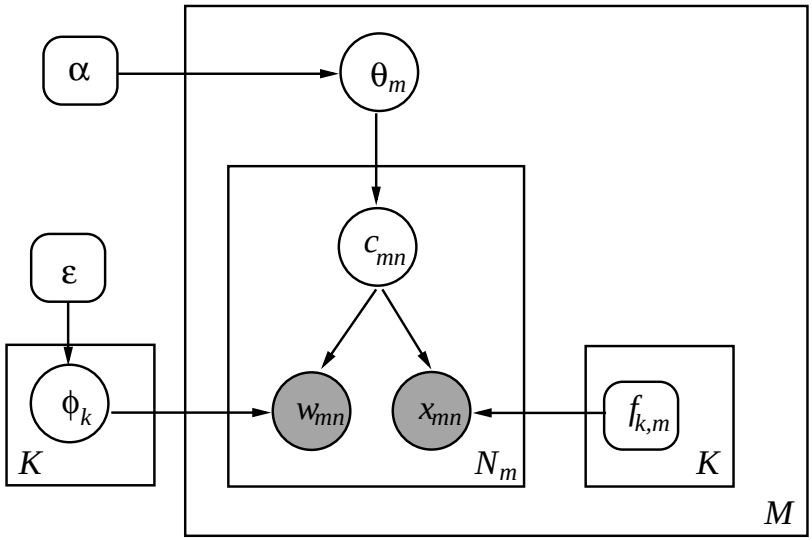


Figure 6.1: The affinity-based LDA model (A-LDA) for learning categorical segments (see Section 6). The two gray nodes x_{mn} and w_{mn} represent the observed quantities in the model: the feature vector (position and color) and the visual word associated with each pixel, respectively. The nodes c_{mn}, $f_{k,m}$, ϕ_k, and θ_m are hidden quantities that represent the segment assignment for x_{mn} and w_{mn}, the probability density of the feature vectors in segment k of image I_m, the visual words distribution for segment k, and the sizes of the segments in image m, respectively. The two squares with rounded corners α and ε represent the hyperparameters of the Dirichlet distributions over θ_m and ϕ_k, respectively. Finally, K is the number of segments, N_m is the number of pixels in image m, and M is the number of images in the collection.

spaced observations (e.g., one sample per pixel). At the n-th observation of image m we measure a feature vector x_{mn}, e.g., the pixel's position and RGB values. We further extract a fixed size image patch centered at the n-th pixel and assign to it a "visual word" w_{mn}. In our implementation, the dictionary of visual words is obtained by vector-quantizing a subset of all the descriptors of the patches extracted from all the images. The w_{mn} variable of an observation is the label of the dictionary entry closest to the descriptor associated to the observation.

Each image is formed by K regions (segments) whose visual words statistics are shared across images. Segment k in image m has a probability distribution $f_{k,m}$ of feature vector

values x_{mn}, and a probability distribution ϕ_k of the visual words w_{mn}. Note, that the distributions $f_{k,m}$ of feature vectors are not shared between images, while the distributions of visual words ϕ_k are shared across images. This is because we assume that the appearance of an object, which is captured by the ϕ_k distributions, is similar in all images. On the other hand the position of an object in a particular image can be assumed independent of the position in other images. For example, a car can appear in various image locations. However, its overall appearance, as described by the visual words, is the same in all images. We model the segment distributions $f_{k,m}$ using the nonparametric model proposed in Chapter 3, while for ϕ_k we use an LDA model, as proposed in [FFP05] and [SRE+05]. Thus if we remove the x_{mn} node from the graphical model we obtain the LDA model. Removing the w_{mn} node from the model yields a collection of M independent models, like the ones described in Chapter 3. We call this new model *affinity-based latent Dirichlet allocation (A-LDA)* since we are using the affinities between pixels (see Eq. 3.3) to describe the segment distributions $f_{k,m}$.

In the A-LDA model, visual words are grouped by segments. This enables learning topics that are related to object parts rather than to whole scenes, as is done with the "bag of words" representation of whole images [FFP05]. A key aspect of the proposed model is that the densities $f_{k,m}$ allow grouping of all the visual words generated from the corresponding topic distribution ϕ_k into a single image segment. Moreover, it is possible to enforce different grouping properties by choosing different forms for the densities $f_{k,m}$. Assuming a Gaussian distribution over the pixel positions in the image, as in Sudderth et al. [STFW05], results in a spatially elliptical cluster of visual words generated from the topic ϕ_k. Assum-

ing a non-parametric distribution (see 3.2.1), results in a more complex grouping based on color information as well as position in the image.

An important remark is that the A-LDA model assumes that the feature vectors x_{mn} and the visual words w_{mn} of a given pixel are independent given the topic assignment for the pixel c_{mn}. It also assumes that visual words are independent given their hidden labels. These two assumptions are theoretically incorrect. The two random variables w_{mn} and x_{mn} are correlated, since both depend on the image patch centered on pixel n. The same is true for the visual words of close (overlapping) patches. However, ignoring these dependencies results in a simpler probabilistic model.

The densities $f_{k,m}$ and the distributions ϕ_k have complementary roles in the model. The density $f_{k,m}$ models segment k in a specific image m, and it forces pixels with high affinity to be grouped together. The multinomials ϕ_k couple together segments in different images of the collection, i.e., they force segments in different images to have the same visual words statistics. All the multinomial coefficients of the ϕ_k are sampled from the same prior distribution — a symmetric Dirichlet distribution [BNJ03] with (scalar) parameter ε:

$$\begin{aligned} \phi_k &\sim \text{Dir}(\varepsilon) \\ w_{mn}|\phi_k &\sim \text{Multinomial}(\phi_k). \end{aligned} \tag{6.1}$$

The K topic/segment distributions are not image-specific like the densities $f_{k,m}$, but rather are shared within the entire collection. This allows coupling segment appearance statistics across multiple images based on the distribution of visual words they contain. However, in a particular image of a collection there may be objects that do not appear in other images.

To model these non-recurring elements, one can extend the model of Fig. 6.1 by forcing some of the ϕ_k to be image specific, like the $f_{k,m}$, rather than common to all the collection. This extension gives a model similar to the one of Fig. 5.1. In our experiments this extended model gives similar results to the one of Fig. 6.1.

6.2 Inference algorithms

Exact inference is impossible for the A-LDA model. Therefore we developed two types of algorithms for approximate inference. The first type is based on MCMC techniques for sampling from the posterior $p(\mathbf{c}|\mathbf{x}, \mathbf{w})$. The second type is based on variational approximation of the intractable posterior.

6.2.1 Sampling-based inference

For the sampling method we propose two different type of procedures: the first one is a Gibbs sampler [GG84], while the second one is based on the more general Metropolis-Hasting [MRR$^+$53, Has70] method (the Gibbs sampler is a special case of Metropolis-Hasting). We need two different sampling strategies to overcome the limitation of the Gibbs sampler.

6.2.1.1 Gibbs sampling

To estimate the posterior distribution $p(\mathbf{c}|\mathbf{x}, \mathbf{w})$ we can extend the Gibbs sampling algorithm previously presented. Let $p(c_{mn}|\mathbf{c}_{-mn}, \mathbf{x}, \mathbf{w})$ be the posterior distribution of the hidden segment label c_{mn} of the n'th pixel in image m, given the class labels $\mathbf{c}_{-mn}$ of all the

other pixels in all the other images, all the feature vectors $\mathbf{x}$, and all the visual words $\mathbf{w}$. This yields:

$$p(c_{mn} = k|\mathbf{c}_{-mn}, \mathbf{x}, \mathbf{w}) \propto p(x_{mn}, w_{mn}|c_{mn} = k, \mathbf{x}_{-mn}, \mathbf{w}_{-mn}, \mathbf{c}_{-mn})p(c_{mn}|\mathbf{c}_{-mn}). \quad (6.2)$$

In our model the feature vector x_{mn} and visual word w_{mn} are assumed to be independent given the segment label c_{mn}. We can, therefore, decompose the likelihood term as the product:

$$\begin{aligned} &p(x_{mn}, w_{mn}|c_{mn} = k, \mathbf{x}_{-mn}, \mathbf{w}_{-mn}, \mathbf{c}_{-mn}) \quad (6.3)\\ &= p(x_{mn}|c_{mn} = k, \mathbf{x}_{-mn}, \mathbf{c}_{-mn})p(w_{mn}|c_{mn} = k, \mathbf{w}_{-mn}, \mathbf{c}_{-mn}). \end{aligned}$$

The first term of Eq. 6.4 is the likelihood of the feature vector x_{mn} to be in the k-th segment of image m. Using the non-parametric approximation of Eq. 3.3, this term can be expressed as:

$$p(x_{mn}|c_{mn} = k, \mathbf{x}_{-mn}, \mathbf{c}_{mn}) = \hat{f}_{k,m}(x) = \frac{1}{N_{k,m}} \sum_{j \in S_{k,m}} K(x_{mn}, x_{mj}) \quad (6.4)$$

where the kernel values $K(x_{mn}, x_{mj}) = A^m_{nj}$ represent the affinity between x_{mn}, and x_{mj}, $S_{k,m}$ is the set of feature vectors in segment k in image m, excluding the vector n, and $N_{k,m}$ is the cardinality of segment $S_{k,m}$.

The second term of Eq. 6.4 is the likelihood of the visual word w_{mn} to belong to the topic distribution ϕ_k. Given the conjugate prior over ϕ_k (see Eq. 6.1) we obtain:

$$p(w_{mn}, |c_{mn} = k, \mathbf{w}_{-mn}, \mathbf{c}_{-mn}) = \frac{N_{w_{mn},k} + \varepsilon}{N_k + \varepsilon V}, \quad (6.5)$$

where $N_{w_{mn},k}$ is the number of pixels with visual word w_{mn} assigned to segment k in all the images of the collection, N_k is the total number of observations assigned to segment k, and ε is the hyperparameter of the Dirichlet prior over the topic distributions ϕ_k's.

As in Chapter 3.4.1, the prior term of Eq. 6.2 can be written as:

$$p(c_{mn} = k|\mathbf{c}_{-mn}) = \frac{N_{k,m} + \alpha_k}{(N_m - 1) + \sum_k \alpha_k}, \tag{6.6}$$

where $N_{k,m}$ is the cardinality of segment S_k in image m, N_m is the number of pixels in image m, and α_k are the hyperparameters of the Dirichlet prior over θ_m.

Combining Eq. 6.4, Eq. 6.5, and Eq. 6.6, we obtain the following expression for the conditional probabilities used in the Gibbs sampling:

$$p(c_{mn} = k|\mathbf{x}, \mathbf{w}, \mathbf{c}_{-mn}) \propto \left(\frac{1}{N_{k,m}} \sum_{j \in S_{k,m}} K(x_{mn}, x_{mj})\right) \left(\frac{N_{w_{mn},k} + \varepsilon}{N_k + \varepsilon V}\right) \left(\frac{N_{k,m} + \alpha_k}{(N_m - 1) + \sum_k \alpha_k}\right). \tag{6.7}$$

All the quantities in Eq. 6.8 can either be precomputed, like the affinities $K(x_i, x_j) = A_{ij}$, or updated very efficiently. Given the samples from $p(\mathbf{c}|\mathbf{x}, \mathbf{w})$ by Gibbs sampling, it is possible to assign each pixel to a segment using the MAP estimator. The segment distributions $f_{k,m}$ and the topic distributions ϕ_k can be estimated given the assignment.

6.2.1.2 Block sampler

The Gibbs sampler is easy to derive and implement. It is computational efficient to obtain new samples since by construction the algorithm accepts all the samples it generates (as

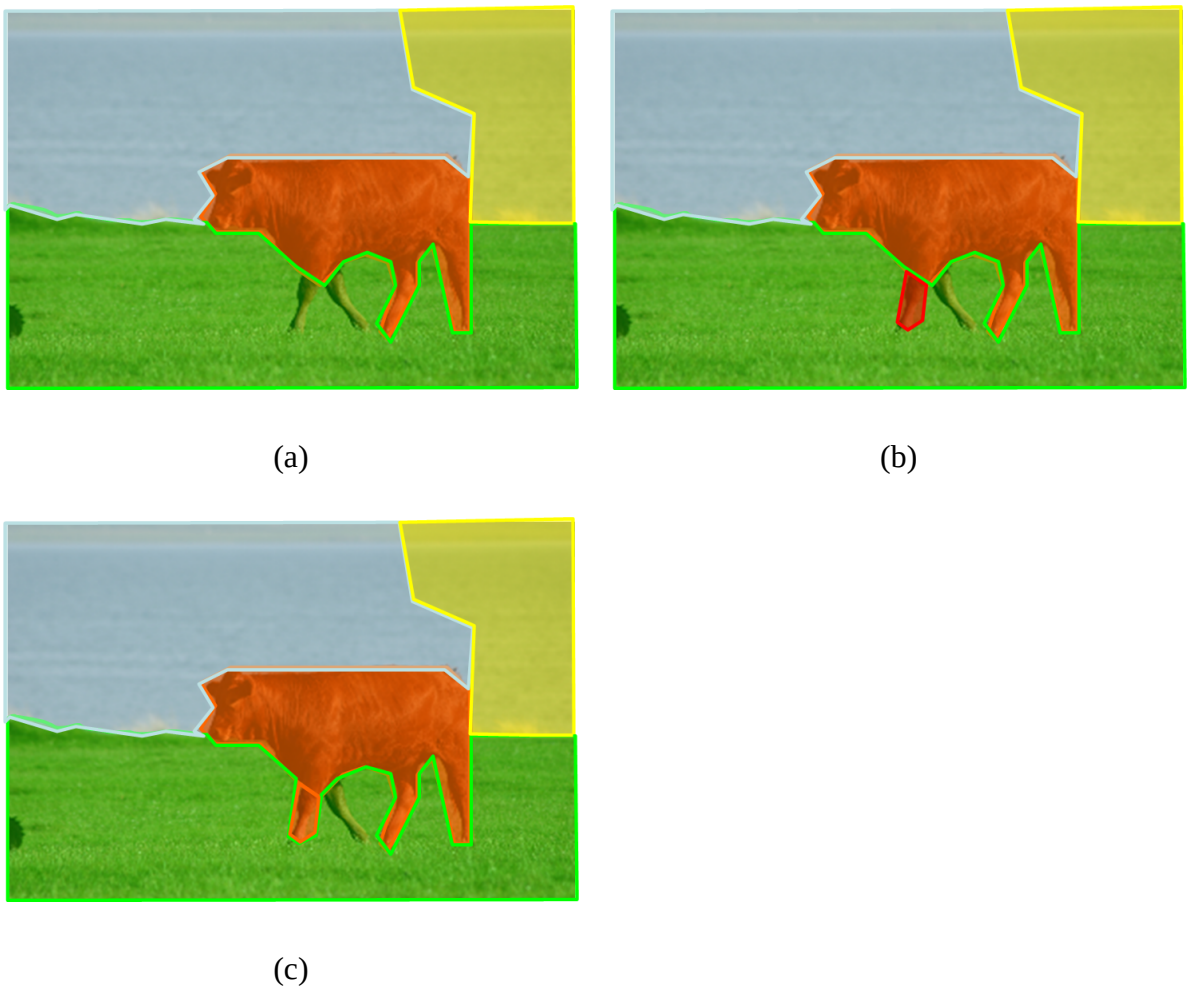
opposed to other Metropolis-Hastings algorithms). Unfortunately, the Gibbs sampler can be trapped in local minima, with the practical effect of not converging to the desired posterior distribution (convergence is only asymptotic). The reason why the Gibbs sampler is trapped in local minima is because the algorithm changes at most the state on one random variable each time a new sample is computed. Therefore, locally stable configurations are never updated (see [BZ05]). A solution for this problem would be to select a set of pixels (block) that are likely to be in the same segment and change their labels in a single step to a new value. As an illustration we consider the steps presented in Fig. 6.2: the current segmentation (a) has grouped the legs of the cows with the grass segment. The block sampler algorithm should select a set of pixels that are likely to be grouped together, such as the red region in (b). All the pixels in the red region have very high affinity between each other so their selection is desired. Finally a new label is sampled and the region updated (c).

To implement the concept of block sampling, we consider the Metropolis-Hasting algorithm [MRR$^+$53, Has70] presented in [BZ05], which is a generalization of the well known Swendsen-Wang sampling algorithm from statistical physics [SW87]. Given a proposal distribution $q(\boldsymbol{c'}; \boldsymbol{c})$ that from the current labeling $\boldsymbol{c}$ returns a new labeling $\boldsymbol{c'}$, the new labeling is kept with probability

$$a = \min\left(1, \frac{p(\boldsymbol{c'}|\boldsymbol{x}, \boldsymbol{w})}{p(\boldsymbol{c}|\boldsymbol{x}, \boldsymbol{w})} \frac{q(\boldsymbol{c}; \boldsymbol{c'})}{q(\boldsymbol{c'}; \boldsymbol{c})}\right). \tag{6.8}$$

To generate the new configuration we proceed as follows: given an image m in the collection create an undirected graph $G = (V, E)$ such that:

- For each observation x_{mn} (pixels in the image) we create a node $v_n \in V$.



(a) (b)

(c)

Figure 6.2: Block sampler. (a) A starting segmentation which assigns part of one object (the legs of the cow) to the wrong segment (grass). (b) The block sampler selects a set of pixels that are likely to have the same label (red region). (c) The sampler reassigns all the pixels in the proposed region to a new segment (the same the cow).

- For each pair of vertexes v_i and v_j we assign an edge $e_{ij} \in E$ if $A_{ij}^m > T$, i.e., the affinity between the two observations x_{mi} and x_{mj} is sufficiently strong to suggest they are in the same segment.

- For each edge e_{ij} we define an binary random variable b_{ij} which is set to 1 with probability $p_{ij} = f(A_{ij}^m)$.

Using the graph G, which is independent on the specific state of segmentation $\boldsymbol{c}$, we can

obtain a new graph $G' = (V, E')$ by removing all the edges between vertexes with different segment labels and by removing an edge e_{ij} with probability $1 - p_{ij}$. A block is selected by choosing at random a connected component S_h of the new graph G'. Finally, a new label is sampled for S_j based on the visual words in it. This will give a new segmentation $\boldsymbol{c'}$ that differs from $\boldsymbol{c}$ for the observation in S_h. Sampling c' from the proposal distribution $q(c'; c)$ can be done efficiently, since it requires computing the connected components of the a sparse graph $G' = (V, E')$. Directly computing the proposal distribution $q(\boldsymbol{c'}; \boldsymbol{c})$ is infeasible, because it requires summing the probability of all the possible ways of creating the connected component S_j. However, only the ratio between the two proposal distributions $q(\boldsymbol{c}; \boldsymbol{c'})$ and $q(\boldsymbol{c'}; \boldsymbol{c})$ is required to run the Metropolis-Hasting algorithm. This ratio can be computed easily because of cancellation of identical factors, and it involves only the edges between the vertexes in S_j and the vertexes with the old and new segment label. See [BZ05] for further details. Using the block sampler and the Gibbs sampler we can create a sampling procedure alternates between the two, with the block sampler responsible for "jumps" between locally optimal segmentation and the Gibbs sampler responsible for the diffusion of labels in "salt and pepper" segmentation that can be created by the block sampler.

6.2.2 Variational inference

To formulate the variational inference on the model of Figure 6.1 we write down the joint distribution of all the random variables

$$p(\boldsymbol{x}, \boldsymbol{w}, \boldsymbol{c}, \theta, \beta) = p(\boldsymbol{x}|\boldsymbol{c})p(\boldsymbol{w}|\boldsymbol{c}, \beta)p(\boldsymbol{c}|\theta)p(\theta)p(\beta), \tag{6.9}$$

where, given our assumptions on the distributions of the model, the expressions for each of the three factors are given by:

$$\begin{aligned}
p(\boldsymbol{x}|\boldsymbol{c}) &= \prod_{m=1}^{M}\prod_{n=1}^{N}\prod_{k=1}^{K}[f_k(x_{mn})]^{c_{mn}(k)} \\
p(\boldsymbol{w}|\boldsymbol{c}, \beta) &= \prod_{m=1}^{M}\prod_{n=1}^{N}\prod_{k=1}^{K}[\beta_{k,w_{mn}}]^{c_{mn}(k)} \\
p(\boldsymbol{c}|\theta) &= \prod_{m=1}^{M}\prod_{n=1}^{N}\prod_{k=1}^{K}(\theta_k^m)^{c_{mn}(k)} \\
p(\theta) &= \prod_{m=1}^{M} C(\alpha)\prod_{k=1}^{K}(\theta_k^m)^{(\alpha_k-1)} \\
p(\beta) &= \prod_{k=1}^{K} C(\eta)\prod_{h=1}^{V}\beta_{k,h}^{(\eta-1)}
\end{aligned} \tag{6.10}$$

with $C(\alpha)$ and $C(\eta)$ the normalization constant of the two Dirichlet distributions of parameter α and η (see [Bis06], p. 687). We then consider a variational distribution $q(\boldsymbol{c}, \theta, \beta)$ for

the hidden variables $\boldsymbol{c}$, θ, and β that factorizes, i.e., assumes independence, as:

$$q(\boldsymbol{c}, \theta, \beta) = q(\boldsymbol{c})q(\theta, \beta). \tag{6.11}$$

Following [Bis06], we derive the update equation for $q(\mathbf{c})$:

$$\begin{aligned}
\log q^*(\boldsymbol{c}) &= E_{q(\theta,\beta)}[\log p(\boldsymbol{x}, \boldsymbol{w}, \boldsymbol{c}, \theta, \beta)] + \text{const} \\
&= E_{q(\theta,\beta)}[\log p(\boldsymbol{x}|\boldsymbol{c})] + E_{q(\theta,\beta)}[\log p(\boldsymbol{w}|\boldsymbol{c}, \beta)] + E_{q(\theta,\beta)}[\log p(\boldsymbol{c}|\theta)] + \text{const} \\
&= \textstyle\sum_{m=1}^{M} \sum_{n=1}^{N} \sum_{k=1}^{K} c_{mn}(k) \\
&\quad \Big(\log f_{k,m}(x_{mn}) + E_{q(\beta)}[\log \beta_{k,w_{mn}}] + E_{q(\theta)}[\log \theta_k]) \Big) + \text{const},
\end{aligned} \tag{6.12}$$

with $E_{q(x)}[z]$ the expectation of random variable z under the probability distribution $q(x)$. In Eq 6.12 we have absorbed the terms $E_{q(\theta,\beta)}[\log p(\theta)]$ and $E_{q(\theta,\beta)}[\log p(\beta)]$ into the constant, since they are independent of $\boldsymbol{c}$. Taking the exponent of both sides of Eq. 6.12 and normalizing provides:

$$q(\boldsymbol{c}) = \prod_{m=1}^{M} \prod_{n=1}^{N} \prod_{k=1}^{K} r_{mn}(k)^{c_{mn}(k)}, \tag{6.13}$$

where we defined the responsibilities:

$$r_{mn}(k) = \frac{f_{m,k}(x_{mn}) \exp(E_{q(\theta)}[\log \theta_k] + E_{q(\beta)}[\log \beta_{k,w_{mn}}])}{\sum_k f_{m,k}(x_{mn}) \exp(E_{q(\theta)}[\log \theta_k] + E_{q(\beta)}[\log \beta_{k,w_{mn}}])}. \tag{6.14}$$

Eq. 6.13 shows that the variational density factorizes into N independent multinomial distributions, one for each term $\boldsymbol{c_n}$. The parameters of each multinomial $q(\boldsymbol{c_n})$ are the responsibilities $(r_{mn}(1), r_{mn}(2), \ldots, r_{mn}(K))$ in Eq. 6.14. Similarly, for the variational distribution $q(\theta, \beta)$, we have the update equation:

$$\begin{aligned} \log q^*(\theta,\beta) &= E_{q(\boldsymbol{c})}[\log p(\boldsymbol{x},\boldsymbol{w},\boldsymbol{c},\theta,\beta)] + \text{const} \\ &= \sum_{m=1}^{M}\sum_{n=1}^{N}\sum_{k=1}^{K} E_{q(\boldsymbol{c})}[c_{mn}(k)]\log\theta_k^m + \log p(\theta) + \\ &\quad \sum_{m=1}^{M}\sum_{n=1}^{N}\sum_{k=1}^{K} E_{q(\boldsymbol{c})}[c_{mn}(k)]\log\beta_{k,w_{mn}} + \log p(\beta) \\ &\quad + \text{const}. \end{aligned} \tag{6.15}$$

The first term of Eq. 6.15 is:

$$\sum_{m=1}^{M}\sum_{k=1}^{K}\Big(\alpha_k + \sum_{n=1}^{N} r_{mn}(k) - 1\Big)\log\theta_k^m, \tag{6.16}$$

while the second term of Eq. 3.17 is:

$$\sum_{k=1}^{K}\sum_{v=1}^{V}\Big(\eta + n_{k,h} - 1\Big)\log\beta_{k,h}, \tag{6.17}$$

with $n_{k,h}$ the sum of all the responsibilities $r_{mn}(k)$ for which $w_{mn} = h$. Taking the exponential of Eq. 3.17 we show that the variational distribution factorizes as:

$$q(\theta,\beta) = q(\theta)q(\beta), \tag{6.18}$$

where the first variational distribution has the functional form of a product of Dirichlet distributions:

$$\prod_{m=1}^{M} C(\gamma) \prod_{k=1}^{K} \theta^{(\gamma_k^m - 1)}, \tag{6.19}$$

with parameters:

$$\gamma_k^m = \alpha_k + \sum_n r_{mn}(k) = \alpha_k + R_{mk}, \tag{6.20}$$

where R_{mk} represents the total responsibility for segment k in document m.

The second factor of the variational distribution $q(\theta, \beta)$ has the functional form:

$$\prod_{k=1}^{K} C(\phi) \prod_{h=1}^{V} \beta^{(\phi_{k,h} - 1)}, \tag{6.21}$$

with parameters:

$$\phi_{k,m} = \eta + n_{k,h} \tag{6.22}$$

with $n_{k,h}$ defined as before.

Note that we did not assume any particular functional form for $q(\boldsymbol{c})$ and $q(\theta, \beta)$. Instead, Eq. 6.13 and Eq. 6.18 follow from the graphical structure and the distributions used in the model, as well as from the factorized form $q(\theta, \beta, \boldsymbol{c}) = q(\beta, \theta) q(\boldsymbol{c})$. Finally, since $q(\theta)$ is a Dirichlet distribution, we can derive a closed-form solution for $E_{q(\theta)}[\log p(\theta_k)] = \Psi(\gamma_k) - \Psi(\sum_k \gamma_k)$, where $\Psi(a)$ is the first derivative of $\log \Gamma(a)$ (see [BNJ03] for the details of the derivation). This expression is then used to compute the responsibilities in Eq. 3.16.

Using Eq. 3.21 together with Eq. 3.15 and Eq. 3.18, we obtain a system of coupled

equations that can be iteratively solved in similar way as described in the algorithm of Fig. 3.4.3. Also in this case we can use the KDE approximation to compute the quantities $f_{k,m}(x_{mn})$ as proposed in Chapter 3.

The variational approximation scheme is particularly suitable for implementation on a parallel system. The computation of the responsibilities $r_{mn}(k)$ and of the parameters γ^m can be done independently for each image m. Once these quantities are available the parameters ϕ can be computed as well by collecting from each image the sum of the responsabilities for each visual word in the dictionary.

Chapter 7

Experimental Results

Following Fei-Fei et al. [FFP05] we extract patches by densely sampling each image with a grid of 4 pixels. For each patch a local descriptor is computed. We experimented with three possible descriptors: the RGB value of the central pixel of the patch, filter bank outputs [WCM05] (see Fig. 7.1), and the well known SIFT descriptor [Low04]. The dimensionality of the descriptor vectors are 3, 17, and 128, respectively. In all three cases a subset of the extracted descriptors is used to construct a visual dictionary via K-means clustering (see Sivic et al. [SRE$^+$05]). We experimented with three different dictionary sizes: 256, 512, and 1024. Finally, the visual word assigned to the patch is the label of the most similar dictionary element. The multinomial distribution of visual words ϕ_k are shared across images since they model the appearance of recurring elements in the collection.

In all our experiments the densities $f_{k,m}$ are non-parametric (see Section 3.2.1) and are assumed independent between images (see Section 6). We use the intervening contour method [CBS05] to compute the affinities used for the non-parametric approximation of $f_{k,m}$. We also experimented with the semi-parametric model (see Section 3.2.3) which achieved comparable performance but required more computational resources for estimating the mean and the covariance of the parametric term.

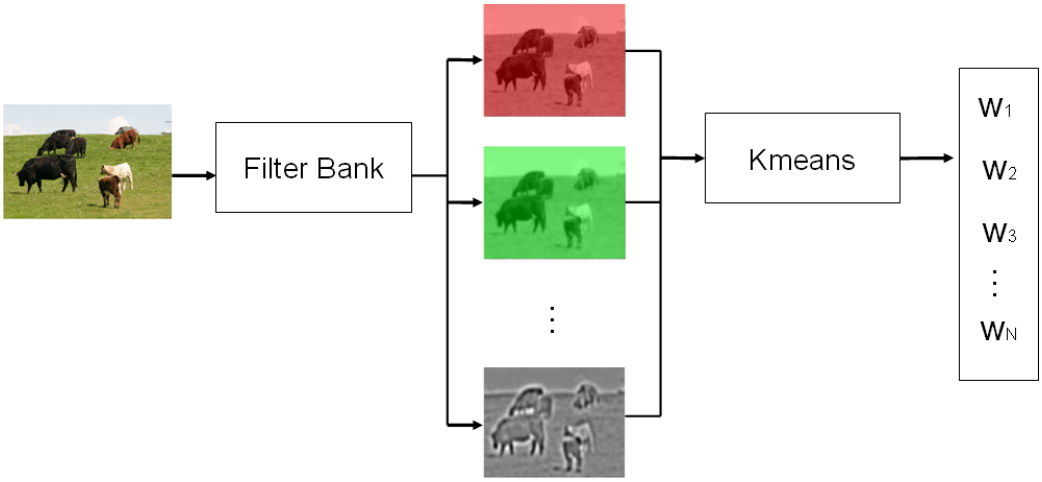


Figure 7.1: Filter banks visual words. The schema shows how visual words are computed using a filter bank. The different color channels in the images are filtered with different Gaussian (low-pass filters for capturing color information) and gradient (high pass filter for capturing edges and texture information) filters. After the filtering each pixel is represented by an 18 dimensional vector. The visual words are obtained by running kmeans over all the pixels, and assigning the discrete label of the cluster to which a pixel is assigned.

The computational cost of the inference algorithm for the model of Fig. 6.1 is linear in the number of images and in the number of topics/segments K (see Appendix 6.2.1.1 for the implementation details). The algorithm is implemented in C++ and it has a running time of about 20 sec. per image (with $K = 20$) on a 2.50GHz Intel Xeon machine. This running time is much smaller that the one reported in Section 5.2 for two reasons. First we are sampling the image on a 4×4 regular grid, hence reducing the number of observations in the collection. Second we are using the non-parametric representation of Section 3.2.1 for the segment densities $f_{k,m}$ rather than the semi-parametric representation used in Section 5.

We tested our system on four databases: the Microsoft Research Cambridge dataset version one (MSRCv1) and version 2 (MSRCv2) [Cri04], a subset of the LabelMe dataset [RES+06], and the scene database of Oliva and Torralba [OT01]. Note that our experiments are completely unsupervised: we do not use any labeling information during inference.

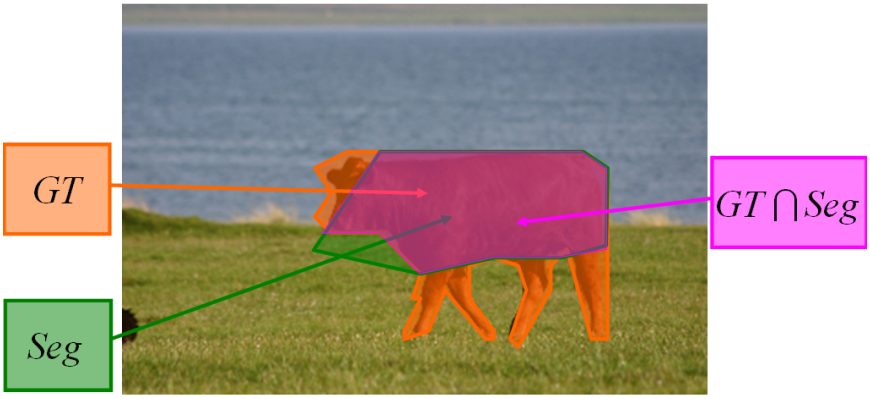


Figure 7.2: Error measures. For each image used in the experiments the ground truth segmentation (GT) is available (orange region). The result segment for the cow category obtained from the A-LDA is displayed in dark green. The intersection of the two regions is the set of correctly identified pixels in the image (magenta).

The "ground truth" segmentation is used only to evaluate the segmentation results. The results of our unsupervised recognition/segmentation system are illustrated by showing the segmentation masks and by reporting numerical evaluation of the segmentation accuracy of the model. Finally, we provide a comparison with three other related probabilistic models: the Gaussian mixture model (GMM), the latent Dirichlet allocation (LDA), and the spatial latent Dirichlet allocation (S-LDA) [WG07].

7.1 Evaluation metrics

To obtain a numerical evaluation of the performance of the A-LDA model we introduce the two error measures of precision and recall. Considering Fig. 7.2, the ground truth GT for the segment containing the cow is represented by the orange region, the segmentation result for the category cow obtained from the A-LDA is represented by the dark green region, and the intersection of the two regions is represented by the magenta region. The precision and

recall values are then defined as:

$$prec = \frac{|GT \cap SEG|}{|SEG|} \qquad rec = \frac{|GT \cap SEG|}{|GT|}. \tag{7.1}$$

Following [EVGW$^+$], we also define the segmentation accuracy for a category as the number of correctly labeled pixels in that category, divided by the number of pixels labeled in that category in either the ground truth or the segmentation results (intersection/union metric).

$$acc = \frac{|GT \cap SEG|}{|GT \cup SEG|}. \tag{7.2}$$

This pixel-based measure has several limitation: it does not take into account multiple instances of the same object category in a single image and it does not consider the quality of the segment contours. Nonetheless we decided to use this particular definition because it is a de facto standard for the computer vision community and it has been used to evaluate other (supervised) segmentation/recognition systems [VT07][SWRC09].

A final caveat for the evaluation of the segmentation performance is the labeling of the topics obtained from the A-LDA model. Since our model is fully unsupervised there is no possibility of understanding which topic corresponds to which category. To be able to use the precision/recall and the accuracy metrics we need to associate topics to categories. We obtain this association by dividing the dataset into two parts of equal size: a probe set and a test set. We use the probe set to compute the matching between the topics and the categories. The precision/recall values and the segmentation accuracy is evaluated on the test set. It is important to emphasize that the segmentation of the dataset is obtained using

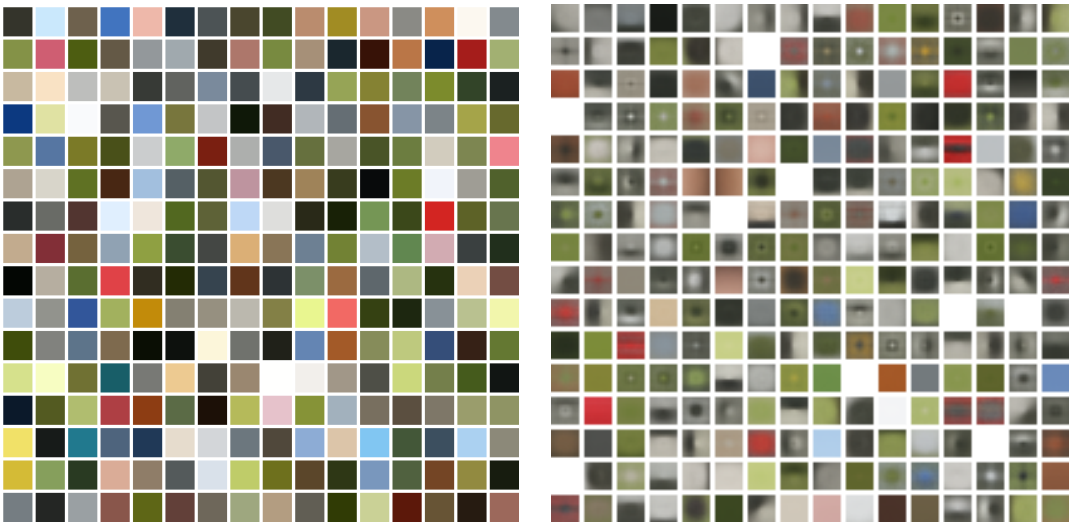


Figure 7.3: Visual words dictionaries. Left: 256 visual words when the pixel color is used as descriptor. Right: average of the patches associated to 256 visual words when the filter bank is used as descriptor.

no human labeling. In principle it would be possible to have a human operator to inspect, at a single glance, all the segments in the same topic and give a category label for that topic, propagating the label to all the pixels in the image collection.

7.2 Comparing different types of visual words

The first descriptor we tested is the RGB value at the center of a patch. The left panel of Fig. 7.3 shows the RGB colors associated with the centroid of the dictionary words (256 visual words). We used the MSRCv1 dataset to obtain these centroids. We observe that a lot of the visual words in the dictionary correspond to green texture. This is a consequence of the large quantity of grass and foliage present in the MSRC dataset[1].

Fig. 7.4 shows unsupervised segmentation results of several images of the MSRCv1 dataset. Each categorical segment is marked with the same color in all the images (arbi-

[1]These two classes account for almost 30% of the pixels in the dataset.

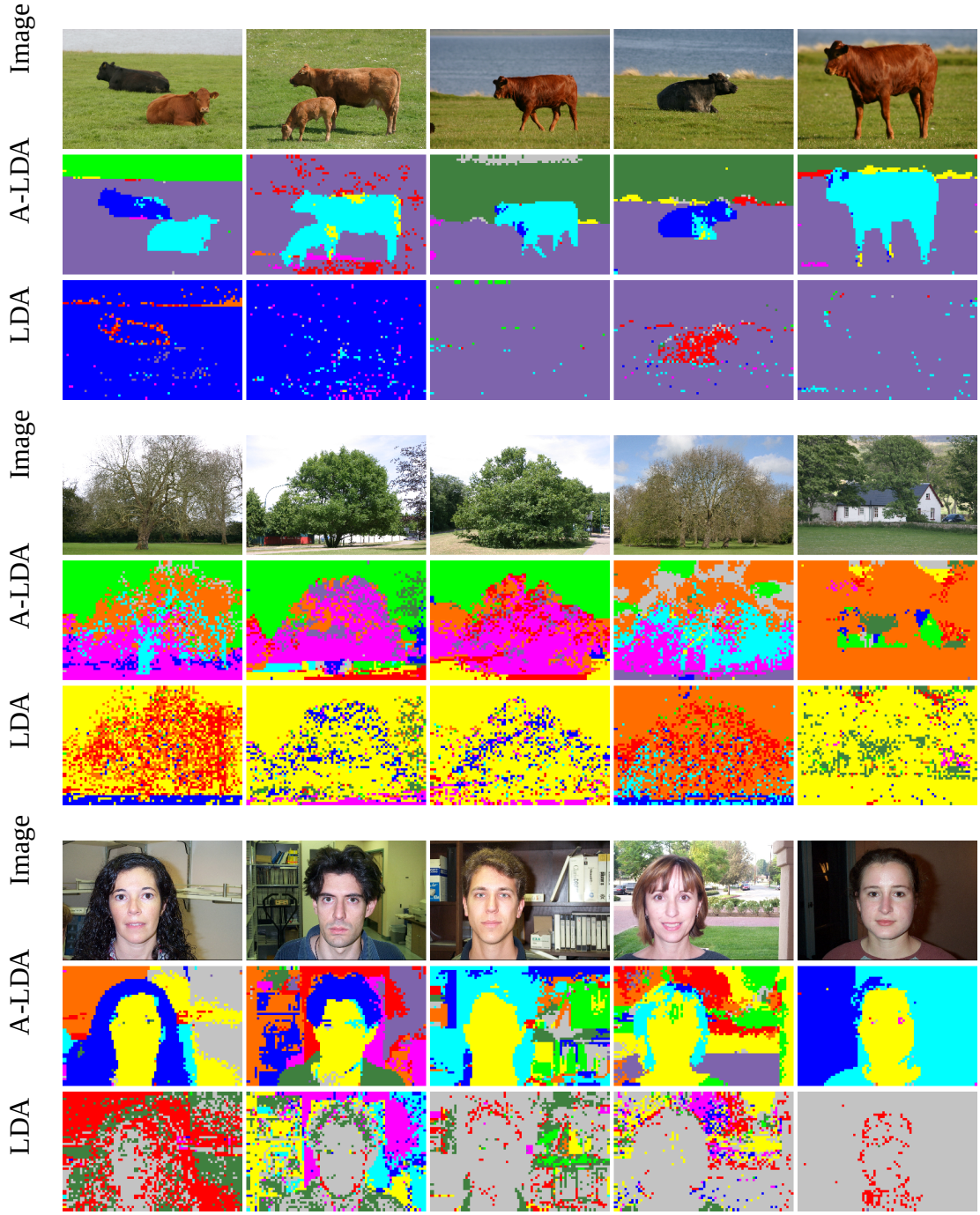


Figure 7.4: Unsupervised segmentation and recognition results when only RGB information is used to construct the visual words. Three panels are presented. In each of the three panels we present the original image, the segmentation using the A-LDA model, and the segmentation using the LDA model. The three panels show the different types of images: cows, trees, and faces. For a specific model the same color in different images identifies the same topic segment.

trarily chosen to highlight individual segments). Notice that corresponding regions tend to have the same color across all images indicating that the unsupervised algorithm has "discovered" the corresponding categories: e.g., the sky segment is always assigned to the green label. To obtain a quantitative evaluation of our system we consider the segmentation error with respect to the ground truth. We consider only a subset of the 13 categories present in the dataset since some categories are very rare, i.e., they occupy less then 1% of the total number of pixels in the collection. In particular we do not consider: sheep (0.45%), horse (0.18%), and mountains (0.25%). Fig. 7.5 shows the precision/recall plots for each category when using a dictionary of 1024 visual words and 20 segments ($K = 20$). We also experimented with other sizes of the dictionary (256 and 512). The overall performance of the system did not change significantly with the dictionary size, with a minor advantage being gained by using a larger dictionary.

The second descriptor we tested is the output of a filter bank [WCM05] at each pixel location. Fig. 7.3 shows the means of all 11×11 patches assigned to each dictionary word when the filter-bank responses are used as basic patch descriptors. We observe that with this descriptor we have two types of visual words: color visual words and texture visual words. The first type describes uniform patches based on their color, while the second type characterizes image patches by the specific gradient pattern they describe (a centered dot or a slanted edge) and usually have a gray color (from averaging patches of different color but similar gradient pattern).

Fig. 7.6 shows unsupervised segmentation results of several images of the MSRCv1 dataset. Each categorical segment is marked by the same color across all images. Fig. 7.7

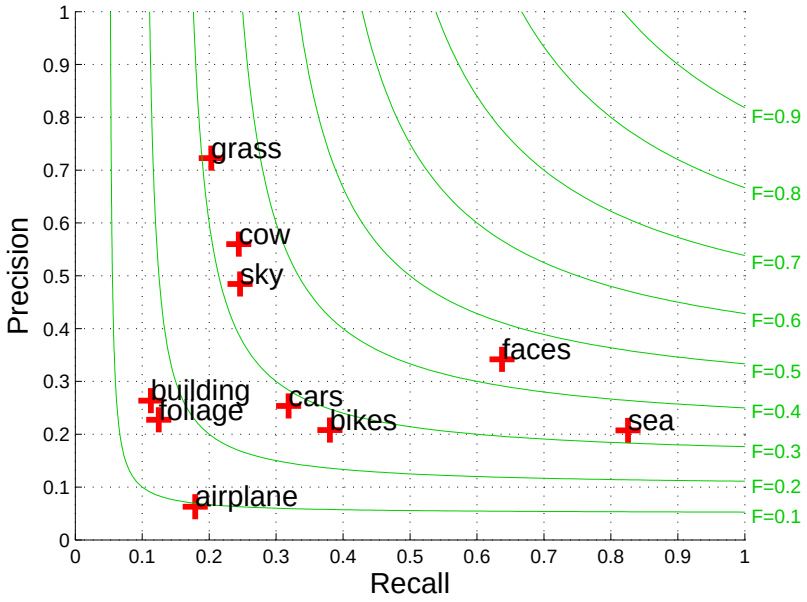


Figure 7.5: Precision/recall plots for the MSRC dataset when using visual word based on RGB color. The dictionary size is of 1024 visual words. The number of topics K is set to 20. The F-measure isolines are defined as in Fig. 5.3.

shows the precision/recall plots for all the considered categories. We can see that our model performs extremely well on the grass category which is the single most popular category in the dataset (20% of the pixels are labeled grass). Other categories like faces, sky, and foliage(tree), have medium performance. The most challenging categories are airplanes, cars, and sea. In particular the airplanes category is almost never recovered. The problem with the airplanes and cars categories is that they have a wide range of appearances and points of view which makes it difficult for the A-LDA model to spot their recurrence across images without supervision. The sea category is relatively rare compared to the others, less than 1% of the dataset.

For each category we compute the segmentation accuracy as a measure of the system performance. Fig. 7.8a shows the scatter plot of the accuracies for each category when

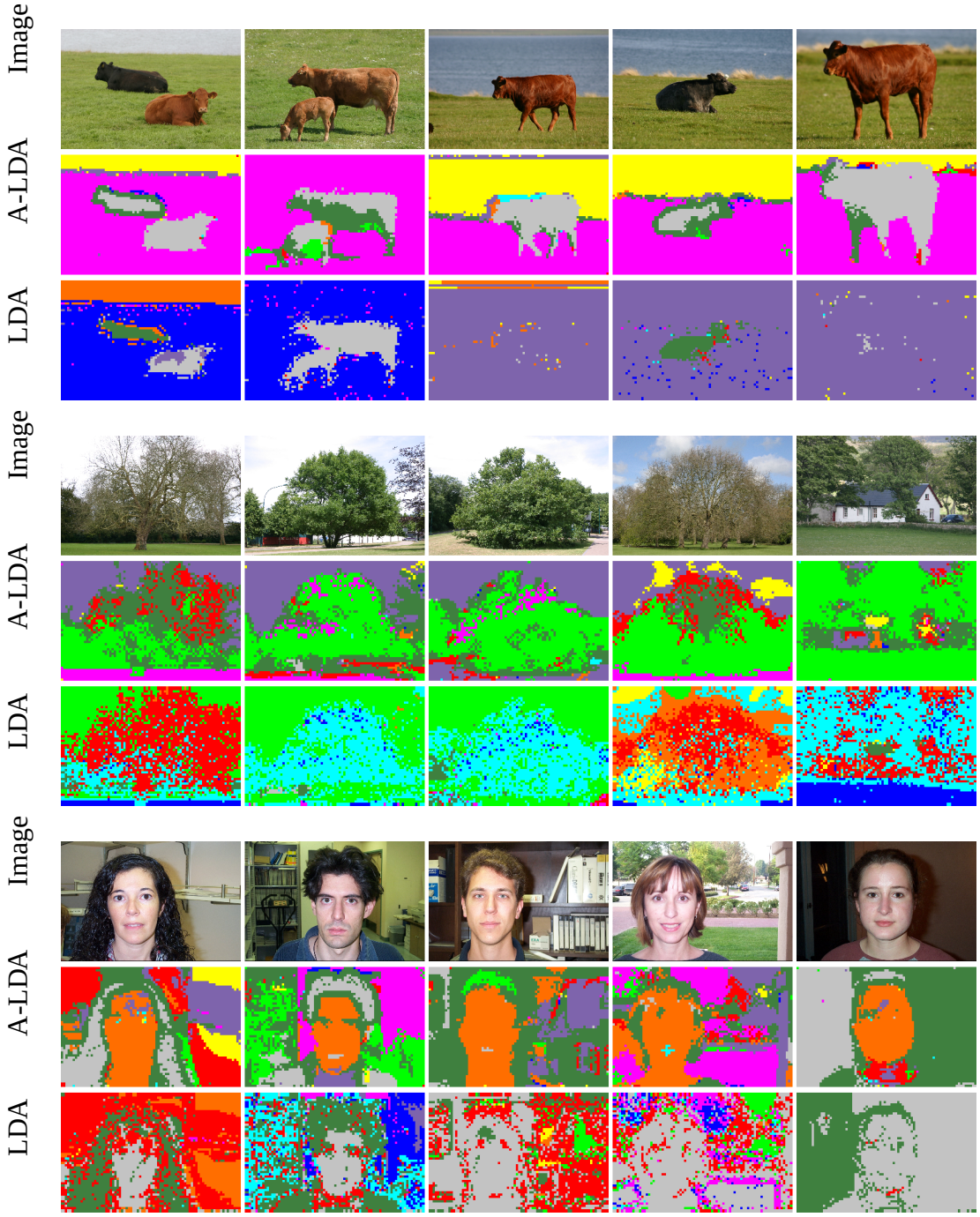


Figure 7.6: Unsupervised segmentation and recognition when filter responses are used to construct the visual words. Similarly to Fig. 7.4, we present three panels.. In each of the three panels we present the original image, the segmentation using the A-LDA models and the segmentation using the LDA model. The three panels show different types of images: cows, trees, and faces. For a specific model the same color in different images identifies the same topic segment.

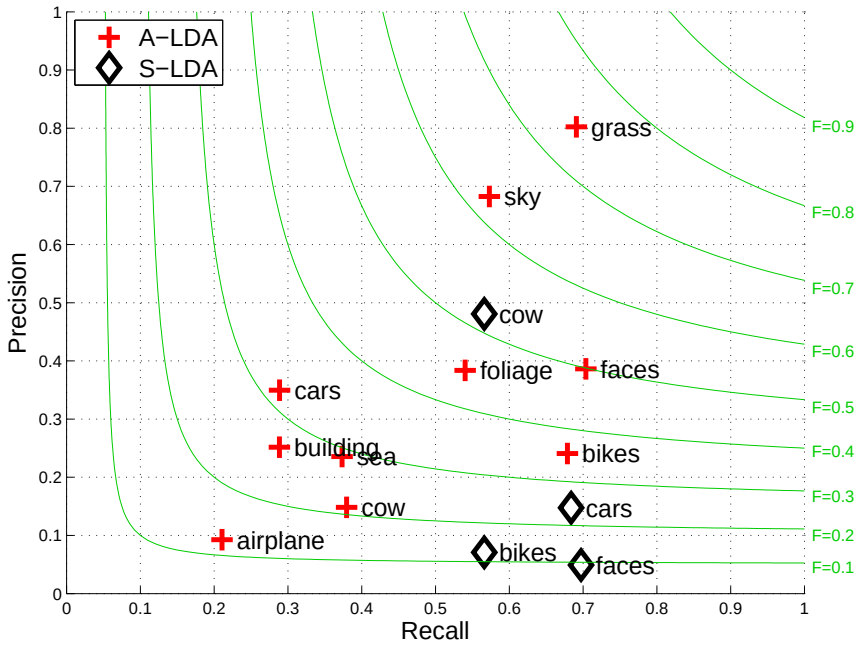
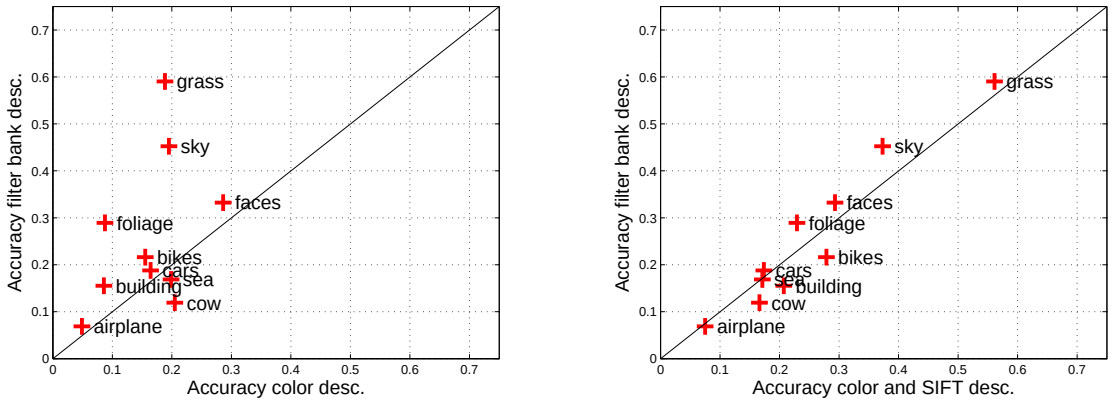


Figure 7.7: Precision/recall results for the MSRC dataset when using visual words based on filter bank responses (red crosses). The dictionary size is 1024 visual words. The number of topics K is set to 20. The precision/recall results for the spatial latent Dirichlet allocation (S-LDA) [WG07] are also reported (black diamonds).

using color visual words and when using vector-quantized filter-bank responses. We see that in general the filter banks perform better, although for the cows and sea categories the color visual words perform better. We also tested a third type of visual words based on the SIFT descriptor. Since the SIFT descriptor is based on the intensity gradient, it does not capture color information. In order to also consider color information, we modified the model of Fig. 6.1 to have two different visual words per observation: one derived from color (see previous discussion) and one derived from SIFT[2]. For a given segment k, visual words of different types are sampled from two independent multinomial distributions ϕ_k^c (color) and ϕ_k^s (SIFT). Fig. 7.8b shows the scatter plot of the accuracies when using color/SIFT visual words and when using filter-bank visual words. We observe that the filter-bank visual

[2]Using only visual words based on SIFT merges categories with similar texture, but different color like grass and sea.



(a) (b)

Figure 7.8: Comparison of the segmentation accuracy of the A-LDA model for different types of visual words: color (RGB) visual words (horizontal axis) and the filter bank visual words (vertical axis).

words and the joint color/SIFT ones have similar accuracy results (close to the diagonal) with the filter-bank visual words performing better for the categories grass, sky, faces, and foliage, and the color/SIFT visual words giving greater accuracy for building, bikes, and cows. As previously observed, filter bank visual words can be divided in two groups: color and texture. Since color/SIFT visual words also capture these two patch properties (in a different way), the similarity of segmentation accuracy is not surprising. In all the following experiments we will always use filter-bank visual words. We also experimented with other collections of images such as the Boston urban area subset of LabelMe [RES+06] and the scene dataset used by Oliva and Torralba [OT01]. Fig. 7.9 and Fig. 7.10 show several examples of categorical segments learned from these datasets.

All the experiments considered so far are completely unsupervised, i.e., neither regions of an image nor whole images have any label. If we allow for a certain amount of supervision we can improve the performance over the unsupervised case. For example, we can

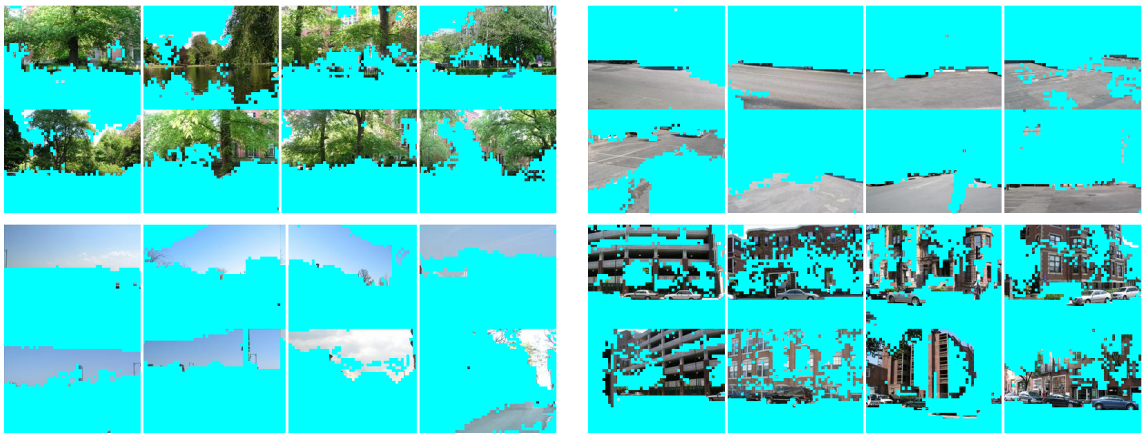


Figure 7.9: Four topics/segments learned from the LabelMe database. Each panel contains 8 segments from the same topic. The four topics represent four different elements of a possible street scene: "tree/foliage", "buildings", "street pavement", and "sky". These topic panels show the consistency we obtain across the images of the collection.

consider the case when we know a priori which objects are present in each image of the collection. In this case we share statistics only between images that contain the same object, as in the experiment of Section 5.2. Fig. 7.14 compares the precision/recall values for the unsupervised case (red crosses) and the semi-supervised case (blue circles). In the first case all the images in the collection are segmented together and the model has to determine which object is present in each image. In the second case, we segment together only images that contain objects from the same category[3]. Using this limited information we can achieve much higher precision/recall values on all the categories we have labeled.

7.3 Comparing different inference algorithms

In Chapter 6, we developed two inference algorithms for the model of Fig. 6.1: one based on sampling, specifically Gibbs sampling and a variation of the Swendsen-Wang sampling

[3]We only consider one category for each image. For example if an image has both cows and grass we only consider cows.

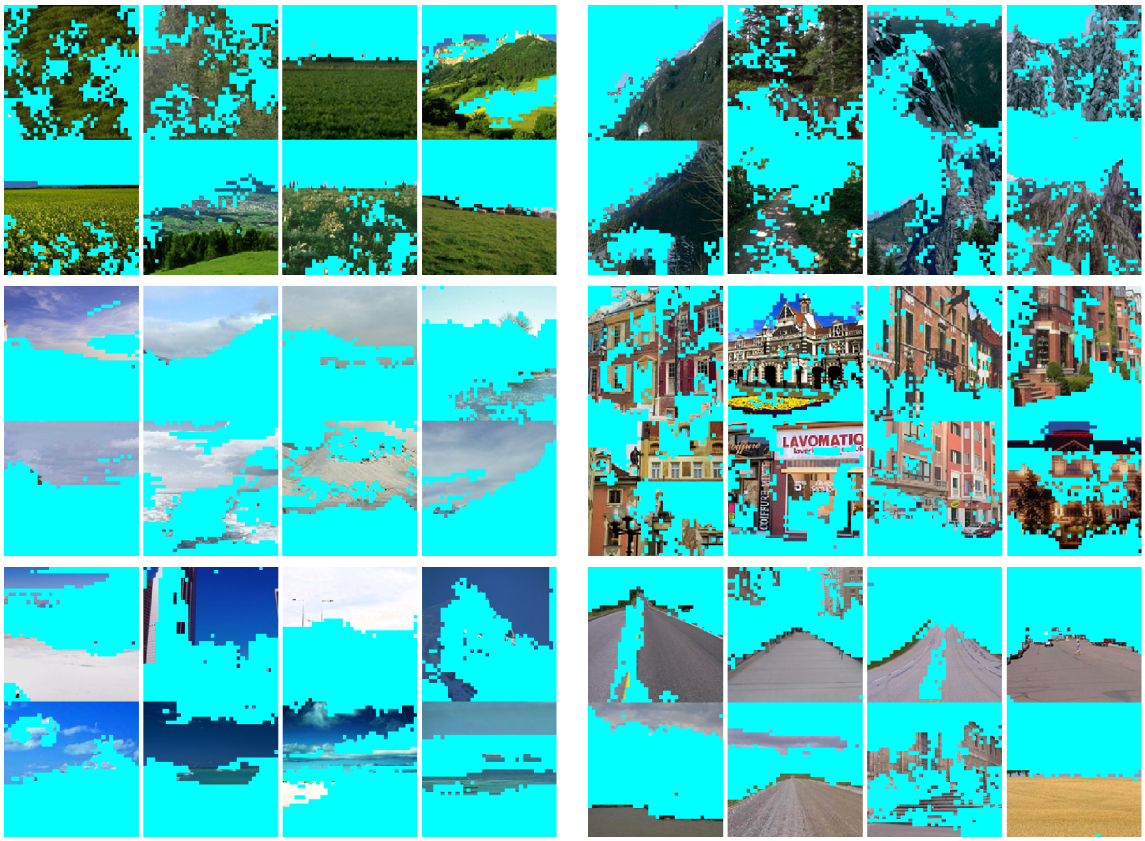


Figure 7.10: Six topics/segments learned from the Scene database. Each panel contains 8 segments from the same topic. Our visual words representation incorporates color information, therefore skies were assigned to two topics, light blue and dark blue.

(block sampler), and one based on variational approximation. In this section we review their performance in term of accuracy and computational cost. We also discuss other aspects such as the suitability for a parallel implementation.

The first inference method is based on two sampling algorithms: the Gibbs sampling and the Swendsen. We first compare the advantage of using both algorithms in alternate steps to using only the Gibbs sampler. Fig. 7.15 shows the precision/recall plot and the scatter plot for the accuracies of the categories in the MSRCv1 for the two sampling algorithms. We can see that performance is fairly similar, with an improvement for the cow and grass categories. Although the block sampler does not improve the accuracy perfor-

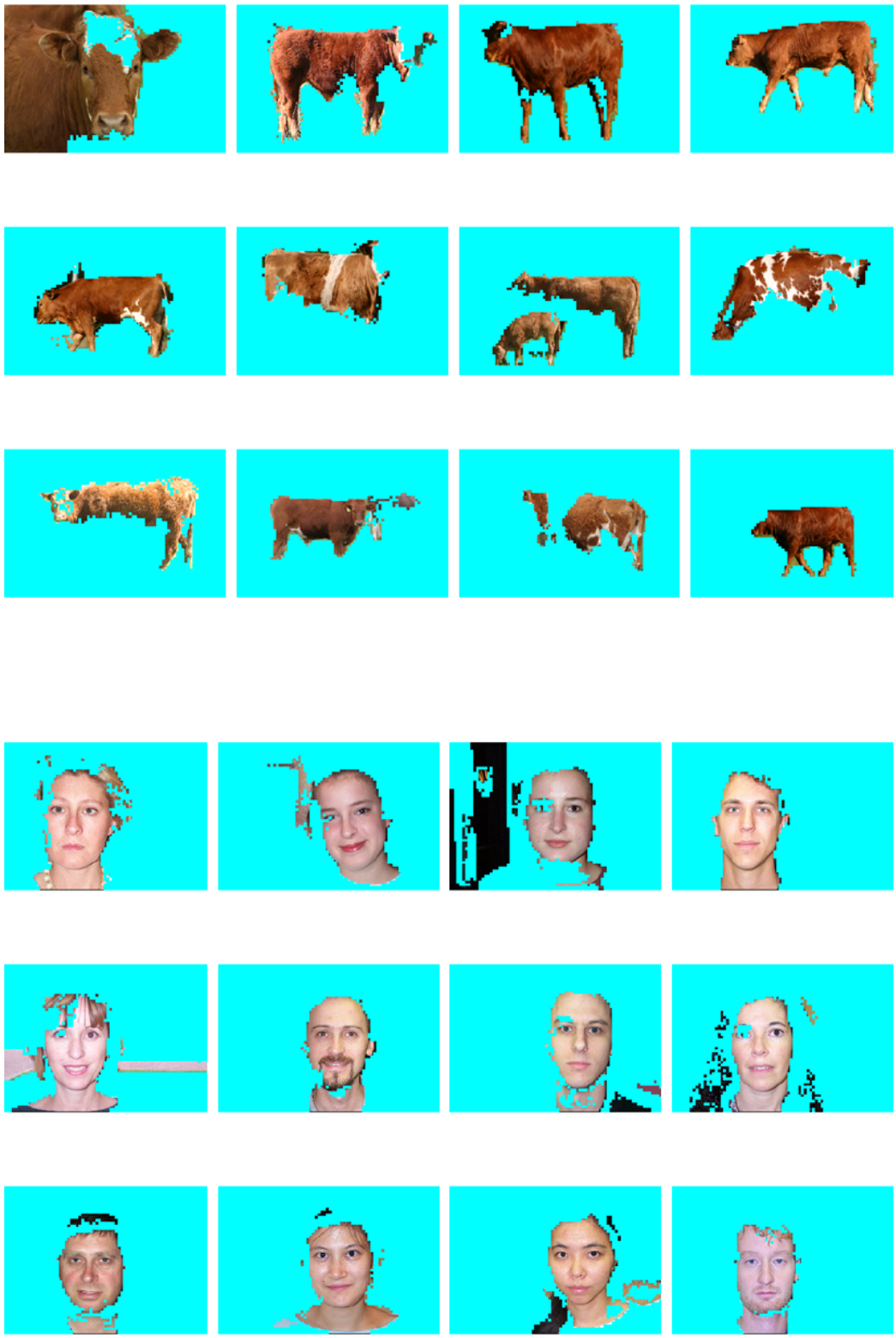


Figure 7.11: Categorical segments from MSRCv1. The top panel shows 12 segments from the category “cows”. The bottom panel shows 12 segments from the category “faces”. These two categories are often confused by the A-LDA model because of the color similarity.

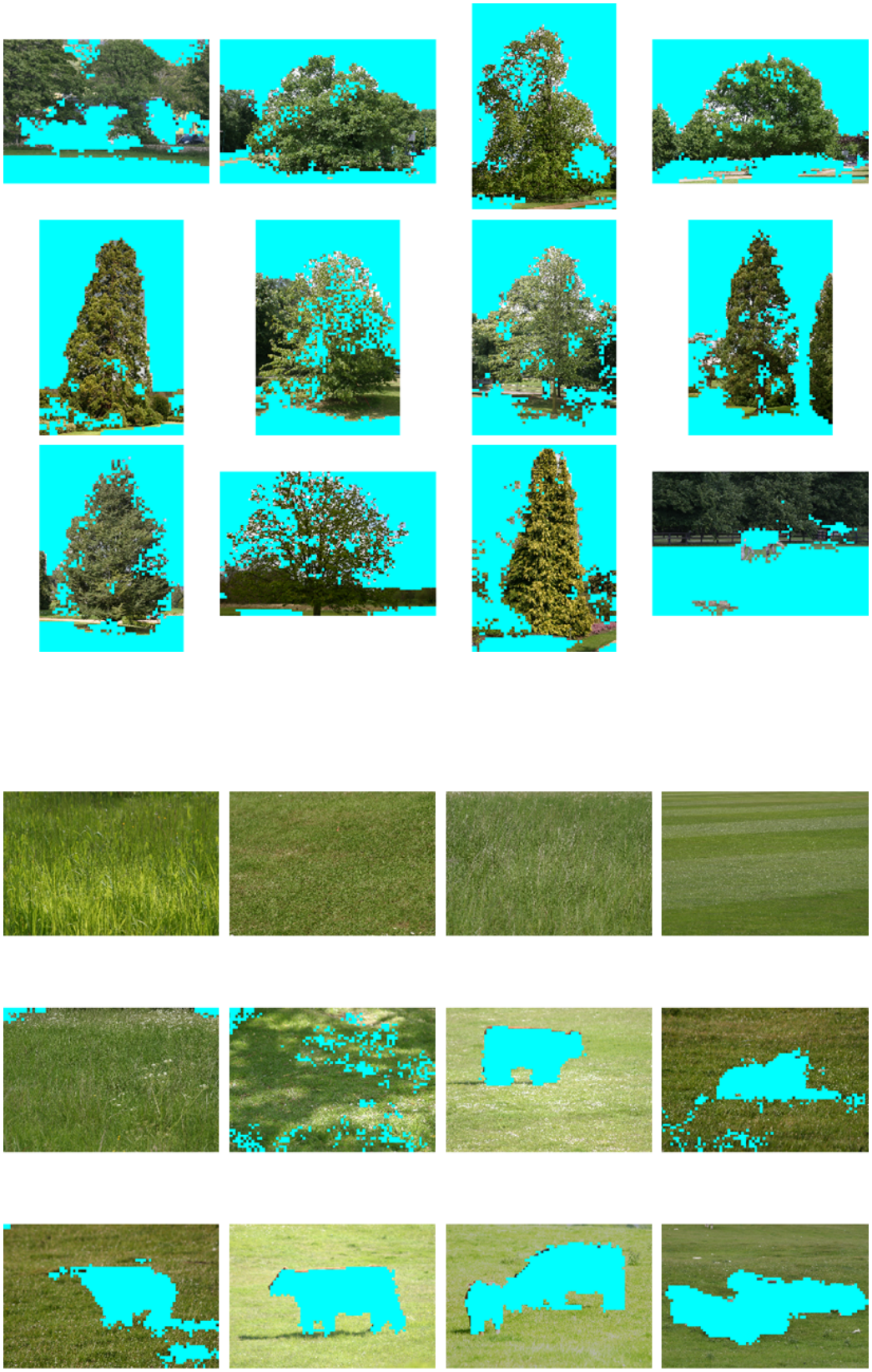


Figure 7.12: Categorical segments from MSRCv1. The top panel shows 12 segments from the category “tree”. The bottom panel shows 12 segments from the category “grass”.

Figure 7.13: Categorical segments from MSRCv1. The top panel shows 12 segments from the category “bicycles”. The bottom panel shows 12 segments from the category “sky”.

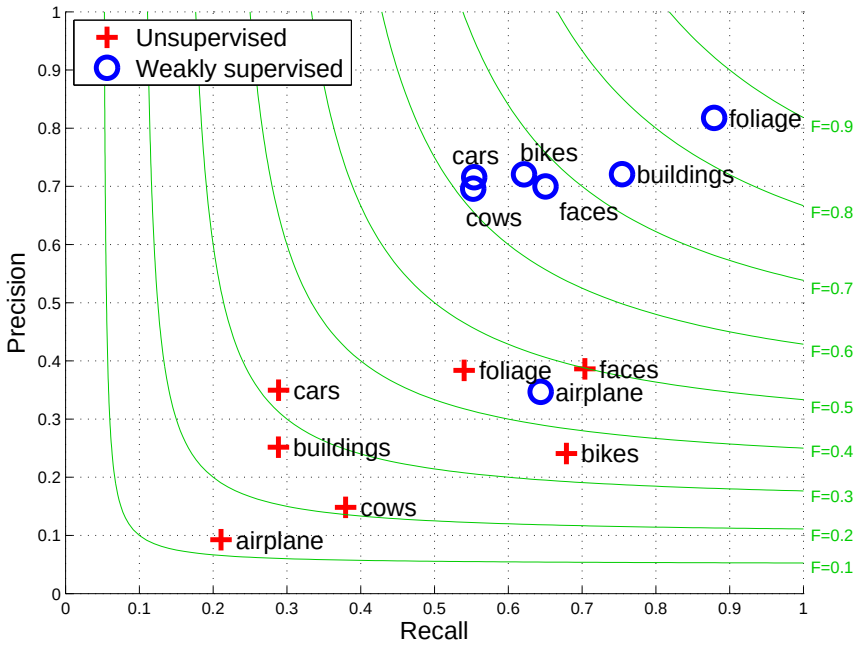
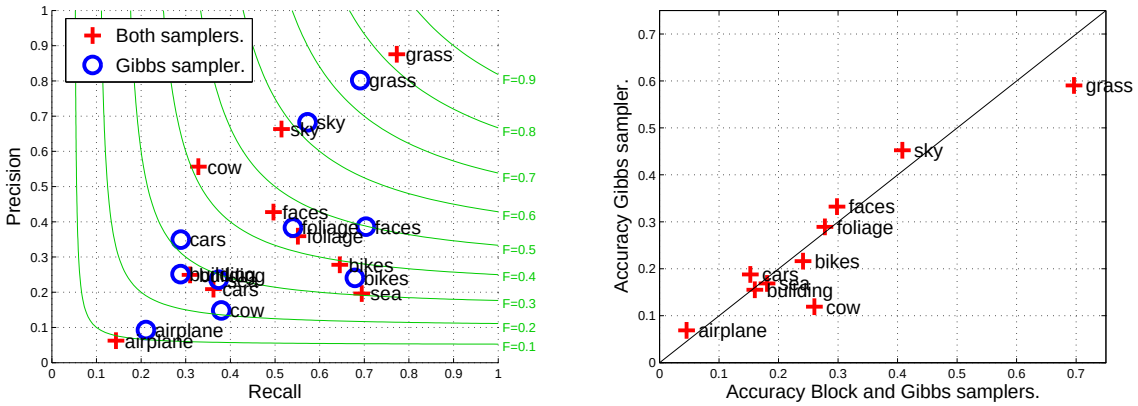


Figure 7.14: Precision/recall plots showing the segmentation/recognition performance of the A-LDA on seven categories: airplanes, bikes, buildings, cars, cows, faces, and trees (foliage). The red crosses refer to the unsupervised case (see Fig. 7.7). The blue circles refer to the weakly-supervised case, where the category label of the objects in an image is known. Even this limited amount of supervision, a single label for the whole image, greatly improves performance of the segmentation.

mance of the model, it is capable of reducing the computational cost of the inference step by reducing the sampling time from 18.75 seconds per image to 3.89 seconds per image.

The second inference algorithm we developed is based on variational approximation of the posterior distribution $p(\mathbf{c}|\mathbf{x}, \mathbf{w})$ (see Section 6.2.2). Fig. 7.16 shows the precision/recall plot and the scatter plot comparing the variational inference and the Gibbs sampling. We can see that the variational algorithm is under-performing for most categories, particularly for those where the Gibbs sampling gives good results. Although less precise in terms of segmentation/recognition accuracy the variational algorithm is one order of magnitude faster than the Gibbs sampler: 1.35 seconds per image as opposed to 18.75 seconds per image. The variational algorithm can also easily be parallelized as observed in Section 6.2.2.



(a) (b)

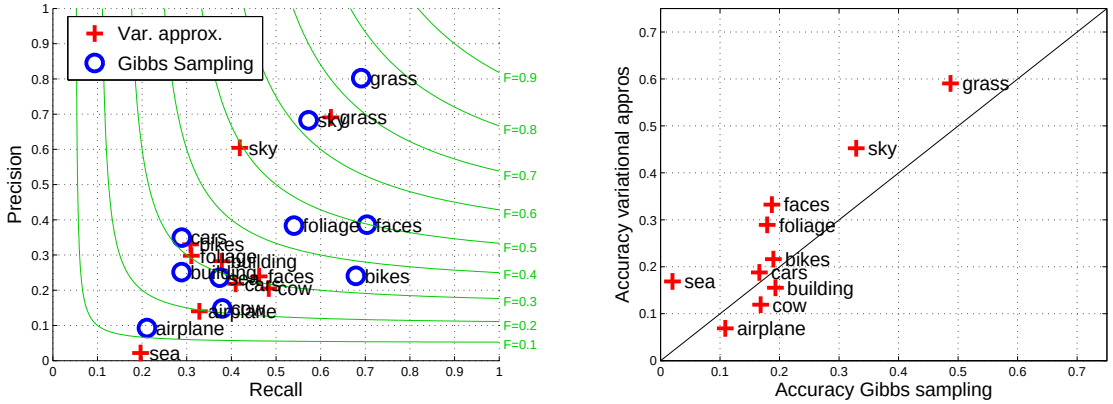
Figure 7.15: Comparison of sampling algorithms. (a) Precision/recall results showing the segmentation/recognition performance of the Gibbs sampler inference algorithm (blue circle), and the Block sampler together with the Gibbs sampler (red crosses). (b) Scatter plot of the accuracy for the two sampling algorithms.

These two properties make it suitable for large scale problems.

7.4 Comparison with other probabilistic models

We compare the A-LDA model with three alternative models. The first model is a simple Gaussian mixture model (GMM) with the same number of components as topics/segments in the A-LDA model. To obtain the model we collect all the descriptors of all the images and estimate the model parameters and the observation assignment using EM. We observe that when estimating the model we use neither any affinity information (segmentation cues) nor image membership.

Another possible probabilistic model is the LDA model. As observed in Section 6, this model can be seen as a simplification of the A-LDA model in which the x_{mn} variable is removed. Therefore, the LDA model does not consider the relationship between the



(a) (b)

Figure 7.16: Comparison of sampling and variational algorithms. (a) Precision/recall plots showing the segmentation/recognition performance of the Gibbs sampler inference algorithm (blue circle) and the variational inference (red crosses). (b) Scatter plot of the accuracy for the Gibbs sampler and variational approximation.

visual words of an image (affinities information), but does consider image membership, i.e., the same visual word may have different meanings in different images. The number of segments is 20 in all the experiments and a dictionary of 1024 visual words is used for both the LDA model and A-LDA model. We use filter bank responses as the descriptor for image patches. Fig. 7.17 shows scatter plots comparing the A-LDA model with GMM (left) and LDA (right). We see that the A-LDA outperforms GMM on all the categories in the dataset. The A-LDA outperforms the LDA in all the categories but two: cars and cows. It is also interesting to study the results from the LDA shown in Fig. 7.18: we can see that the LDA model returns whole images without any meaningful segmentation of the elements in them, i.e., the LDA can characterize an image based on his content, but it can not segment it. On the other hand the A-LDA, which uses affinities information, returns regions that correspond to a single object in the images (see Fig. 7.11).

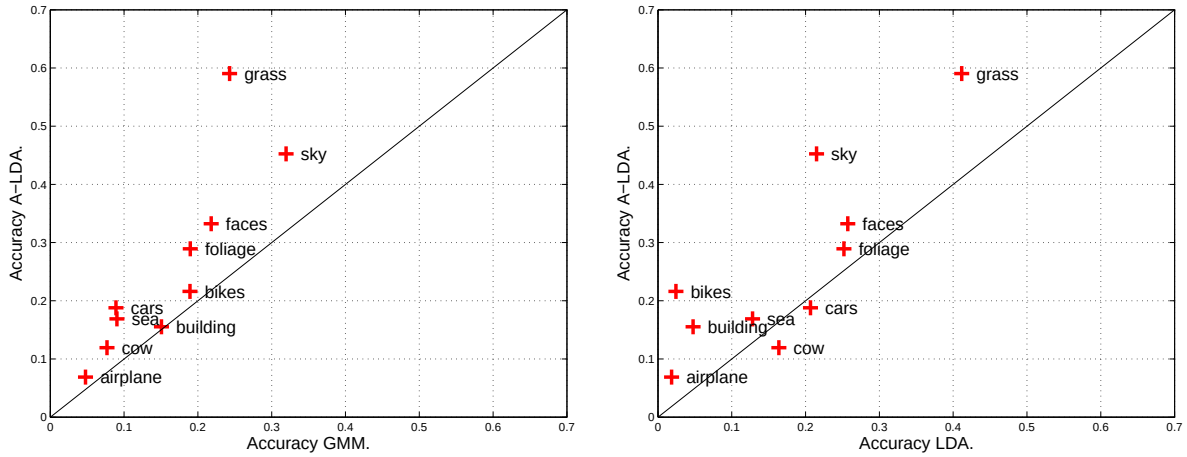


Figure 7.17: Left: scatter plot comparing the A-LDA model with a Gaussian mixture model (GMM)). We can see that the A-LDA model always outperforms the GMM. Right: scatter plot comparing the A-LDA model with an LDA model. In this case the A-LDA model has better accuracy for almost all categories. All of the three models use 20 segments and are unsupervised.

We compare our model (A-LDA) with the spatial latent Dirichlet allocation (S-LDA) model proposed by Wang and Grimson [WG07]. This model extends LDA by considering the proximity of visual words in an image, but without using information based on the local similarity of the image patches. Table 7.1 reports the detection/false alarm rates and the accuracy[4] of the two systems. In three out of the four categories reported in [WG07] we obtain higher accuracy and lower false alarm rates. For two categories: bikes and faces, we also have a higher detection rate. Furthermore, we report results on six categories ignored by [WG07].

Finally, we tested the A-LDA system on the more challenging MSRCv2 dataset. This dataset contains a total of 591 images and 23 categories[5]. Since this dataset is a super-

[4]The accuracy values for S-LDA were not reported in [WG07]. We estimated them from the detection and false alarm rates reported in [WG07] and ground truth by calculating for each category the number of true positive, false positive, false negative, and true negative.

[5]Two categories, horse and mountains. were not considered in the experiments because of the limited number of pixels with those labeled.

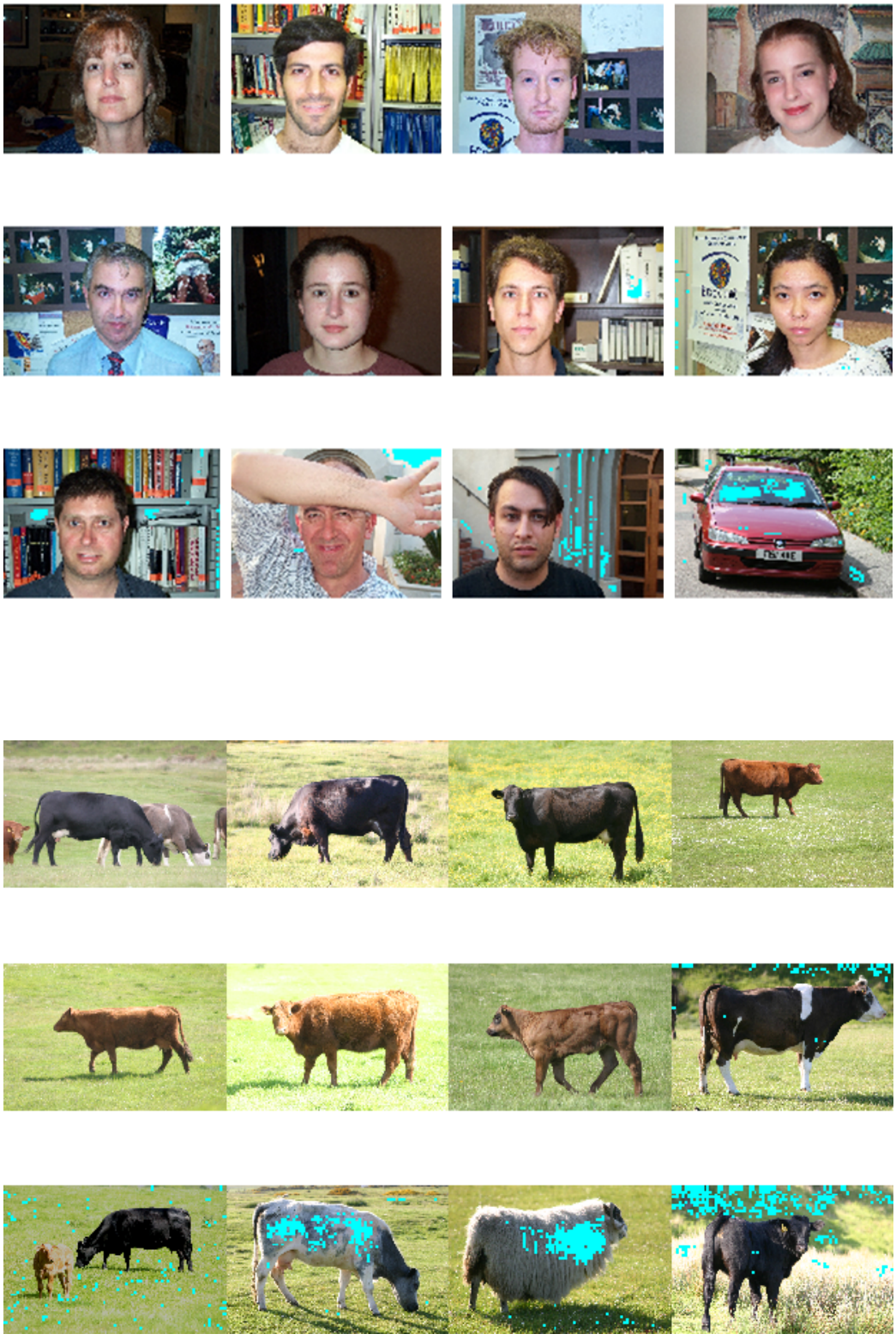


Figure 7.18: LDA results from MSRCv1. The top panel shows 12 segments from the category “faces”. The bottom pannel shows 12 segments from the category “cow”. See Fig. 7.11 for the corresponding results from the A-LDA model.

Class	**S-LDA (Wang et al.)**			**A-LDA**		
	Detection	False Al.	Accuracy*	Detection	False Al.	Accuracy
cows	0.5662	0.0334	0.3513	0.3796	0.1191	0.1193
grass	N/A	N/A	N/A	0.6910	0.0434	0.5904
cars	0.6838	0.2437	0.1381	0.2888	0.0331	0.1878
sea	N/A	N/A	N/A	0.3735	0.0087	0.1688
buildings	N/A	N/A	N/A	0.2884	0.1004	0.1552
foliage	N/A	N/A	N/A	0.5403	0.0852	0.2892
sky	N/A	N/A	N/A	0.5729	0.0271	0.4524
airplanes	N/A	N/A	N/A	0.2108	0.0539	0.0688
bikes	0.5661	0.3714	0.0672	0.6789	0.1072	0.2161
faces	0.6973	0.4217	0.0481	0.7038	0.0349	0.3323

Table 7.1: Comparison of our model (A-LDA) with the probabilistic model of Wang and Grimson (S-LDA) [WG07].

Model	Buil.	Grass	Tree	Cow	Sheep	Sky	Airpl.	Water	Face	Car	Bic.
A-LDA (v1)	16	60	29	12	X	45	7	17	33	19	22
A-LDA	11	61	32	10	4	39	3	20	22	6	32
LDA	4	47	8	6	5	22	7	16	24	6	0
[VT07]	52	87	68	73	84	94	88	73	70	68	74
[SWRC09]	62	98	86	58	50	83	60	53	74	63	75
[SJC08]	49	88	79	97	97	78	82	54	87	74	72

Model	Flower	Sign	Bird	Book	Chair	Road	Cat	Dog	Body	Boat
A-LDA (v1)	X	X	X	X	X	X	X	X	X	X
A-LDA	16	8	1	9	4	16	5	3	3	4
LDA	29	2	0	24	3	14	0	1	5	0
[VT07]	89	33	19	78	34	89	46	49	54	31
[SWRC09]	63	35	19	92	15	86	54	19	62	7
[SJC08]	74	36	24	93	51	78	75	35	66	18

Table 7.2: Segmentation accuracy (in percent) for the MSRCv2 dataset. The results are divided in two tables. The first row of each table reports the accuracy for the MSRCv1 dataset, a subset of the MSRCv2 dataset

set of the MSRCv1, we can also observe if and how much the segmentation accuracy of the A-LDA decreases when more categories need to be identified. Besides the usual comparison with the LDA model, we also consider the three supervised segmentation systems described in [VT07], [SWRC09], and [SJC08]; this comparison provides an upper bound on the performance of the system. The accuracy results are reported in Table 7.2. For both

the A-LDA model and the basic LDA we use a dictionary of 1024 visual words obtained from filter-bank descriptors and $K = 60$ topics in the model[6]. We ran the two inference algorithms (Gibbs sampling) for approximately the same amount of time. We observed that A-LDA outperformed the standard LDA for most categories, with the major exceptions of the categories flower and book (see second and third row of Table 7.2). Both unsupervised methods had considerable difficulties in recognizing and segmenting object categories like cat, boat and body. These categories have a wide range of variability and represent only a small fraction of the pixels in the collection so it is challenging to spot the statistical regularity of their appearance. The same categories are better handled when a certain amount of supervision is provided, as shown by the bottom three rows of Table 7.2. For all three methods both the visual words and the category model are built in a discriminative way. It is also interesting to compare results for the A-LDA model when applied to the MSRCv1 subset of images. We see that accuracy is lower for the more challenging MSRCv2. This is a consequence of the larger number of categories the system is trying to identify. For categories with a large number of observations, like grass, trees, bicycle, sky, and water the segmentation accuracy is comparable if not larger. For these categories, the dataset provides enough evidence for building a good statistical model. This observation is further analyzed in Section 7.5.

Of course, even for the grass category, (which is the largest in both the MSRCv1 and MSRCv2), the performance of the A-LDA is lower than the corresponding one for the supervised methods. These methods use a large amount of supervision, as seen [SJC08].

[6]We used a larger number of topics ($K = 60$) than we did for the MSRCv1 ($K = 20$) because of the larger number of categories in the dataset.

Those results were obtained using 276 training images with pixel level labeling. More complex datasets like the Pascal VOC were not considered for experimental evaluation. These datasets were designed to be challenging for supervised systems, and will be almost impossible for the unsupervised case[7]. Even a relatively "easy" dataset for supervised recognition, such as the MSRCv2, can be quite challenging for the unsupervised methods like A-LDA.

7.5 Accuracy vs. category sample size

Since our model (Fig. 6.1) is completely unsupervised, it has to rely on the co-occurrences of visual words w_{mn} to identify different categories. Therefore, we expect that the larger the number of pixels in a category, the higher the accuracy will be for that category, since there is more evidence to identify co-occurring visual words in that category. To verify this intuition we consider the MSRC dataset, remove all the images of faces, and progressively add new images from the faces category in the Caltech101 dataset[8]. In each iteration, we add a new batch of 10 images to the collection, then run our inference algorithm to obtain the categorical segments and compute the accuracies for the faces, as well as for all the other categories in the datasets.

Fig. 7.19 shows the mean accuracies of each category in the dataset for different numbers of pixels in the faces category[9]. As expected, the accuracy for the faces category,

[7] If the recognition accuracy is very low for all the unsupervised methods tested, it would be difficult to draw any conclusion.

[8] The 30 images in the MSRC dataset with face labels are a subset of the faces category of the Caltech101.

[9] We repeat this experiment 20 times. Each time we randomly select the batch of 10 images to add from the list of unused images.

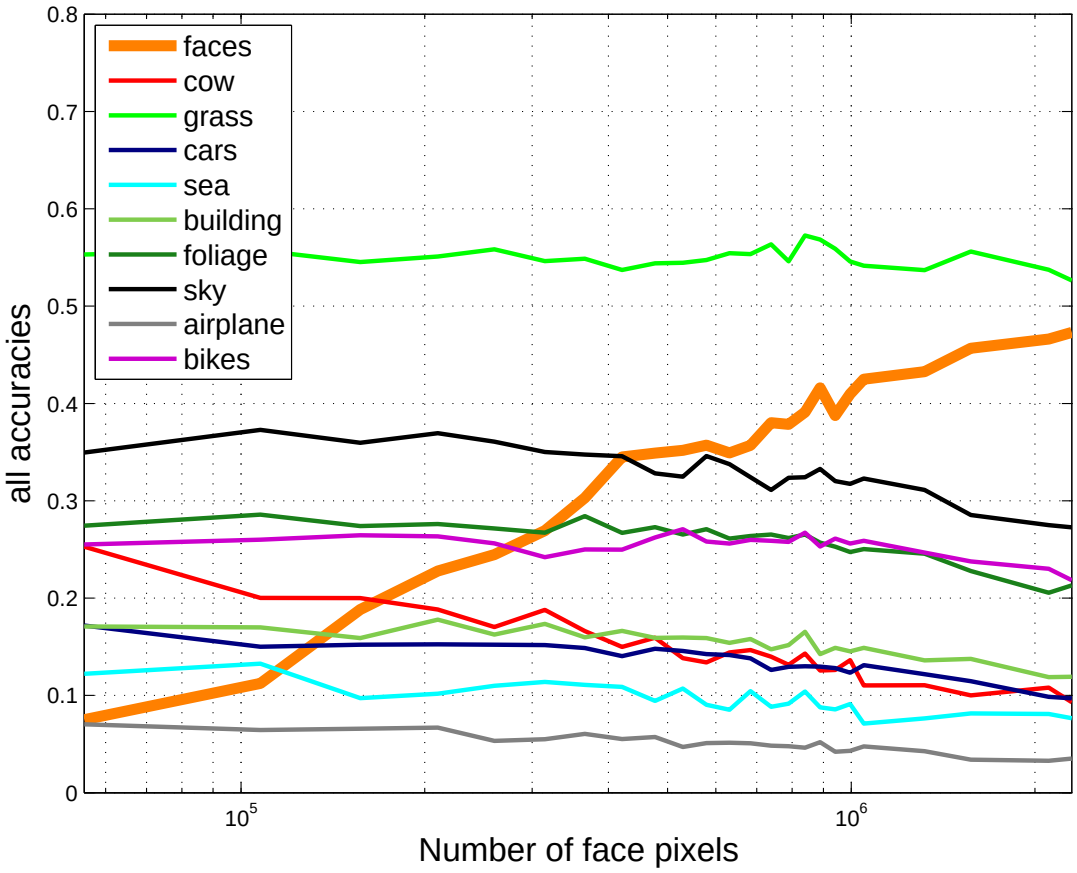


Figure 7.19: Accuracies of different classes as the size of the faces category in the collection increases. The accuracy for the faces category (solid orange) keeps improving as the size of this class increases. The accuracies of other categories like grass, foliage, and buildings are fairly constant. The accuracy for the cow category decreases as the number of pixels in the faces category increases, suggesting that it is more difficult to discriminate between these two categories given our visual words. We confirmed this effect by exploring segmentation results for individual images: the reddish cows are sometimes confused with pink-brown faces.

depicted with thick solid orange, increases as its size increases. In particular, the accuracy increases faster at the beginning, when the number of pixels is relatively small and slows down when the number of pixels is greater than 40000. The accuracies of the other categories are fairly stable, with the exception of a few categories which decrease as the face category becomes large. Among these exceptions the category which decreases the most is cows, with a drop of 0.16 in accuracy. This is due to the similarity between the visual words

distribution of the faces and cows categories. As the size of the faces category increases, the prior probability for a pixel to be a face also increases, leading our inference algorithm to label ambiguous pixels as faces instead of cows.

Chapter 8

Conclusions

We proposed a probabilistic model for simultaneously segmenting and recognizing consistent objects or object parts without the use of human supervision. Our system differs from previous work, which either cascaded or interleaved segmentation and recognition instead of integrating them into a single process. We first introduced a simple semi-parametric mixture model (SPMM) that can be used for single-image segmentation. With respect to other probabilistic models, such as GMM, this image-segmentation model has the advantage of allowing a more flexible representation of the segments composing an image. Our experiments on single-image segmentation show that in this context our model is superior to GMM. The same experiments show performance that, in this experimental scenarion, is comparable with normalized cuts. The advantage of our model is in providing a consistent probabilistic framework that can be easily extended to address more complex vision problems.

We extended the single-image model to approach the more challenging problems of simultaneous segmentation and recognition of an entire image collection, with limited or no supervision. We found that sharing information about the shape and appearance of a segment across a collection of images of objects belonging to the same category can

improve performance. To address the more general case of the simultaneous unsupervised segmentation and recognition of multiple categories in a collection of images, we further extended our model by also using visual words to describe recurring categorical segments in different images. The statistics of the visual words in each segment are shared across images, helping the segmentation process and automatically discovering recurring elements in the image collection. Our experiments show that our model (A-LDA model) outperforms other probabilistic models such as GMM, LDA, and S-LDA. We also show how a limited amount of supervision, namely the label of the object present in an image, can greatly improve the segmentation results. Finally, we studied the relation between the performance and the number of observations in a given category, and found that the accuracy increases with the number of observations.

In our experiments we considered observations sampled from a regular grid in the image. An alternative approach that can be pursued is the use of superpixels [RM03] as observations. This would result in a reduction of the number of observations and a corresponding speed up of the system. Three types of descriptors were used in our experiments: RGB color and filter bank responces, and SIFT [Low04]. Other types based on decision trees may be used to replace or supplement the ones used here.

Bibliography

[AMFM09] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. From contours to regions: An empirical evaluation. In *Computer Vision and Pattern Recognition (CVPR)*, Jun 2009.

[AMFM10] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2010.

[AT07] N. Ahuja and S. Todorovic. Learning the taxonomy and models of categories present in arbitrary images. In *IEEE International Conference on Computer Vision*, Rio de Janeiro, Brazil, 2007. IEEE Computer Society.

[AZMP07] M. Andreetto, L. Zelnik-Manor, and P. Perona. Non-parametric probabilistic image segmentation. In *International Conference on Computer Vision (ICCV)*, 2007.

[Bis06] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[BNJ03] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.

[BRCS07] T. Brox, B. Rosenhahn, D. Cremers, and H.-P. Seidel. Nonparametric density estimation with adaptive, anisotropic kernels for human motion tracking. In *Workshop on Human Motion in International Conference of Computer Vision (ICCV07)*, pages 152–165, 2007.

[BU02] E. Borenstein and S. Ullman. Class-specific, top-down segmentation. In *ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part II*, pages 109–124, London, UK, 2002.

[BU04] E. Borenstein and S Ullman. Learning to segment. In *Proc. 8th European Conference on Computer Vision (ECCV)*, May 2004.

[BZ05] A. Barbu and S.C. Zhu. Generalizing Swendsen-Wang to sampling arbitrary posterior probabilities. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 27(8):1239–1253, 2005.

[Can86] J. Canny. A computational approach to edge detection. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[Cas99] R. Casella. *Monte Carlo Statistical Methods*. Springer, 1999.

[CBGM02] C. Carson, S. Belongie, H. Greenspan, and J. Malik. Blobworld: Image segmentation using expectation-maximization and its application to image querying. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 24(8):1026–1038, 2002.

[CBS05] T. Cour, F. Benezit, and J. Shi. Spectral segmentation with multiscale graph decomposition. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, pages 1124–1131, Washington, DC, USA, 2005. IEEE Computer Society.

[CFF07] L. Cao and L. Fei-Fei. Spatially coherent latent topic model for concurrent object segmentation and classification. In *International Conference on Computer Vision (ICCV)*, 2007.

[CM02] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.

[Cri04] A. Criminisi. Microsoft research cambridge object recognition image database, version 1.0, 2004.

[DGK04] I. S. Dhillon, Y. Guan, and B. Kulis. Kernel k-means, spectral clustering and normalized. In *Proceedings of the 10th ACM SIGKDD Int. Conference on Knowledge Discovery and Data Mining (KDD04)*, 2004.

[DHS00] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2000.

[DS03] G. Dorkó and C. Schmid. Selection of scale-invariant parts for object class recognition. In *ICCV '03: Proceedings of the Ninth IEEE International Conference on Computer Vision*, page 634, Washington, DC, USA, 2003.

[EVGW+] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2009 (VOC2009) Results. http://www.pascal-network.org/challenges/VOC/voc2009/workshop/index.html.

[FFP05] L. Fei-Fei and P. Perona. A Bayesian hierarchical model for learning natural scene categories. In *CVPR '05: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05)*, pages 524–531, Washington, DC, USA, 2005.

[FH04] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2), 2004.

[FMM03] C. Fowlkes, D. Martin, and J. Malik. Learning affinity functions for image segmentation: Combining patch-based and gradient-based approaches. In *Computer Vision and Pattern Recognition CVPR*, Madison, WI, 2003.

[FPZ03] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. *Proc of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2003.

[FVS09] B. Fulkerson, A. Vedaldi, and S. Soatto. Class segmentation and object localization with superpixel neighborhoods. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2009.

[GG84] S. Geman and D. Geman. Stochastic relaxation, Gibbs distribution and the Bayesian restoration of images. *IEEE Transaction of Pattern Analysis and Machine Intelligence*, 6:721–741, 1984.

[GL91] G.H. Golub and C.F. Van Loan. *Matrix Computation*. John Hopkins University Press, 1991.

[Has70] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[JDK05] R. Jin, C. Ding, and F. Kang. A probabilistic approach for optimizing spectral clustering. In *Advances in Neural Information Processing Systems (NIPS)*, 2005.

[JF01] N. Jojic and B. Frey. Learning Flexible Sprites in Video Layers. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2001.

[Jor04] M. I. Jordan. Graphical model. *Statistical Science*, 19(1):140–155, 2004.

[KS01] S. Khan and M. Shah. Object based segmentation of video using color, motion and spatial information. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2001.

[KVV04] R. Kannan, S. Vempala, and A. Vetta. On clusterings: Good, bad and spectral. *J. ACM*, 51(3):497–515, 2004.

[LLS04] B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an implicit shape model. In *ECCV'04 Workshop on Sta-*

tistical Learning in Computer Vision, pages 17–32, Prague, Czech Republic, May 2004.

[LM01] T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *Int. J. Comput. Vision*, 43(1):29–44, 2001.

[LMH01] A. Lee, D. Mumford, and J. Huang. Occlusion models for natural images: A statistical study of a scale-invariant dead leaves model. *International Journal of Computer Vision*, 41(1-2):7–27, 2001.

[Low04] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, 2004.

[LSP05] S. Lazebnik, C. Schmid, and J. Ponce. A maximum entropy framework for part-based texture and object recognition. In *IEEE International Conference on Computer Vision (ICCV)*, volume 1, pages 832–838, Beijing, China, October 2005.

[Mar82] D. Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. Henry Holt and Co., Inc., New York, NY, USA, 1982.

[MBLS01] J. Malik, S. Belongie, T. Leung, and J. Shi. Contour and texture analysis for image segmentation. *International Journal of Computer Vision*, 43(1):7–27, 2001.

[MFTM01] 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. In *Proc of the 8th International Conference of Computer Vision*, pages 416–423, Jul 2001.

[MRR+53] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21(6):1087–1092, 1953.

[MS00] M. Meila and J. Shi. Learning Segmentation by Random Walks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 873–879, 2000.

[NJW01] A. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems (NIPS)*, 2001.

[OB07] P. Orbanz and J. M. Buhmann. Nonparametric Bayesian image segmentation. *International Journal of Computer Vision*, 2007.

[OT01] A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3), 2001.

[RES+06] B. C. Russell, A. A. Efros, J. Sivic, W. T. Freeman, and A. Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[Rij79] C. Van Rijsbergen. *Information Retrieval, 2nd ed.* Dept. of Comp. Sci. Univ. of Glasgow, England: Glasgow, 1979.

[RKB04] C. Rother, V. Kolmogorov, and A. Blake. "GrabCut": interactive foreground extraction using iterated graph cuts. *ACM Trans. Graph.*, 23(3):309–314, 2004.

[RM03] X. Ren and J. Malik. Learning a classification model for segmentation. In *9th IEEE International Conference on Computer Vision (ICCV 2003)*, pages 10–17, Nice, France, 2003.

[RVG+07] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie. Objects in context. In *ICCV*, Rio de Janeiro, 2007.

[SJ08] E. B. Sudderth and M. I. Jordan. Shared segmentation of natural scenes using dependent Pitman-Yor processes. In *Advances in Neural Information Processing Systems (NIPS)*, 2008.

[SJC08] J. Shotton, M .Johnson, and R. Cipolla. Semantic texton forests for image categorization and segmentation. In *Conference on Computer Vision and Pattern Recognition*, pages 1–8, jun 2008.

[SM00] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[SRE+05] J. Sivic, B. C. Russell, A. A. Efros, A. Zisserman, and W. T. Freeman. Discovering objects and their location in images. In *International Conference on Computer Vision (ICCV)*, 2005.

[SSM98] B. Scholkopf, A. J. Smola, and K. Muller. Nonlinear component analysis as a kernel eigenvalue problem, 1998.

[STFW05] E. B. Sudderth, A. Torralba, W. T. Freeman, and A. S. Willsky. Learning hierarchical models of scenes, objects, and parts. In *ICCV '05: Proceedings of the Tenth IEEE International Conference on Computer Vision*, pages 1331–1338, Beijing, China, 2005.

[SW87] R. H. Swendsen and J. Wang. Nonuniversal critical dynamics in Monte Carlo simulations. *Physical Review Letters*, 58(2):86–88, 1987.

[SWRC09] J. Shotton, J. M. Winn, C. Rother, and A. Criminisi. Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. *International Journal of Computer Vision*, 81(1):2–23, 2009.

[TA06] S. Todorovic and N. Ahuja. Extracting subimages of an unknown category from a set of images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'06)*, pages 927–934, New York, NY, USA, 2006.

[TJBB03] Y. Teh, M. Jordan, M. Beal, and D. Blei. Hierarchical dirichlet processes, 2003.

[TZ02] Z. Tu and S.-C. Zhu. Image segmentation by data-driven Markov chain Monte Carlo. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 24(5):657–673, 2002.

[UPH07] R. Unnikrishnan, C. Pantofaru, and M. Hebert. Toward objective evaluation of image segmentation algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2007.

[VJ04] P. Viola and M. J. Jones. Robust real-time face detection. *Int. J. Comput. Vision*, 57(2):137–154, 2004.

[VNU03] M. Vidal-Naquet and S. Ullman. Object recognition with informative features and linear classification. In *ICCV*, pages 281–288, 2003.

[VT07] J. Verbeek and B. Triggs. Region classification with Markov field aspect models. In *Conference on Computer Vision & Pattern Recognition*, pages 1–8, jun 2007.

[Was06] L. Wasserman. *All of Nonparametric Statistics*. Springer, 2006.

[WBBP10] P. Welinder, S. Branson, S. Belongie, and P. Perona. The multidimensional wisdom of crowds. In *Conference on Neural Information Processing Systems (NIPS)*, 2010.

[WCM05] J. Winn, A. Criminisi, and T. Minka. Object categorization by learned universal visual dictionary. In *ICCV '05: Proceedings of the Tenth IEEE International Conference on Computer Vision*, Beijing, China, 2005.

[WG07] X. Wang and E. Grimson. Spatial latent dirichlet allocation. In *Advances in Neural Information Processing Systems (NIPS)*, Vancouver, Canada, 2007.

[WT90] G. C. G. Wei and M. A. Tanner. A Monte Carlo implementation of the em algorithm and the poor man's data augmentation algorithms. *Journal of the American Statistical Association*, 85(411):699–704, 1990.

[WWP00] M. Weber, M. Welling, and P. Perona. Unsupervised learning of models for recognition. In *ECCV '00: Proceedings of the 6th European Conference on Computer Vision-Part I*, pages 18–32, London, UK, 2000.

[YS04] S. X. Yu and J. Shi. Segmentation given partial grouping constraints. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):173–183, 2004.

[Zip49] G. K. Zipf. *Human Behaviour and the Principle of Least-Effort*. Addison-Wesley, Cambridge, MA, 1949.

[ZMP05] L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1601–1608, 2005.

[ZS05] R. Zass and A. Shashua. A Unifying Approach to Hard and Probabilistic Clustering. In *IEEE International Conference on Computer Vision (ICCV)*, volume 1, pages 294–301, October 2005.