Unsupervised Statistical Models for General Object Recognition

by

Peter Carbonetto

B.Sc., McGill University, 2001

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

Master of Science

in

THE FACULTY OF GRADUATE STUDIES

(Department of Computer Science)

We accept this thesis as conforming
to the required standard

The University of British Columbia

August 2003

© Peter Carbonetto, 2003

Abstract

We approach the object recognition problem as the process of attaching meaningful labels to specific regions of an image. Given a set of images and their captions, we segment the images, in either a crude or sophisticated fashion, then learn the proper associations between words and regions. Previous models are limited by the scope of the representation, and performance is constrained by noise from poor initial clusterings of the image features. We propose three improvements that address these issues. First, we describe a model that incorporates clustering into the learning step using a basic mixture model. Second, we propose Bayesian priors on the mixture model to stabilise learning and automatically weight features. Third, we develop a more expressive model that learns spatial relations between regions of a scene. Using the analogy of building a lexicon via an aligned bitext, we formulate a probabilistic mapping between the image feature vectors and the supplied word tokens. To find the best hypothesis, we hill-climb the log-posterior using the EM algorithm. Spatial context introduces cycles to our probabilistic graphical model, so we use loopy belief propagation to compute the expectation of the complete log-posterior, and iterative scaling and iterative proportional fitting on the pseudo-likelihood approximation to render parameter estimation tractable. The EM algorithm is no longer guaranteed to converge with an intractable posterior, but experiments show the approximate E and M Steps consistently converge to a local solution. Empirical results on a diverse array of images show that learning image feature clusters using a standard mixture model, feature weighting using Bayesian shrinkage priors and spatial context potentials considerably improve the accuracy of general object recognition. Moreover, results suggest that good performance can be obtained without expensive image segmentations.

Contents

List of Figures

Acknowledgements

There are many people whose help and support must to be acknowledged: Most of all, my supervisor Prof. Nando de Freitas, for encouraging without pressuring and giving me a wonderful introduction to the world of research. Prof. David Lowe and Prof. Jim Little for being both excellent teachers and supervisors. Prof. Paul Gustafson for giving the Bayesian education that will never leave my side. Prof. Kobus Barnard for his generous assistance, rewarding discussions and provision of essential data. Yee Whye Teh for his discussions and thorough explanations on graphical models. Mark Paskin for lending a hand on the document style. Pantelis Elinas and Don Murray for creating the robot data. Kevin Murphy for his expertise on belief propagation. Eric Brochu, Kejie Bao, Natalie Thompson, Nando's gang and the rest of the students at the Laboratory for Computational Intelligence for being, most of all, friendly and entertaining people. And last but not least, my friends and family for teaching me everything that computer science does not.

PETER CARBONETTO

University of British Columbia
August 2003

Chapter 1

Introduction

Humans have the amazing capacity to recognise objects, but understanding the mechanism that produces such flexibility and accuracy continues to elude us. One popular view of object recognition is scene reconstruction — recognising 3D objects from 2D light signals [Marr, 1982]. In exploring this domain, we ask: What is a representation that can accurately describe a wide variety of objects? This very question occupies a great deal of research and debate in psychology and neuroscience. Some research suggests that humans do not have a single-purpose object recognition facility but instead have several specialised modules. Of late, the computer vision community has shown a great deal of interest in learning models for flexible object recognition (see Section 1.1). This work is certainly admirable because object recognition is a hard problem that requires harnessing a wide variety of resources for learning and image processing.

We maintain that it is helpful to consider a general object recognition model for learning and evaluation, independent of representation. The research strategy we adopt is a process of iterative refinement, from a naive framework to one that implements more structure, and understanding of low and high-level vision. As such, our attempts for general object recognition are still very much in the naive stage, so the precision we show in experiments will appear to be unremarkable. However, the consolation for poor performance is that we are tackling a hard problem with very few constraints.

We say an object is *recognised* when it is labelled with a concept in an appropriate and consistent fashion. This allows us to propose a somewhat satisfying answer to the question of what is an object: an object is a semantic concept (in our case, usually a noun) in an image caption. Our approach still fails to address the problem of object boundaries. For instance, does one define "tree" to be a single plant or the collection of plants making up a wooded area? This dilemma motivates a probabilistic representation of object boundaries in which we consider several hypotheses. We claim the models are a small advance toward a solution to this problem.

The goal is to construct a model that properly identifies, or annotates, an object in a scene. The objective can be construed as a classification problem: given training data composed of set of semantic concepts (the classes) and a set of object descriptions (the data points), learn a mapping from objects to concepts. There is some disagreement and a great deal of uncertainty as to what constitutes an *object*, which motivates a probabilistic representation of objects.

The facility of collecting data for training our object recognition models is an important consideration. The collection of supervised data by manually labelling semantically-contiguous regions of images is both time-consuming and problematic. For an autonomous agent, the user is required to tediously feed the agent with self-annotated regions of images. If we relax our requirement on training data acquisition by requesting captions at an image level, not at an image region level,

boxes fan backpack wall | person face shelves wall computer chair | boat water sky house trees | water boat

Figure 1.1: Examples of images paired with their captions. The two instances on the left are from the *Robomedia* data set, and the two on the right are from the *Corel* data set. A crucial point is that we do not know beforehand the labels on the objects in the scene, therefore the model has to learn which word belongs with which blob.

the acquisition of labelled data is suddenly much less challenging. Throughout this paper, we use manual annotations purely for testing only — we emphasize that the training data includes *only* the labels paired with images.

As a result, we are no longer exploring object recognition as a strict classification problem, and we do so at a cost since we are no longer blessed with the exact associations between objects and semantic concepts. In order to learn a model that *annotates*, *labels* or *classifies* objects in a scene, training implicates finding the *associations*, *alignments* or *correspondence* between objects and concepts in the data. Note that we use *annotation*, *labelling* and *classification* interchangeably. Accordingly, *association*, *alignment* and *correspondence* mean approximately the same thing.

With these thoughts in mind, the learning problem is considered to be semi-supervised or unsupervised. We use the latter methodology to develop our models. We adapt the work in another unsupervised problem — learning a lexicon from an aligned bitext in statistical machine translation [Brown *et al.*, 1993] — to the general object recognition problem, as first proposed in [Duygulu *et al.*, 2002].

We are given a set of images paired with image captions. Abstractly, a caption consists of a bag of semantic concepts that does a reasonably good job describing the objects contained in the image scene. For the time being, we restrict the set of concepts to English nouns (e.g. “face”, “toothbrush”, “floor”). This constraint limits the range of data available for training our models. For example, we cannot use news photo captions [Edwards *et al.*, 2003] without losing substantial information after text processing. Regardless, it is important to fix the spotlight on the computer vision aspect of the object recognition problem, and not get encumbered by the language understanding side. See Figure 1.1 for some examples of images paired with their captions.

Despite the restriction of concepts as English nouns, we still leave ourselves open to a great deal of equivocation and uncertainty, in part because nouns as object descriptions are ambiguous (e.g. “bank” is ground bordering a lake, river or sea and an establishment for the custody, loan, exchange or issue of money), redundant (e.g. one can use both “ocean” and “sea” to describe a body of water) and share multiple levels of specificity (e.g. “golden eagle”, “eagle”, “bird” and “animal”). In our data sets, we ensure each word has only one meaning, each noun refers to a single concept, and we align the captions to a consistent level of specificity. We try to make the terms specific enough so that the object recognition task is not too demanding (e.g. we use “bird” instead of “animal”) but not so precise that the undertaking is no longer interesting (we use the term “car” instead of “Volvo 244”).

Each image consists of a set of *blobs* that identify the objects in the scene. A *blob* is a set of

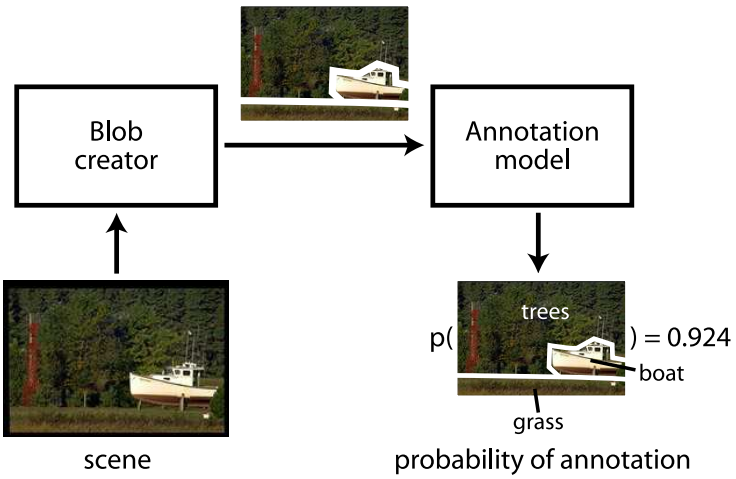

Figure 1.2: Diagram of the annotation process. The *blob creator* receives a scene as input and outputs a set of blobs that describe the objects contained in the scene. The *annotation model* is provided a set of blobs and returns a probability associated with each scene annotation. The diagram shows one possible labelling.

features that describes, more or less successfully, an object. In related papers [Barnard *et al.*, 2003b; Carbonetto *et al.*, 2003; Carbonetto and de Freitas, 2003; Duygulu *et al.*, 2002; Duygulu, 2003] we toss around similar terms *blob*, *patch*, *segment* and *image region*. In this work, we stick to the word *blob* since the last three terms imply that the scene is necessarily segmented, and this is an undue constraint on the problem formulation. For example, one could apply the work of template matching using local invariant point features [Lowe, 1999; Mikolajczk and Schmid, 2003]. For an overview of methods for object detection and representation, see [Forsyth, 2003].

Multiple blobs can refer to the same object in a scene. When the blobs are segmented image regions, we call this *over-segmentation*. Ideally, one would like a single blob per object, but this is not imperative for finding the correct associations. As we will see later on in Chapter 4, however, over-segmentation or a one-to-many mapping between objects and blobs can mislead our statistical models. In the final model, we introduce spatial relations which helps deal with over-segmented scenes.

The complete translation process, from an image composed of pixel values to a scene annotation, is illustrated in Figure 1.2. The learning objective, as elucidated above, is to construct a model from the training data that accepts a set of blobs and words as input and returns the probability that the blobs are annotated with the words. In other words, we want a model that gives us a good prediction of an annotation given a scene. What is meant by a "good" model prediction will be discussed at length in Chapter 4.

1.1 Related Work

A considerable body of research in computer vision is devoted to the exploitation of annotated image sets specifically for the purpose of image classification and clustering. Examples include [Smith and Chang, 1996; Huang and Zabih, 1998; Tieu and Viola, 2000]. The most direct and obvious application of image classification is image retrieval; once a classifier has learned to pair images of certain description (e.g. colour and texture histograms) with the conjoined text, the classifier can

function as a search engine. Inspiration for our work on object recognition comes from statistical models for large annotated image collections, which demonstrate that integrating semantic and visual information improves the performance on retrieval tasks [Barnard and Forsyth, 2001a; 2001b; Barnard *et al.*, 2003a]. Blei and Jordan [2003] propose generative latent variable models for representing the relationship between multiple types of data, namely images and captions. Work in [Duygulu *et al.*, 2002; Duygulu, 2003] takes the leap from image retrieval to object recognition by learning correspondences between words in the labels and regions within images.

Barnard *et al.* [2003b] have touched upon the issue of joint segmentation and object recognition, through the application unsupervised models that learn correspondences between words and pictures. They show that translation models can suggest appropriate blob merges based on word predictions. Additionally, experiments suggest that segmentation can play a critical role in performance, a question we examine in this work.

Fergus *et al.* [2003] do not use annotated data, but their proposed unsupervised model for object recognition is both interesting and relevant. They detect object classes using a representation by parts that implements scale-invariant point features. Their generative model learns both appearance and structure on reasonably-broad classes of objects. However, it assumes that objects are isolated in a scene, so it does learn correspondences.

Note

Portions of the material presented in the introduction and succeeding chapters appeared in the Ninth International Workshop on Artificial Intelligence and Statistics [Carbonetto *et al.*, 2003] and the NAACL Human Language Technology Conference Workshop on Learning Word Meaning from Non-Linguistic Data [Carbonetto and de Freitas, 2003].

Chapter 2

Background

The computer vision research we present relies heavily on techniques developed in the machine learning and Bayesian statistics communities. In this chapter we survey methods for statistical inference and learning in graphical models. Also, in Section 2.7 we set the scene for the object recognition models presented in Chapter 3.

2.1 Undirected graphical models

Undirected graphical models often have an intuitive and conveniently powerful application to problems such as segmentation, texture classification and object recognition, which makes them a popular probabilistic representation in the vision community. Undirected models are more flexible than their directed counterparts, but flexibility comes at a cost. Despite the explosion of research in machine learning, computation of undirected models tends to be expensive and the weak guarantees on convergence are unappetizingly restrictive. We show experimentally that the strengths of undirected graphical models outweigh their computational impediments, at least in the arena of object recognition.

Let $G = (V, E)$ denote an undirected graph with vertices V and edges E, and let $\mathcal{C}$ be the set of maximal cliques[1] over G. The non-negative potential $\psi_c(x_c)$ in the undirected graphical model represents the affinities between the nodes in $c \in \mathcal{C}$, where x_c is a subset of variables in the clique c. Thanks to the Hammersley-Clifford theorem, we can justifiably represent the probability density function as a normalised product of potentials

$$p(x) = \frac{1}{Z(\psi)} \prod_{c \in \mathcal{C}} \psi_c(x_c) \tag{2.1}$$

where $Z(\psi)$ is the partition function ensuring the distribution sums to unity and is defined as

$$Z(\psi) \triangleq \sum_{x} \prod_{c \in \mathcal{C}} \psi_c(x_c)$$

The partition function sums over all realisations of the variables x_c. Conditional independence in the undirected graphical model is very intuitive; we say that two subsets of nodes X_A and X_B are independent given the nodes X_C if and only if the subset of graph vertices C separate A and B.

[1] A **clique** is a fully connected subset of nodes. A clique is **maximal** if and only if there is no strict superset of nodes that is also fully connected.

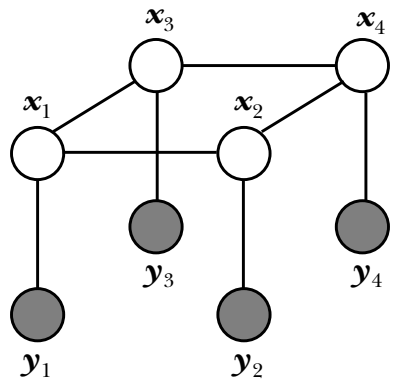

Figure 2.1: A sample Markov random network with $N = 4$. The observed variables are shaded. The potentials are defined over the undirected edges.

Equivalently, the Markov blanket of a single node x_i is merely the set of immediate neighbours $x_{\mathcal{N}_i}$, where $\mathcal{N}_i$ is the set of vertices adjacent to vertex i in the graph G.

There exist two complementary operations on graphical models, inference and learning. Inference is the estimation of the clique potentials given the model parameters, and learning is the estimation of the model parameters in accordance with the empirical distribution over the variables. When some of the variables are unobserved, which is the case with the models we present in this work, we need to manage both inference and learning simultaneously. The Expectation Maximisation (EM) algorithm handles both.

The bugbear of the undirected graphical model is the partition function. In general, the time required for exact inference is exponential in the number of variables since one cannot compute a marginal probability over a subset of variables without considering the marginals of all the other variables. One can resort to a local approximation, such as the Bethe free energy [Teh and Welling, 2002] or region graphs [Yedidia *et al.*, 2002], to reduce inference complexity. The trade-off for making such approximations is the risk of overlooking important interactions between distant variables, a characteristic of hard inference problems such as the satisfiability problem [Mézard and Zecchina, 2002]. The partition function handicaps learning as well. Using the ordinarily-reliable gradient ascent is tricky with undirected models because the gradient of a single parameter is intractable.

2.1.1 Markov networks

In the final model we propose, we introduce dependencies between segments in an image via a Markov random field (MRF). The topology is defined by an undirected graphical model with pairwise compatibility functions that represent local spatial dependencies and parameterised potentials that capture the relationship between the observed and latent, or underlying, variables. We adopt the structure proposed by Freeman *et al* [2000]. We assume the observed variables provide all the necessary information regarding the scene and long-range interactions are less important, and therefore a basic Markov network captures all the necessary dependencies.

The joint probability of the observed variables $y \triangleq \{y_1, ..., y_N\}$ and the unobserved variables representing the underlying scene $x \triangleq \{x_1, ..., x_N\}$ is given by

$$p(x, y) = \frac{1}{Z} \prod_{i \in V} \phi_i(x_i, y_i) \prod_{(i,j) \in \mathcal{C}} \psi_{ij}(x_i, x_j) \tag{2.2}$$

where $\mathcal{C}$ is the set of pairwise cliques in the MRF, V is the set of unshaded vertices, $N = |V|$ is the number of unshaded vertices in the graph, and ϕ_i and ψ_{ij} are pairwise potentials parameterising the scene.

2.2 Maximum likelihood

Parameter estimation using Expectation Maximisation (EM) and iterative scaling (IS) is framed using maximum likelihood.

In classical learning, the objective is to estimate a set of parameters $\widehat{\theta}$ that satisfy the *maximum a posteriori* (MAP) product over all instances of the observed data x:

$$\begin{aligned}\widehat{\theta}_{MAP} &= \operatorname*{argmax}_{\theta} \prod_{x} p(\theta \mid x)^{\widetilde{p}(x)} \\ &= \operatorname*{argmax}_{\theta} \prod_{x} p(x \mid \theta)^{\widetilde{p}(x)} p(\theta) \end{aligned} \tag{2.3}$$

$\widetilde{p}(x)$ is the probability that instance x will occur in the training data and $p(x \mid \theta)$ is the distribution of x parameterised by θ. In practice, it is more convenient to maximise the logarithm of the distribution

$$\widehat{\theta}_{MAP} = \operatorname*{argmax}_{\theta} \sum_{x} \widetilde{p}(x) \log p(x \mid \theta) + \log p(\theta) \tag{2.4}$$

Due to the monotonicity of the logarithm, we obtain the same objective function as (2.3) using the log posterior (2.4).

The MAP approach acknowledges that tuning the parameters too narrowly to the training data reduces global performance. The penalty component $p(\theta)$ is introduced to prevent over-fitting, hence the oft-used term *penalised maximum likelihood*. Equivalently, Bayesians view the $p(\theta)$ term as one's prior belief in the nature of θ. MAP learning educes to the maximum likelihood (ML) setting when the prior over the model parameters is uniform, and we get

$$\widehat{\theta}_{ML} = \operatorname*{argmax}_{\theta} \sum_{x} \widetilde{p}(x) \log p(x \mid \theta) \tag{2.5}$$

True Bayesian learning comprehends estimation of the posterior distribution $p(\theta \mid x)$, typically through application of stochastic methods such as Markov Chain Monte Carlo (MCMC) or Gibbs Sampling when the posterior cannot be derived analytically. There are strong arguments and theoretical results showing that a distribution over the model parameters is more advantageous than a point estimate [Tham, 2002]. The principal argument is that a point estimate is a major loss of information when put to the task of classification, and can be misleading when the true distribution is multi-modal or vague. Despite these contentions, we resort to classical learning because MCMC techniques are still prohibitively expensive on large models, and practical methods for learning distributions over graphs with cycles do not yet exist. In the future, we would like to reinvestigate some of the problems presented in this paper using Bayesian learning because it is necessary for rigorous analysis and generally leads to more robust classification performance.

2.3 Pseudo-likelihood

For most models, the maximum likelihood is hampered by the intractable partition function. An alternative to the maximum likelihood estimator is the *pseudo-likelihood*, which maximises local

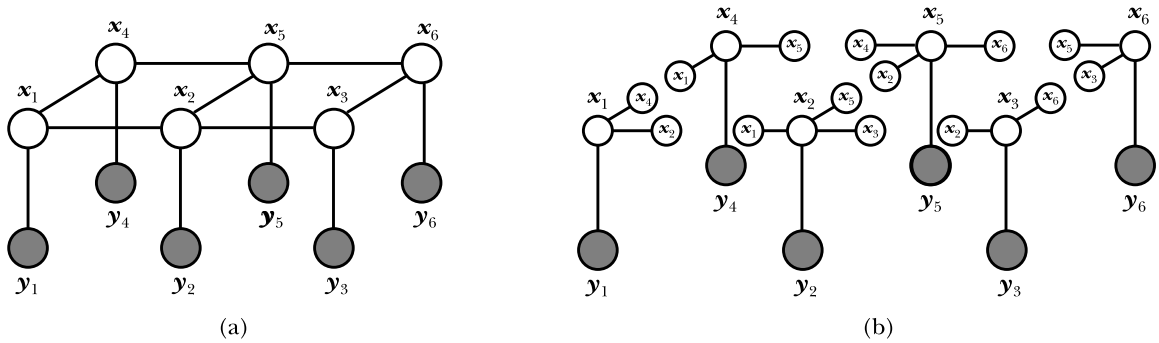

Figure 2.2: *(a)* A sample Markov random field with 6 sites. *(b)* The pseudo-likelihood representation of the MRF on the left.

neighbourhood conditional probabilities at sites in the MRF, independent of other sites. The conditionals over the neighbourhoods of the vertices allow the partition function to decouple and render parameter estimation tractable. However, because the estimate only optimises local joint probabilities, it fails to account for long-range dependencies.

Assuming a pairwise Markov network with observed and unobserved nodes (2.2), the pseudo-likelihood approximation is given by

$$p\ell(x, y) = \prod_{i \in V} \frac{1}{Z_i} \phi_i(x_i, y_i) \prod_{j \in \mathcal{N}_i} \psi_{ij}(x_i, x_j) \tag{2.6}$$

where Z_i is the partition function ensuring the distribution for the ith neighbourhood sums to unity. It is defined as

$$Z_i \triangleq \sum_x \sum_y \phi_i(x_i, y_i) \prod_{j \in \mathcal{N}_i} \psi_{ij}(x_i, x_j)$$

Essentially, the pseudo-likelihood is a product of undirected models, where each undirected model is a single latent variable x_i and its observed partner y_i conditioned on the variables in its neighbourhood, or Markov blanket. See Figure 2.2 for a concrete example.

The pseudo-likelihood does not completely decouple its neighbours since it captures first-order dependencies, but parameter estimation is now realisable using Newton-Raphson gradient ascent [Paget, 1999]. Cadez and Smyth [1998] prove that indeed the gradient of the pseudo-likelihood with respect to a global parameter is well-conditioned since it has a unique positive root. Instead of a basic application of gradient ascent, we found that using an iterative scaling bound was more straightforward to implement. The iterative scaling updates have a very slow rate of convergence but they reliably find a unique solution. There is a definite limitation to the pseudo-likelihood approximation, that being neglect of long-range interactions, but empirical trials demonstrate reasonable and consistent results [Seymour, 1993].

2.4 Belief propagation

Generally one presents variable elimination in decomposable graphical models as a motivation for the junction tree algorithm. However, our goal is only to outline the major points of belief propagation (BP), not lay out a complete derivation. The junction tree is an alternative representation of

probabilistic undirected graphical models that supplies a natural data structure for belief propagation. It keeps track of current beliefs on the maximal cliques as well as at intersections of the cliques, called *separators*. Local consistency conditions on the marginals engenders the well-known Hugin BP updates, where the separators play the role of messages passed between corresponding cliques. In lieu, one can parameterise the messages to remove explicit reference to separators and allay the need for separators. In turn, in some cases this dispels the need for the junction tree representation itself. This alternative method is the Shafer-Shenoy BP algorithm.

When the junction tree is singly connected[2] and satisfies the running intersection property[3] (local consistency implies global consistency), inference is exact using the collect and distribute-evidence protocols and requires at most $4N$ updates, where N is the number of nodes in the graph [Jordan, 2003]. When the graph contains cycles, and therefore no longer singly connected, belief propagation is not guaranteed to converge to the true distribution. However, experiments show that standard BP applied to graphs with cycles, dubbed "loopy BP", often produces surprisingly accurate estimates of the true distribution [Murphy *et al.*, 1999].

We can understand BP as the process of minimising the Gibbs free energy or the KL divergence. As it turns out, loopy BP converges to stationary points of an approximation called the Bethe free energy [Yedidia *et al.*, 2001; Teh, 2003]. In a similar vein, one can derive "generalised belief propagation" updates from sophisticated approximations using region graphs [Yedidia *et al.*, 2002]. The Bethe Free energy is a often a good approximation, so Teh and Welling [2002] took the logical step of deriving an algorithm that directly minimises it. Their unified propagation and scaling (UPS) algorithm assures convergence to a local minimum or saddle point of the Bethe Free energy and has favourable performance. However, for our purposes we found the naive loopy BP rarely diverged and is, computationally, a much cheaper alternative.

2.4.1 Junction trees

The junction tree is an enhancement of the undirected graph representation that includes potentials over separators. Let $\mathcal{C}$ be the set of cliques and let S be the separators, where each $s \in S$ is the intersection of two cliques. The probability distribution is given by

$$p(x) = \frac{\prod_{c \in \mathcal{C}} p_c(x_c)}{\prod_{s \in S} p_s(x_s)} \tag{2.7}$$

In essence, the junction tree represents the probability table $p(x)$ as a bunch of smaller tables over the cliques $p_c(x_c)$ separators $p_s(x_s)$. To respect the true probability $p(x)$, we need to ensure that the clique and separator tables are consistent.

For the time being, we suspend any concerns about graphs with cycles and assume that the junction tree satisfies the sufficient properties for correct belief propagation. As mentioned above, adopting BP to graphs with cycles is straightforward, as long as the incertitude of convergence is not a serious concern. Additionally, we skip over the discussion on the construction of junction trees through triangulation [Jordan, 2003] because the models handed to us already contain maximal cliques.

[2] A graph is **singly connected** if and only if there is exactly one path between each pair of clusters.

[3] A clique tree possesses the **running intersection property** if for each pair of cliques A and B containing the node i, all the cliques on the path between A and B also contain i.

2.4.2 Hugin belief propagation

The central tenet behind belief propagation is the responsibility to ensure consistency between neighbouring clique potentials. Based on this consideration, a single update from clique v to clique w at time t is given by equations

$$p_s^{(t)}(x_s) = \sum_{v \setminus s} p_v^{(t-1)}(x_v) \tag{2.8}$$

$$p_w^{(t)}(x_w) = \frac{p_s^{(t)}(x_s)}{p_s^{(t-1)}(x_s)} p_w^{(t-1)}(x_w) \tag{2.9}$$

where s is the unique separator for cliques v and w.

The Hugin algorithm consists of initialising the separators to unity, then iteratively passing information between cliques following a specified schedule. If the schedule satisfies the condition that a clique sends a message to one of its neighbours only after receiving messages from all its other neighbours, Hugin BP converges optimally on trees [Jordan, 2003]. The COLLECTEVIDENCE and DISTRIBUTEEVIDENCE protocol is such an instance.

2.4.3 Shafer-Shenoy belief propagation

In some cases, knowing the marginals of the separators is not particularly useful (after all, they are simply the marginals of the clique probabilities). The Shafer-Shenoy BP algorithm passes messages directly between cliques, so we can dispose of the separators and thereby the junction tree representation. Let s be the set of nodes at the intersection of cliques i and j, and let r be the set of nodes at the intersection of cliques i and k. The following equation updates the message from clique i to clique j at time t:

$$M_{ij}^{(t)}(x_s) = \sum_{x_i \setminus x_s} \psi_i(x_i) \prod_{(k \neq i) \cap (k \in \mathcal{N}_i)} M_{ki}^{(t-1)}(x_r) \tag{2.10}$$

where $\mathcal{N}_i$ is the set of cliques joined to clique i by an edge in the junction tree, or equivalently it is the set of cliques that have a non-empty intersection with clique i. Once a clique has received converged messages from all of its neighbours, the unnormalised marginal probability of a clique i is given by

$$p(x_i) \;\propto\; \psi_i(x_i) \prod_{k \in \mathcal{N}_i} M_{ki}(x_r)$$

For the most part, we stick to the Hugin belief propagation paradigm since it offers a more natural representation for unified belief propagation and scaling described in section 2.5.2.

2.5 Iterative scaling

Iterative scaling (IS) is a method for learning on fully-observed arbitrarily-parameterised undirected graphical models using the maximum likelihood framework. Iterative scaling is particularly desirable because it places few constraints on the model and guarantees convergence when inference is exact. Another advantage is that all the parameters are updated in parallel, so inference need only be performed once at every time step.

By and large, parameter estimation is difficult due to the partition function term in the log likelihood. Iterative scaling mitigates the problem by introducing a bound on the likelihood and then successively tightening the bound. On reasonably large models such as the ones we propose for general object recognition, however, the bounds are loose and convergence to a unique solution is painfully slow. To make learning practical, we perform iterative scaling updates on the pseudo-likelihood instead.

Consider a general exponential family model

$$p(x \mid \theta) = \frac{1}{Z(\theta)} \exp\left\{\sum_i \theta_i f_i(x)\right\} \tag{2.11}$$

where the $f_i(x)$ are arbitrary real-valued functions of x, the $\theta \triangleq \{\theta_i\}$ are the model parameters and $Z(\theta)$ is the normalisation factor given by

$$Z(\theta) = \sum_x \exp\left\{\sum_i \theta_i f_i(x)\right\}$$

We assume the features are non-negative ($f_i(x) \geq 0$ for all x) and the sum over the features for each instance x is defined by

$$f^{\#}(x) \triangleq \sum_i f_i(x) \tag{2.12}$$

Assuming a fully-observed process, the $\widetilde{p}(x)$ for each x are the empirical counts. The objective is to find a set of parameters that maximises the log likelihood

$$\begin{aligned} \widetilde{\ell}(\theta) &= \sum_x \widetilde{p}(x) \log p(x \mid \theta) \\ &= \sum_i \theta_i \sum_x \widetilde{p}(x) f_i(x) - \log Z(\theta) \end{aligned} \tag{2.13}$$

The log likelihood is intractable due to the partition function, so we decouple the parameters by bounding $\log Z(\theta)$ using the convexity result of the logarithm, and we take advantage of Jensen's inequality to decouple the terms in the exponential. Defining $\Lambda(\theta)$ to be the IS lower bound on $\widetilde{\ell}(\theta)$, the derivative of $\Lambda(\theta)$ with respect to a single parameter θ_i is

$$\frac{\partial \Lambda}{\partial \theta_i} = \sum_x \widetilde{p}(x) f_i(x) - \sum_x p(x \mid \theta) f_i(x) \exp\left\{\triangle\theta_i f^{\#}(x)\right\} \tag{2.14}$$

where $\triangle\theta_i \triangleq (\theta_i^{(new)} - \theta_i)$, θ_i is the current parameter estimate, and $\theta_i^{(new)}$ is the updated estimate [Berger, 1997]. $p(x \mid \theta)$ is the probability density given the current parameters θ.

If the feature counts are independent on the assignments of the variables, so $f^{\#}(x) = f^{\#}$, then solving for $\partial\Lambda/\partial\theta_i = 0$ gives us

$$\triangle\theta_i = \frac{1}{f^{\#}} \log\left(\frac{\sum_x \widetilde{p}(x) f_i(x)}{\sum_x p(x \mid \theta) f_i(x)}\right) \tag{2.15}$$

When the feature counts depend on the assignments, it is not possible to obtain a simple closed form equation for $\triangle\theta_i$.

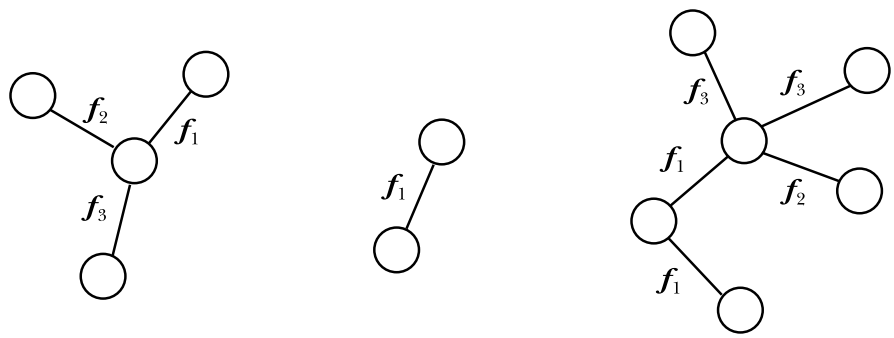

Figure 2.3: A collection of three disjoint undirected graphs with arbitrary features on the cliques. The feature counts are, from left to right, 3, 1 and 5. We note that we can use iterative proportional fitting instead of iterative scaling on feature f_2 since there is no more than one instance of the feature on each disjoint graph, assuming a fully-parameterised potential.

Now let's consider iterative scaling for the pseudo-likelihood (2.6). In such a situation, the neighbourhoods for the sites decouple. This is akin to having a product of N disjoint general exponential distributions such that the set of features are defined on the pairwise cliques of the MRF (in the succeeding notation we make this association implicit), and all the disjoint distributions draw from the same pool of features. As illustrated in Figure 2.3, each disjoint graph can have a different composition of features on its cliques. We define x_n to be the subset of variables in disjoint graph n. For the purposes of iterative scaling, the only significant difference between each disjoint graph is the feature count since the graphs may vary in size, and thus the number of cliques. We define $f_n^{\#}$ to be the sum over the i features in graph n, assuming independence from the assignments x_n. In the limiting case when a particular feature f_i is not present in a graph, we define $f_i(x_n)$ to be 0. The derivative of $\Lambda(\theta)$ with respect to a single parameter θ_i is

$$\frac{\partial \Lambda}{\partial \theta_i} = \sum_{n=1}^{N} \sum_{x_n} \widetilde{p}(x_n) f_i(x_n) - \sum_{n=1}^{N} \exp\left\{\triangle\theta_i f_n^{\#}\right\} \sum_{x_n} p(x_n \mid \theta) f_i(x_n) \tag{2.16}$$

In order to solve for $\triangle\theta_i$ when $\partial\Lambda/\partial\theta_i = 0$, we observe that equation (2.16) is the polynomial formula

$$A - \sum_{n=1}^{N} B_n x^{c_n} \tag{2.17}$$

where we set $A = \sum_{n=1}^{N} \sum_{x_n} \widetilde{p}(x_n) f_i(x_n)$, $B_n = \sum_x p(x_n \mid \theta) f_i(x_n)$, $c_n = f_n^{\#}$ and $x = \triangle\theta_i$. Therefore, updating θ_i is a matter of finding the positive root to (2.17). Cadez and Smyth [1998] prove that the positive root is unique for the pseudo-likelihood.

2.5.1 Iterative proportional fitting

When the potentials on the cliques are not arbitrary features but fully-parameterised probability tables, the better route is through iterative proportional fitting (IPF) [Jordan, 2003]. This coordinate ascent method converges faster than IS because it doesn't have to bound the gradient of the log likelihood. One disadvantage is that the potentials must be updated sequentially, so the computational bottleneck is the estimation of the marginals $p(x_c)$ given the current potential values.

The potential ψ defined on clique c at time t has the update rule

$$\psi^{(t)}(x_c) = \psi^{(t-1)}(x_c) \times \frac{\widetilde{p}(x_c)}{p(x_c \,|\, \psi^{(t-1)})} \tag{2.18}$$

As with Section 2.5, we turn to the case in which we have a product of disjoint probabilities with a common set of potentials, such as with the pseudo-likelihood (2.6). When the fully-parameterised potential ψ occurs in each of the disjoint graphs no more than once, as explained in Figure 2.3, then we can appeal to an IPF averaging update. Let N be the number of disjoint graphs, let x_{nc} be the variables in clique c of disjoint set n associated with the potential ψ and let $\widehat{x}$ be an assignment of x, then the update rule for the parameter ψ is

$$\psi^{(t)}(\widehat{x}) = \psi^{(t-1)}(\widehat{x}) \times \frac{\sum_{n=1}^{N} \widetilde{p}(X_{nc} = \widehat{x})}{\sum_{n=1}^{N} p(X_{nc} = \widehat{x} \,|\, \psi^{(t-1)})} \tag{2.19}$$

2.5.2 Universal propagation and scaling

Teh and Welling [2002] generalise inference and scaling on junction trees by deriving fixed point equations from the constrained maximum entropy. They show that their algorithm, called unified propagation and scaling (UPS), is guaranteed to converge in $2|S|$ propagation updates by visiting the cliques in a depth-first manner. $|S|$ is the number of separators in the junction tree.

The fixed point equations solving the constrained maximisation problem are the familiar belief propagation rule (2.8, 2.9) and the iterative proportional fitting update (2.18). The main idea behind the UPS algorithm is to perform an IPF update on a single potential, successively propagate the information to all its neighbours, and then repeat with the optimal depth-first scheduling.

2.6 Expectation Maximisation

So far, we have examined methods for learning the model parameters when all the variables are observed. Now we turn to a framework for estimating the parameters when the distributions of some of the variables are not known beforehand. We rely heavily on Expectation Maximisation (EM) in this work since all the translation models proposed have latent variables

The explanation in this section follows the unified derivation of EM as coordinate minimisation of the variational free energy, as presented in [Neal and Hinton, 1998; Teh, 2003]. This section also makes use of results presented in [Jordan, 2003]. A formalisation of EM using the variational free energy will turn out to be convenient for motivating approximate algorithms when the posterior distribution of the model parameters is intractable. Note that we observe the physics convention of minimising free energies as in [Teh, 2003], instead of [Neal and Hinton, 1998] where they maximise the negation of the variational free energy.

Let y be the observed variables and x be the unobserved, or latent, variables. From the data set we have an empirical distribution over the observed variables $\widetilde{p}(y)$. We also have a model distribution $p(x, y \,|\, \theta)$ parameterised by θ, and a prior $p(\theta)$ defined on the parameters. EM is a coordinate ascent algorithm for computing the *maximum a posteriori* (MAP) point sample

$$\begin{aligned} \widehat{\theta} &= \operatorname*{argmax}_{\theta} \sum_{y} \widetilde{p}(y) \log p(\theta \,|\, y) \\ &= \operatorname*{argmax}_{\theta} \sum_{y} \widetilde{p}(y) \log p(y \,|\, \theta) + \log p(\theta) \end{aligned} \tag{2.20}$$

bearing a provisional guarantee that the model parameter estimate $\widehat{\theta}$ is a local maximum of the incomplete log posterior. The log posterior (2.22) is simply a marginalisation of the complete log posterior (2.21) over the unknown variables:

$$\widetilde{\ell}_c(\widetilde{p}, \theta) \;=\; \widetilde{p}(x, y) \log p(x, y \,|\, \theta) + \log p(\theta) \tag{2.21}$$

$$\widetilde{\ell}(\widetilde{p}, \theta) \;=\; \sum_x \widetilde{\ell}_c(\widetilde{p}, \theta) \;=\; \widetilde{p}(y) \log p(y \,|\, \theta) + \log p(\theta) \tag{2.22}$$

Note that we can compute the log posterior (2.22), but not (2.21), since we do not have the empirical distribution $\widetilde{p}(x \,|\, y)$. Therefore, we will have to estimate it.

If we could observe the latent variables, then the learning algorithm would be straightforward. While it is unfortunate that we cannot observe the latent variables, this remark is the motivation for the EM algorithm: assuming for a moment that we *can* observe the latent variables, all we have to do is find a set of parameters $\widehat{\theta}$ that maximises $\widetilde{\ell}_c(\widetilde{p}, \theta)$. This is called the *Maximisation Step*, or M Step.

Choosing a set of latent variables that best satisfies the complete joint posterior is difficult because the joint generally does not decouple. If we compute estimates based on a lower bound of the complete posterior, however, the problem becomes easier to solve. In the *Expectation Step*, or E Step, we compute $p(x \,|\, y, \widehat{\theta})$ maximising the expectation of the lower bound of $\widetilde{\ell}_c(\widetilde{p}, \theta)$.

With the preceding motivation, we now formalise the EM algorithm using the generalised framework proposed by [Neal and Hinton, 1998]. The variational free energy of the posterior $p(\theta \,|\, x, y)$ is defined as

$$\mathcal{F}_{var}(\widetilde{p}, \theta) = -\mathbb{E}_{\widetilde{p}} \log p(x, \theta \,|\, y) + \mathbb{E}_{\widetilde{p}} \log \widetilde{p}(x) \tag{2.23}$$

$$= KL\left(\widetilde{p}(x) \parallel p(x \,|\, y, \theta)\right) - \log p(\theta \,|\, x) \tag{2.24}$$

where $\mathbb{E}_{\widetilde{p}} \log \widetilde{p}(x)$ is the entropy for the estimate $\widetilde{p}(x)$ and the Kullback-Leibler divergence is defined by

$$KL(q(x) \parallel p(x)) = \int q(x) \log \frac{q(x)}{p(x)} dx$$

The $\widetilde{p}(x)$ in (2.23) and (2.24) is the current guess of the distribution of the latent variables. As we will show next, the best guess is always $\widetilde{p}(x) = p(x \,|y, \theta)$.

The EM algorithm starts with an initial guess of the model parameters, $\theta^{(0)}$. By iterating on the E Step and M Step, the algorithm is guaranteed to converge to a local minimum of the variational free energy, regardless of the initialisation. $\theta^{(t)}$ is the estimate of the parameters at time t and $\widetilde{p}^{(t)}(x)$ is the estimate of the latent variables at time t. The E Step minimises the variational free energy $\mathcal{F}_{var}(\widetilde{p}, \theta)$ with respect to $\widetilde{p}$ and the M Step minimises $\mathcal{F}_{var}(\widetilde{p}, \theta)$ with respect to θ.

The second term in (2.23) does not depend on θ. Therefore, minimising $\mathcal{F}_{var}(\widetilde{p}, \theta)$ with respect to θ in the M Step is equivalent to maximising $\mathbb{E}_{\widetilde{p}} \log p(x, \theta \,|\, y)$, the expected complete log posterior. Similarly, the only term in equation (2.24) that depends on $\widetilde{p}(x)$ is the first. Taking the partial derivative of the variational free energy with respect to the components of $\widetilde{p}(x)$ and keeping θ constant, we get

$$\frac{\partial \mathcal{F}_{var}}{\partial \widetilde{p}(x)} = \log \widetilde{p}(x) - \log p(x \,|\, y, \theta) - 1$$

Applying the normalising constraint $\sum_x \widetilde{p}(x) = 1$, the unique solution is $\widetilde{p}(x) = p(x \mid y, \theta)$. Using these results, we can now present the complete EM algorithm.

E Step Compute distribution $\widetilde{p}^{(t)}(x)$ over the range of x such that $\widetilde{p}^{(t)}(x) = p(x \mid y, \theta^{(t-1)})$.

M Step Compute $\theta^{(t)}$ using the expected complete log posterior, so $\theta^{(t)} = \operatorname*{argmax}_{\theta} \mathbb{E}_{\widetilde{p}^{(t)}} \log p(x, y \mid \theta) + p(\theta)$.

When calculation of the latent variable marginals and the gradient of the estimated complete log posterior is exact, EM is guaranteed to converge to a local minimum of the incomplete log posterior. If the methods are only approximation, we lose all guarantees on convergence. Such is the case for the last model we propose in this work since $p(x \mid y, \theta)$ and $p(x, \theta \mid y)$ are both intractable. Despite the lack of theoretical support, in experiments we show dependent convergence. If we could find a reasonable approximation to the variational free energy, we could ensure convergence on intractable models. Unfortunately, no such unifying approximation currently exists.

2.7 Object recognition as machine translation

We can cast generic object recognition as a machine translation problem, as originally proposed in [Duygulu *et al.*, 2002] and elaborated further in [Duygulu, 2003]. Given a text in two languages (such as the Canadian parliamentary Hansard bitext which is published in both England and French) and assuming we known the correspondences at roughly a sentence level, the objective in machine translation is to learn a dictionary of precise translations from English to French. The bilingual dictionary, called a *lexicon*, is the model. Learning a lexicon is a standard problem in machine translation. Analogously, in object recognition we build a lexicon that predicts one representation (words) given another representation (blobs), and the aligned corpus is a set of images paired with word captions.

The problem of lexicon acquisition involves determining the precise correspondences between the words of the languages in the aligned sentences — we know the caption text goes with the image, but we don't know beforehand which word goes with which blob. Since we are not blessed with the exact associations between blobs and words, the image classification problem is unsupervised. (Alternatively, one can tackle the translation problem with a constrained semi-supervised classification approach, whereupon the exact classes are sampled from a bag of labels for each document [Andrews *et al.*, 2002; 2003].) For a single training image and a particular word token, we have to learn both the probability of generating that word given an object description and the correct association to one of the regions within the image. In this section we outline a general probabilistic framework for object recognition as machine translation based on the translation models proposed in [Brown *et al.*, 1993].

First we introduce some notation. The observed variables are the words $w_{n1}, \ldots, w_{nL_n}$ and the blobs $b_{n1}, \ldots, b_{nM_n}$ paired with each image (or document) n. M_n is the number of blobs or regions in image n, and L_n is the size of the image caption. The quantity earmarked for learning is the word-to-blob translation parameterisation θ. We also have another set of unknown quantities, the latent alignment variables. While we don't require the alignment variables to define the log posterior, they will be handy for computing the model parameters given the known quantities. For each blob b_{nj} in image n, we need to align it to a word from the attached caption. This unknown association is represented by the variable a_{nj}, such that $a_{nj} = i$ if and only if blob b_{nj} corresponds to word w_{ni}.

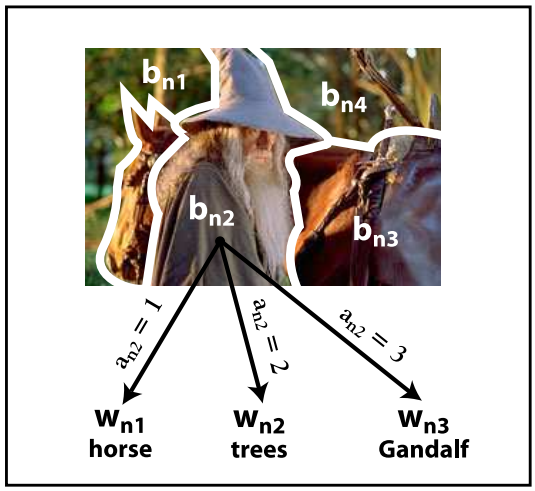

Figure 2.4: The alignment variables represent the correspondences between the words and blobs. In this case, the correct value for a_{n2} is 3.

Additionally, we define $a^i_{nj} = 1$ if $a_{nj} = i$; otherwise, $a^i_{nj} = 0$. Therefore, $p(a^i_{nj}) = p(a_{nj} = i)$ is the probability that blob b_{nj} is aligned with word w_{ni} in document n. See Figure 2.4 for an illustration of the alignment process. The sets of words, blobs and alignments for all documents are denoted by w, b and a, respectively. The directed and undirected graphical models for this distribution are shown in Figure 2.5.

We assume words w_{ni} are discrete tokens from the set $\{1, \ldots, W\}$, where W is the total number of word tokens. Each word token represents a separate semantic concept or object. The blob tokens can be either treated as discrete tokens or more faithfully represented in a continuous space. Each b_{nj} is a vector of features in the blob description space $\mathbb{R}^F$ where F is the number of features. We use simple colour features to represent the blobs, although the object recognition framework we outline in this section is independent of any choice of blob representation.

The goal is to find the set of parameters that best satisfies the distribution of the observed variables. Ideally, one should estimate the distribution of the parameters given the data, but given that we are dealing with large and often intractable models, it is more realistic to find a single parameter at one of the modes using EM. The drawback is that our models — with the exception of the first discrete model which is convex — tend to be highly multi-modal and so a single point sample is deceptive and can introduce artificial variance in the results.

Without loss of generality, the joint distribution of the alignment variables and parameters given the observed variables is defined as

$$\begin{aligned}
p(\theta, a \,|\, b, w) &\propto p(\theta)\, p(b, w, a \,|\, \theta) \\
&= p(\theta)\, p(b, a \,|\, w, \theta)\, p(w) \\
&= p(\theta) \prod_{n=1}^{N} p(b_{n1:M_n}, a_{n1:M_n} \,|\, w_{n1:L_n}, \theta) \\
&= p(\theta) \prod_{n=1}^{N} \prod_{j=1}^{M_n} p(a_{nj} \,|\, a_{n1:j-1}, b_{n1:j}, w_{n1:L_n}, \theta)\, p(b_{nj} \,|\, b_{n1:j-1}, a_{n1:j-1}, w_{n1:L_n}, \theta)
\end{aligned} \tag{2.25}$$

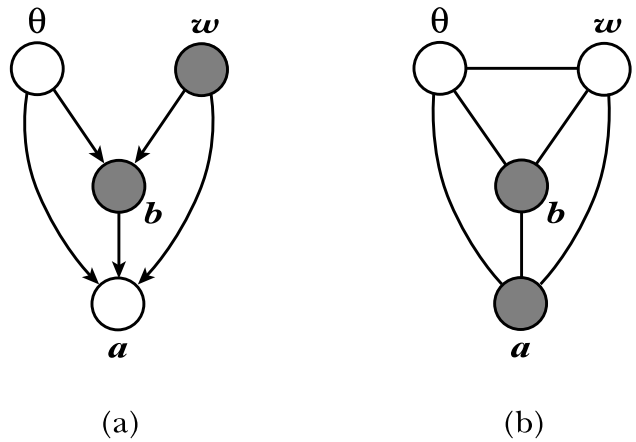

Figure 2.5: *(a)* The directed graphical model for the observed variables, latent variables and model parameters. The observed variables are shaded and the arrows represent cause-and-effect. We have omitted the subscripts on the variables. *(b)* The undirected graphical model, found by moralising the directed graph.

where $x_{ni:j}$ is the set of variables $\{x_{ni}, \ldots, x_{nj}\}$ or the empty set if $i > j$. Some notes about the above equations: on the first line we apply Bayes' Theorem to derive the posterior from the prior and likelihood; the second line uses the definition of conditional probability; since we don't have any prior knowledge about the probability of a word, we assume a uniform distribution and $p(w)$ drops from the distribution on the third line.

Learning also involves computation of the incomplete posterior $p(\theta \,|\, b, w)$. The incomplete posterior can be found by marginalising the latent variables, $p(\theta \,|\, b, w) = \sum_a p(\theta, a \,|\, b, w)$.

There's one missing link in the translation model presented so far. We've defined a lexicon that predicts blobs given words, but how do we predict words in a scene? We appeal to Bayes' Theorem and use the fact that the prior probability $p(w)$ is uniform over all words. The probability of a scene annotation for a new document $N + 1$ is given by

$$\begin{aligned} p(w \,|b, \theta) &= \frac{p(b \,|\, w, \theta)\, p(w)}{\sum_w p(b \,|\, w, \theta)\, p(w)} \\ &= \frac{p(b \,|\, w, \theta)}{\sum_w p(b \,|\, w, \theta)} \end{aligned} \tag{2.26}$$

The above equations also use the notation b and w to refer to all the blobs and words in the new document $N + 1$. Note that we could just as easily write the joint probability as the translation from blobs to words, instead of the other way around as we've done above. In our words-to-blobs model, the advantage is that the E Step in Expectation Maximization is easier because the alignments are over the words, safely assuming there are less word tokens than blobs. On the other hand, the disadvantage is we have to find the marginals over the blobs in the M Step when the joint distribution is intractable. We do not gain or lose anything with a blobs-to-words model because the advantages are reversed.

One shortcoming of the translation model is that it does not know now to deal with regions not represented by any of the words. This incapacity introduces noise. Duygulu *et al.* propose a model with NULL translation probabilities demonstrating some success in experimental results, but it introduces another parameter to tune, the NULL threshold [Duygulu *et al.*, 2002; Duygulu, 2003].

Chapter 3

Unsupervised models for object recognition

We propose four unsupervised statistical translation models for the application of object recognition, *dML1*, *gML1*, *gMAP1* and *dML1MRF*. Model *dML1* is the most basic model and closely honours the first machine translation model of [Brown *et al.*, 1993]. It has a discrete representation of blobs and assumes independence between objects in a scene. It was first introduced by [Duygulu *et al.*, 2002] and we include it for completeness. Model *gML1* progresses from *dML1* by incorporating blob clustering as part of model learning. Model *gMAP1*, proposed in [Carbonetto *et al.*, 2003], introduces priors on the cluster parameters to stabilise learning and automatically penalise irrelevant features. Model *dML1MRF* increases the expressiveness of *dML1* by learning spatial relations between objects. We compare the precision of the models on object recognition tasks in Section 4.

3.1 Model *dML1*

First proposed by Duygulu *et al.* [2002], *dML1* is the most basic model we present and most closely resembles the statistical machine translation IBM Model 1 of [Brown *et al.*, 1993]. This model is characterised by a blob representation using discrete tokens, an independence assumption on the objects in a scene and uniform priors on the parameters.

The *dML1* model introduces structure to the general translation equation (2.25) by making a number of independence assumptions. Generally speaking, an alignment between a word and blob depends on the other alignments in the image, simply because objects in a scene are not independent of each other. To ensure tractability of model *dML1*, we presuppose that each blob and its corresponding alignment variable are independent of all the other blobs and alignments in the image. We also assume the caption words are independent given the alignments. Accordingly, the probabilities simplify to

$$\begin{aligned} p(a_{nj} \,|\, a_{n1:j-1}, b_{n1:j}, w_{n1:L_n}, \theta) &= p(a_{nj} \,|\, b_{nj}, w_{n1:L_n}, \theta) \\ p(b_{nj} \,|\, b_{n1:j-1}, a_{n1:j-1}, w_{n1:L_n}, \theta) &= p(b_{nj} \,|\, a_{nj}, w_{n1:L_n}, \theta) \end{aligned}$$

The advantage of treating the blobs as discrete variables is that it allows one to directly exploit the IBM Model 1 translation model [Brown *et al.*, 1993]. It is also well known that the log likelihood of this model is convex, so the local maximum is also the global maximum and EM will produce the same parameter estimate independent of the initialisation of the model parameters. The translation lexicon is a $B \times W$ discrete probability table with entries $t(b^\star|w^\star)$, where b is the total number of

blob tokens, W is the number of word tokens, $w^\star$ denotes a particular word token and $b^\star$ denotes a particular blob token. Another consequence is that the E Step is trivial because the unnormalised latent alignment probabilities are equal to the translation probabilities. We use k-means to assign each blob b_{nj} in the feature space $\mathbb{R}^F$ to one of the B clusters. When B is greater than W, the translation model acts as a non-linear classification.

The down side is that the artificial discretisation of the blobs results in a significant loss of information and contributes noise. Since we rely solely on low-level image features such as colour and texture to cluster the blobs, it is prone to placing blobs from the same object in separate clusters. More irreparably, it tends to put several objects in the same cluster because the features are insufficient. Model *dML1MRF* remedies these problems by introducing dependencies between the blobs. The advantage of the discrete model is the introduction of a transparent nonlinear classification, the disadvantage is that the noise introduced through discretisation aggravates the problem of finding the correct associations.

The number of blob clusters, B, is a significant factor in the success of the translation model. As a rule of thumb, we found $B = 5W$ to work well. If there are too few clusters, the classification is non-separable. On the other hand, if B is too large than finding the correct associations becomes near-impossible and the classification over-fits on the training data. The discretisation of the blobs would improve with access to the information in the image labels, which is exactly what we do in the next model, *gML1*.

Given the independence assumptions, the complete likelihood of model *dML1* is

$$\begin{aligned} p(\theta, a \mid b, w) &= \prod_{n=1}^{N} \prod_{j=1}^{M_n} p(a_{nj} \mid b_{nj}, w_{n1:L_n}, \theta)\, p(b_{nj} \mid a_{nj}, w_{n1:L_n}, \theta) \\ &= \prod_{n=1}^{N} \prod_{j=1}^{M_n} \prod_{i=1}^{L_n} t(b_{nj} \mid w_{ni})^{a_{nj}^i} \end{aligned} \tag{3.1}$$

where we define $\theta \triangleq \{t\}$ and

$$\begin{aligned} p(a_{nj} \mid b_{nj}, w_{n1:L_n}, \theta)\, p(b_{nj} \mid a_{nj}, w_{n1:L_n}, \theta) &\triangleq \prod_{i=1}^{L_n} p(b_{nj} \mid w_{ni}, \theta)^{a_{nj}^i} \\ p(b_{nj} \mid w_{ni}, \theta) &\triangleq t(b_{nj} \mid w_{ni}) \end{aligned}$$

We use EM to compute an ML estimate for the model parameters. The E Step equation is

$$\widetilde{p}(a_{nj} = i) = \frac{t(b_{nj} \mid w_{ni})}{\sum_{k=1}^{L_n} t(b_{nj} \mid w_{nk})} \tag{3.2}$$

and the M Step update for parameter t is

$$t^{(new)}(b^\star | w^\star) = \frac{\sum_{n=1}^{N} \sum_{j=1}^{M_n} \sum_{i=1}^{L_n} \delta(b_{nj} = b^\star)\, \delta(w_{ni} = w^\star)\, \widetilde{p}(a_{nj} = i)}{\sum_{n=1}^{N} \sum_{j=1}^{M_n} \sum_{i=1}^{L_n} \delta(w_{ni} = w^\star)\, \widetilde{p}(a_{nj} = i)} \tag{3.3}$$

The complete derivation of the E and M Steps is carried out in Appendix B.1.

3.2 Model *gML1*

Model *gML1* incorporates clustering of the blobs into the translation, which eliminates the problem of a poor initial clustering faced in model *dML1*. Model *gML1* uses a standard Gaussian mixture and assumes the same dependence structure outlined in Section 3.1.

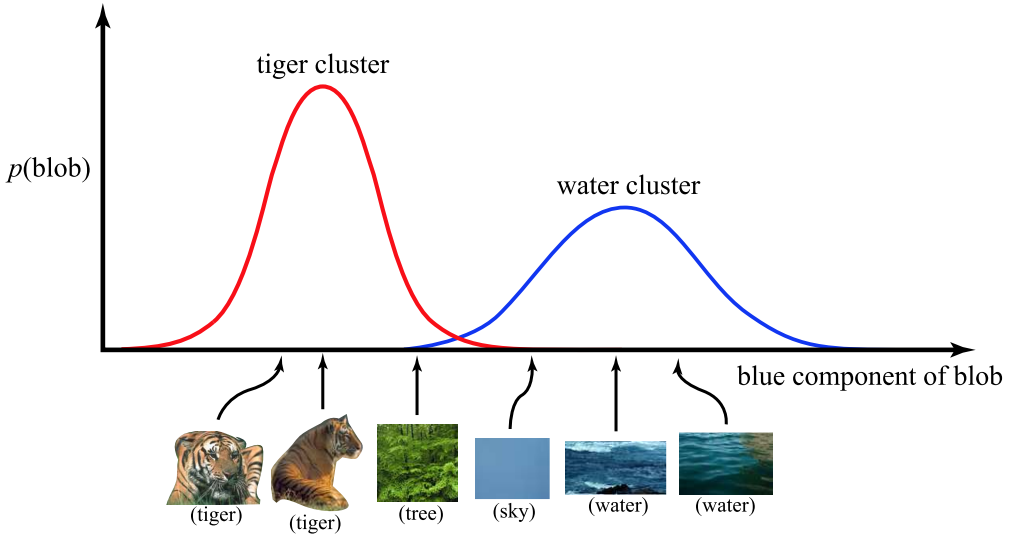

Figure 3.1: A fabricated example with two words, “tiger” and “water”. The graph shows a slice of the Gaussian distributions on the dimension of the blob feature space. The horizontal axis is the average amount of blue in the blob and the vertical axis is the probability of the blob occurring in a particular cluster. Note that certain concepts such as “sky” and “water” are indistinguishable in the depicted dimension because they both have approximately equal amounts of blue.

For a particular semantic concept c, we posit that the associated blobs are distributed normally in the space of features $\mathbb{R}^F$. The model parameters are the mean μ_c and covariance Σ_c for a each word c in the training set. We use EM to iteratively update the associations between the blobs and words and adjust each word cluster to the appropriate location. For an illustration of translation rules as normal distributions, see Figure 3.1.

One of the immediate shortcomings of the model with Gaussian translation rules is that the log likelihood surface is very bumpy. Plainly speaking, the EM algorithm is prone to converge to one of the many local maximums and only rarely the global maximum of the distribution. This is contrary to the discrete model. By virtue of having a convex log likelihood surface, EM on the discrete translation model is always guaranteed to hit upon the unique maximum. In actual fact, the local maximum in model *dML1* exists in the k-means discretisation step. In this respect, we argue the Gaussian translation model still poses an advantage because it lends itself to finding the global maximum using, for instance, and MCMC sampler.

In theory, when the features are adequate for representing the objects in a scene, the Gaussian assumption is fairly safe. However, finding such a set of features is a very difficult task, which means that in practice the concepts in the data set can take on distributions that are far from being Gaussian. Linear classification can turn out to be a severe handicap when the objects are not linearly separable. The larger the training sets, the more variation in the objects and noise in the blob features, and accordingly the more the covariances of the clusters increase and spread out over the space of the blobs. The unfortunate consequence is that the *gML1* model is not scalable to large datasets. We retain the somewhat naive Gaussian assumption for simplicity and ease of computation, but in the future we would like to examine the application of nonlinear unsupervised classification models such as hierarchical mixture models [Barnard *et al.*, 2003a] or mixtures-of-experts [Jacobs *et al.*, 1991].

For this model, we do not presume any prior knowledge on the model parameters. We place a

uniform distribution over θ, and consequently the prior cancels out from the joint.

Putting it all together, the set of unknown model parameters is $\theta \triangleq \{\mu_1, ..., \mu_W, \Sigma_1, ..., \Sigma_W\}$ and, referring to equation 2.25, the joint of the observed and unobserved variables for model *gML1* is

$$\begin{aligned} p(\theta, a \,|\, b, w) &= \prod_{n=1}^{N} \prod_{j=1}^{M_n} p(a_{nj} \,|\, b_{nj}, w_{n1:L_n}, \theta)\, p(b_{nj} \,|\, a_{nj}, w_{n1:L_n}, \theta) \\ &= \prod_{n=1}^{N} \prod_{j=1}^{M_n} \prod_{i=1}^{L_n} \prod_{c=1}^{W} \mathcal{N}(b_{nj} \,|\, \mu_c, \Sigma_c)^{\delta(w_{ni}=c)\, a_{nj}^{i}} \end{aligned} \tag{3.4}$$

where we define

$$\begin{aligned} p(a_{nj} \,|\, b_{nj}, w_{n1:L_n}, \theta)\, p(b_{nj} \,|\, a_{nj}, w_{n1:L_n}, \theta) &\triangleq \prod_{i=1}^{L_n} p(b_{nj} \,|\, w_{ni}, \theta)^{a_{nj}^{i}} \\ p(b_{nj} \,|\, w_{ni}, \theta) &\triangleq \mathcal{N}(b_{nj} \,|\, \mu_c, \Sigma_c) \end{aligned}$$

The EM algorithm is derived in Appendix B.2. The resulting E Step is

$$\widetilde{p}(a_{nj} = i) \;=\; \frac{\sum_{c=1}^{W} \delta(w_{ni} = c) \mathcal{N}(b_{nj} \,|\, \mu_c, \Sigma_c)}{\sum_{k=1}^{L_n} \sum_{d=1}^{W} \delta(w_{nk} = d) \mathcal{N}(b_{nj} \,|\, \mu_d, \Sigma_d)} \tag{3.5}$$

while the M Step updates for the cluster means and covariances are

$$\mu_c^{(new)} = \frac{\sum_{n=1}^{N} \sum_{j=1}^{M_n} \sum_{i=1}^{L_n} \delta(w_{ni} = c)\, \widetilde{p}(a_{nj} = i)\, b_{nj}}{\sum_{n=1}^{N} \sum_{j=1}^{M_n} \sum_{i=1}^{L_n} \delta(w_{ni} = c)\, \widetilde{p}(a_{nj} = i)} \tag{3.6}$$

$$\Sigma_c^{(new)} = \frac{\sum_{n=1}^{N} \sum_{j=1}^{M_n} \sum_{i=1}^{L_n} \delta(w_{ni} = c)\, \widetilde{p}(a_{nj} = i)\, (b_{nj} - \mu_c)^T (b_{nj} - \mu_c)}{\sum_{n=1}^{N} \sum_{j=1}^{M_n} \sum_{i=1}^{L_n} \delta(w_{ni} = c)\, \widetilde{p}(a_{nj} = i)} \tag{3.7}$$

One issue with this model is that there is no safeguard preventing singular cluster covariances Σ_c arising from an unbounded likelihood. In practice, we add the term σI_F to the $\Sigma_c^{(new)}$ update in order to spruce up the diagonal entries and avoid ill-conditioning. I_F is the $F \times F$ identity matrix and σ is a small number, typically on the order of 10^{-8}. Model *gMAP1*, presented in the next section, has a more appealing solution to the problem of ill-conditioning.

3.3 Model *gMAP1*

Model *gMAP1* assumes the same structure as model *gML1*. The difference is the additional equipment in the form of Bayesian priors that introduce stability to the parameters μ_c and Σ_c. In particular, the prior on the word cluster means has a feature weighting effect, as demonstrated in [Carbonetto *et al.*, 2003].

The idea behind feature weighting is that we would like to distinguish between dimensions that are useful for classification (i.e. dimensions in which there exists a strong correlation between the blob value and the associated concept) and dimensions that are not useful (i.e. dimensions that only contribute noise to the classification). One could argue that feature weighting or selection is best left up to the user since it should be fairly obvious which features are important. For example, the

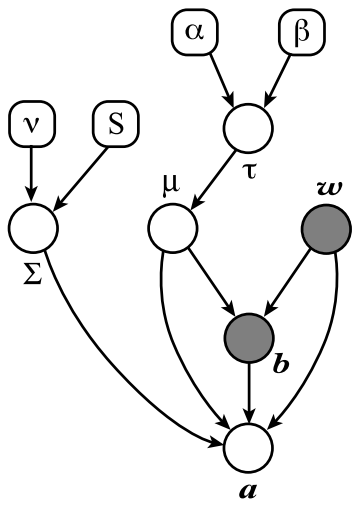

Figure 3.2: The directed graphical representation of model *gMAP1*. The observed variables are shaded and the fixed parameters are the square nodes.

blob's vertical position in a scene is often appropriate for distinguishing between concepts while the horizontal position is rarely useful. In contrast, we argue that a well-serviced automatic feature weighting procedure poses a distinct advantage over manual feature weighting for two reasons. First, the utility of features is not always readily discernible. Second, in our experience we have found that the relevance of features for classification is not universal, and often depends on the data set. For more discussion on this topic, see [Carbonetto *et al.*, 2003].

Irrelevant features exacerbate classification on the mixture components because the model may detect idiosyncrasies in the data. After pruning or reducing the weight of irrelevant features, the model predictions will be less prone to artifacts in the sample data. The overall effect is to stabilise the model predictions by rendering it less susceptible to noise. The Bayesian priors act as supervisor or quarterback for the dimensions of the cluster means; if a certain dimension is posing little overall utility for distinguishing between concepts, then the priors will push the individual clusters toward a global mean, thereby reducing the impact in classification for the particular dimension. The scheme presented in [Carbonetto *et al.*, 2003] using shrinkage priors is particularly appealing compared to some other methods such as principal component analysis because selection is integrated within the model computation, avoiding a separate decision step.

We point out a couple caveats regarding feature weighting. In our experience, feature weighting is a boon to unsupervised classification, however, we advise a degree of caution. It is unclear how well the feature weighting scales to data in high dimensions, especially when faced with a paucity of data. Conceivably, there may be situations in which feature weighting hinders accurate classification on certain concepts. Say, for instance, a feature f acts as noise on all concepts except word c, in which it is the *only* relevant feature. In such a scenario, feature weighting could conceivably hinder performance on the that concept. In general, optimal selection of shrinkage hyperparameters α and β is difficult.

We now specify the hierarchical Bayesian model, where the unknown parameters are regarded as being drawn from appropriate prior distributions. The complete hierarchical model is depicted as a directed graphical model in Figure 3.2. We assume the cluster covariances Σ_c are drawn from an inverse-Wishart distribution $\Sigma_c \sim \mathcal{IW}_F(\nu, \nu S)$. The effect of the inverse-Wishart is to prevent

Σ_c from becoming singular or going to zero, which may happen when there are few instances of the word token c. We discuss the choice of hyperparameters ν and S later in this section. We place a Gaussian prior on each cluster mean, $\mu_c \sim \mathcal{N}(\mu^*, T^2)$. T^2 is an $F \times F$ diagonal matrix with entries $\tau_1^2, ..., \tau_F^2$ and we take μ^* to be the sample mean of the blobs. Notice that a τ_f near 0 forces the corresponding component f of the means μ_c to shrink to a common sensible value. We assign an inverse-Gamma prior to each of the shrinkage parameters $\tau_f^2 \sim \mathcal{I}g(\alpha, \beta)$. All the shrinkage parameters τ_f^2 are assumed to be independent. We now summarise the priors, hyperparameters and independence assumptions of the model and present the complete joint posterior for the model parameters $\theta \triangleq \{\mu_1, ..., \mu_W, \Sigma_1, ..., \Sigma_W, \tau_1, ..., \tau_F\}$,

$$p(\theta, \mathbf{a} \,|\, \mathbf{b}, \mathbf{w}) = \prod_{c=1}^{W} \mathcal{IW}_F(\Sigma_c \,|\, \nu, \nu S)\, \mathcal{N}(\mu_c \,|\, \mu^*, T^2) \prod_{f=1}^{F} \mathcal{I}g(\tau_f^2 \,|\, \alpha, \beta)$$
$$\times \prod_{n=1}^{N} \prod_{j=1}^{M_n} \prod_{i=1}^{L_n} \prod_{c=1}^{W} \mathcal{N}(b_{nj} \,|\, \mu_c, \Sigma_c)^{\delta(w_{ni}=c)\, a_{nj}^i} \tag{3.8}$$

The alignment probabilities and word-to-blob translation probabilities are the defined the same as in equation (3.4).

The E Step remains unchanged from expression (3.5) derived for the *gML1* model. Observing the derivation of the M Step in Appendix B.3, the update equations are

$$\mu_c^{(new)} = T^2 A_c^{-1} \sum_{n=1}^{N} \sum_{j=1}^{M_n} \sum_{i=1}^{L_n} \delta(w_{ni} = c)\, \widetilde{p}(a_{nj} = i)\, b_{nj} + \Sigma_c A_c\, \mu^* \tag{3.9}$$

$$\Sigma_c^{(new)} = \frac{\sum_{n=1}^{N} \sum_{j=1}^{M_n} \sum_{i=1}^{L_n} \delta(w_{ni} = c)\, \widetilde{p}(a_{nj} = i)\, (b_{nj} - \mu_c)^T (b_{nj} - \mu_c) + \nu S}{\sum_{n=1}^{N} \sum_{j=1}^{M_n} \sum_{i=1}^{L_n} \delta(w_{ni} = c)\, \widetilde{p}(a_{nj} = i) + \nu + F + 1} \tag{3.10}$$

$$\tau_f^{2\,(new)} = \frac{\sum_{c=1}^{W} (\mu_{c,f} - \mu_f^*)^T (\mu_{c,f} - \mu_f^*) + 2\beta}{W + 2\alpha + 2} \tag{3.11}$$

where A_c is defined in (B.21).

From the M Step update equations (3.9, 3.10, 3.11) one can readily observe shrinkage in action. As τ_f^2 tends toward zero, the right-hand term in (3.9) dominates on dimension f and the fth component of the mean shrinks to the sample mean. Moreover, shrinkage causes the respective dimension of the covariance matrices to converge to a common value, as observed in (3.10). The role of the shrinkage hyperparameters as quarterback to the cluster means is apparent from equation (3.11). If the majority of the means to not stray far from the sample mean, chances are they are fitting on noise, resulting in a decrease in τ_f^2. The νS in the covariance update (3.10) acts as a stabilising term.

Examination of the update equations also lends guidance to the selection of sensible values for the fixed parameters. Since we use a gradient ascent algorithm for finding a point sample at a local maximum, placing vague priors on the parameter distributions allows for too much variance, while strong priors may bias results. The inverse-Gamma prior β establishes the mode while small values of α produce a flat distribution. We found that a moderately vague distribution ($\alpha \approx 0.75$) with a peak near zero ($\beta \approx 0.5$) worked well for object classification. Carbonetto *et al.* [2003] advocate setting the prior S on the inverse-Wishart distribution to the sample covariance. However, this has the undesirable consequence of widening the clusters to the range of the data. Instead, we suggest $S = \Sigma^*/W$ where Σ^* is the sample covariance and W is the number of clusters, classes or words. In practice, we found a weak prior on the covariances $\nu = F + 2$ behaves best.

3.4 Model *dML1MRF*

One of the limitations made by existing approaches for translating image regions to words is the assumption that blobs are statistically independent. This assumption is usually made to reduce the translation model to a mixture model and consequently simplify computation. Model *dML1MRF* relaxes this assumption and allows for interactions between blobs through a Markov random field. That is, the probability of an image blob being aligned to a particular word depends on the word assignments of its neighbouring patches. We still retain some structure due to the Markov assumption. One could introduce additional relations at different scales using a hierarchical representation, as in [Freeman *et al.*, 2000]. Otherwise, the *dML1MRF* model retains the dependence assumptions and the blob discretisation of model *dML1*.

Dependence between neighbouring objects introduces spatial context to the classification. Spatial context increases expressiveness; two words may be indistinguishable using low-level features such as colour (e.g. "sky" and "water") but neighbouring objects may help resolve the context (e.g. "airplane"). Context also alleviates some of the problems caused by a poor blob clustering. For example, colourful birds tend to be segmented into several blobs, which will inevitably get placed in separate bins due to the different colours. Model *dML1MRF* can learn these blobs often co-occur and increase the probability of classifying them as "bird" when they appear next to each other in a scene. Experiments in Section 4 confirm our intuition, that spatial context combined with a basic nonlinear decision boundary produces relatively accurate scene labellings.

We claim model *dML1MRF* is a stepping stone for solving scene segmentation. Segmentation algorithms, even the best, commonly *over-segment* — that is, they separate a single object into several parts — because low-level features are insufficient for forming accurate boundaries between objects. The object recognition data has semantic information in the form of captions, so it is reasonable to expect that additional high-level information could improve segmentations. Model *dML1MRF* does in fact ameliorate segmentations by fusing adjacent image blobs that possess high affinities. However, we have not yet attained a model that solves both scene segmentation and object recognition in parallel.

Spatial context increases expressiveness, but this comes at an increased computational cost due to cycles introduced in the undirected graph. Since the model is intractable and the potentials over the cliques are not complete[1], parameter estimation in the M Step is difficult. Iterative scaling works on arbitrary exponential models, but it is not a saving grace because convergence on large models is exponentially-slow. We approximate the likelihood using the pseudo-likelihood (2.6). The pseudo-likelihood loses sight of some long-range interactions, but it allows us to get approximate M Step updates in a reasonable amount of time.

Let t be a $B \times W$ table of translation potentials $t(b^\star, w^\star)$ and let ψ be a $W \times W$ table of potentials describing the "next to" relation between blobs. We define the spatial context features to be symmetric, so $\psi(w^\star, w^\diamond) = \psi(w^\diamond, w^\star)$. The set of model parameters is $\theta \triangleq \{t, \psi\}$. We define the set of cliques in each document MRF by $\mathcal{C}_n$. The complete likelihood of model *dML1MRF* is

$$\begin{aligned} p(\theta, a \,|\, b, w) &= \prod_{n=1}^{N} \prod_{u=1}^{M_n} p(a_{nu} \,|\, a_{n1:u-1}, b_{n1:u}, w_{n1:L_n}, \theta)\, p(b_{nu} \,|\, a_{nu}, \theta) \\ &= \prod_{n=1}^{N} \frac{1}{Z_n(\theta)} \prod_{u=1}^{M_n} \Phi(b_{nu}, a_{nu}) \prod_{(u,v)\,\in\, C_n} \Psi(a_{nu}, a_{nv}) \end{aligned} \tag{3.12}$$

[1] A parameterisation is **complete** if we can attain a "perfect" model; i.e. the log likelihood is zero [Berger, 1997].

where we define the potentials on the translation cliques and the spatial context cliques to be

$$\Phi(b_{nu}, a_{nu}) = \prod_{i=1}^{L_n} t(b_{nu}, w_{ni})^{a^i_{nu}}$$
$$\Psi(a_{nu}, a_{nv}) = \prod_{i=1}^{L_n}\prod_{j=1}^{L_n} \psi(w_{ni}, w_{nj})^{a^i_{nu}\times a^j_{nv}}$$

$Z_n(\theta)$ is the partition function for the disjoint graph of document n. The pseudo-likelihood approximation of (3.12) is

$$\begin{aligned} p\ell(\theta, a \,|\, b, w) &= \prod_{n=1}^{N}\prod_{u=1}^{M_n} p(b_{nu}, a_{nu} \,|\, a_{\mathcal{N}_{nu}}) \\ &= \prod_{n=1}^{N}\prod_{u=1}^{M_n} \frac{1}{Z_{nu}(\theta)}\, p(b_{nu}, a_{nu}) \prod_{v\in\mathcal{N}_{nu}} p(a_{nu} \,|\, a_{nv}) \\ &= \prod_{n=1}^{N}\prod_{u=1}^{M_n} \frac{1}{Z_{nu}(\theta)} \Phi(b_{nu}, a_{nu}) \prod_{v\in\mathcal{N}_{nu}} \Psi(a_{nu}, a_{nv}) \end{aligned} \tag{3.13}$$

where $\mathcal{N}_{nu}$ is the set of blobs adjacent to node u in its local neighbourhood and $Z_{nu}(\theta)$ is the partition function for the neighbourhood at site u in document n.

Next, we describe the approximate EM algorithm for model *dML1MRF*. We use loopy belief propagation [Murphy *et al.*, 1999] on the complete likelihood (3.12) to compute the marginals $\widetilde{p}(a_{nu} = i)$ and $\widetilde{p}(a_{nu} = i, a_{nv} = j)$ in the E Step. The M Step consists of iterative scaling updates on the translation and spatial context parameters and, since IS is still quite slow to converge, we boost it with an additional iterative proportional fitting update on t.

To derive the iterative scaling equations we bound the pseudo-likelihood (3.13). Each iterative scaling update is the unique root of the partial derivative of t or ψ with respect to the lower bound on the pseudo-likelihood, $\Lambda(\theta)$. The partial derivatives, motivated by (2.16), are

$$\begin{aligned} \frac{\partial\Lambda}{\partial t(b^\star, w^\star)} &= \sum_{n=1}^{N}\sum_{u=1}^{M_n}\sum_{i=1}^{L_n} \delta(b_{nj} = b^\star)\,\delta(w_{ni} = w^\star)\,\widetilde{p}(a_{nu} = i) \\ &\quad + \sum_{n=1}^{N}\sum_{u=1}^{M_n} \triangle t(b^\star, w^\star)^{|\mathcal{N}_{nu}|+1} \sum_{i=1}^{L_n} \delta(w_{ni} = w^\star)\, p(b_{nu} = b^\star, a_{nu} = i \,|\, \theta) \end{aligned} \tag{3.14}$$

$$\begin{aligned} \frac{\partial\Lambda}{\partial\psi(w^\star,w^\diamond)} &= \sum_{n=1}^{N}\sum_{u=1}^{M_n}\sum_{v\in\mathcal{N}_{nu}}\sum_{i=1}^{L_n}\sum_{j=1}^{L_n} \delta(w_{ni} = w^\star)\,\delta(w_{nj} = w^\diamond)\,\widetilde{p}(a_{nu} = i, a_{nv} = j) \\ &\quad + \sum_{n=1}^{N}\sum_{u=1}^{M_n}\sum_{v\in\mathcal{N}_{nu}} \triangle\psi(w^\star,w^\diamond)^{|\mathcal{N}_{nu}|+1} \sum_{i=1}^{L_n}\sum_{j=1}^{L_n} \delta(w_{ni}=w^\star)\delta(w_{nj}=w^\diamond) p(a_{nu}=i \,|\, \widetilde{a}_{nv}=j, \theta) \end{aligned} \tag{3.15}$$

where we take $p(a_{nu}=i \,|\, \widetilde{a}_{nv}=j, \theta)$ to be the estimated conditional of the alignment a^i_{nu} conditional on the empirical distribution $\widetilde{p}(a_{nv})$ and the current parameters. To find the conditionals for the iterative scaling update (3.15), we run universal propagation and scaling at each pseudo-likelihood site nu with the neighbours $v \in \mathcal{N}_{nu}$ clamped to the current marginals $\widetilde{p}(a_{nv})$. Note that UPS is

exact because the undirected graph at each neighbourhood is a tree. Also note that the IS update (3.14) requires estimation of the blob densities in addition to the alignment marginals. Equations (3.14) and (3.15) are polynomial formulas where each degree $|\mathcal{N}_{nu}| + 1$ is the clique count of the blob neighbourhood u in image n. The final matter is to find the unique root for $\triangle t(b^\star, w^\star)$ and plug it into (3.16) and correspondingly find unique solution $\triangle\psi(w^\star, w^\diamond)$ for update equation (3.17).

$$t^{(new)}(b^\star, w^\star) = t(b^\star, w^\star) \times \triangle t(b^\star, w^\star) \tag{3.16}$$

$$\psi^{(new)}(w^\star, w^\diamond) = \psi(w^\star, w^\diamond) \times \triangle\psi(w^\star, w^\diamond) \tag{3.17}$$

EM for model *dML1MRF* is summarised by the algorithm below.

Expectation Maximisation algorithm for model *dML1MRF*.

1: Initialise $t^{(0)}$ and $\psi^{(0)}$ to arbitrary values.
2: $s \leftarrow 0$
3: **while** s is less than the maximum number of iterations and EM convergence criterion is not met **do**
4: $\quad s \leftarrow s + 1$
5: $\quad$ Run loopy BP to find the marginals $\widetilde{p}(a_{nu} = i)$ and $\widetilde{p}(a_{nu} = i, a_{nv} = j)$.
6: $\quad$ Find marginals $p(b_{nu} = b^\star, a_{nu} = i)$ and $p(a_{nu} = i \,|\, \widetilde{a}_{nv} = j)$ using UPS, given current parameter estimates $t^{(s-1)}$ and $\psi^{(s-1)}$ and empirical marginals $\widetilde{p}(a_{nu} = i)$.
7: $\quad$ Compute $t^{(s)}$ using IS update (3.16).
8: $\quad$ Compute $\psi^{(s)}$ with IS update (3.17).

On large data sets we found the iterative scaling updates were too slow. Optionally, one can boost the M Step with an additional IPF step. We are permitted to perform an IPF update on t because, as justified in Section 2.5.1, there is only one clique associated with the translation parameter in each neighbourhood. The IPF equation for t is

$$t^{(new)}(b^\star, w^\star) = t(b^\star, w^\star) \times \frac{\sum_{n=1}^{N} \sum_{u=1}^{M_n} \sum_{i=1}^{L_n} \delta(w_{ni} = w^\star)\, \delta(b_{nu} = b^\star)\, \widetilde{p}(a_{nu} = i)}{\sum_{n=1}^{N} \sum_{u=1}^{M_n} \sum_{i=1}^{L_n} \delta(w_{ni} = w^\star)\, p(b_{nj} = b^\star, a_{nu} = i \,|\, \theta)} \tag{3.18}$$

To stabilise the parameter updates, we place vague priors on t and ψ. Typically, we set the prior on t to 10^{-5} and the ψ prior to 10^{-4}. The latter prior is strong enough to apply a small amount of uniformness and slightly decrease the influence of spatial context in labeling decisions. We found this reasonably-vague prior on ψ works well, although we caution that prior selection in Markov random fields is notoriously difficult [Besag, 1986].

Chapter 4

Experiments

The experiments compare the four models proposed in Section 3. We show annotation results on a variety of images and evaluation of object recognition performance using two different methods of segmentation.

4.1 Evaluation metric considerations

Before we discuss what makes a good evaluation metric, it will help if we answer this question: what makes a good image annotation? As we will see, there is no straightforward answer.

It is fair to say that certain concepts in an image are more prominent than others. One might take the approach that the objects that consume the most space in an image are the most important, and this is roughly the evaluation criterion used in previous work [Duygulu *et al.*, 2002; Carbonetto *et al.*, 2003]. Consider the image on the left in Figure 4.1. We claim that "polar bear" is at least as important as snow. There is an easy way to test this assertion — pretend the image is annotated either entirely as "snow" or entirely as "polar bear". In our experience, people find the latter annotation as appealing, if not more, than the former. Therefore, one would conclude that it is better to weight all concepts equally, regardless of size, which brings us to the image on the right. If we treat all words equally, having many words in a single label obfuscates the goal of getting the most important concept, "train", correct.

The relative importance of objects in a scene is task-dependent. For instance, in robotics the identification of obstacles such as tables, chairs and people is more important than predicting the

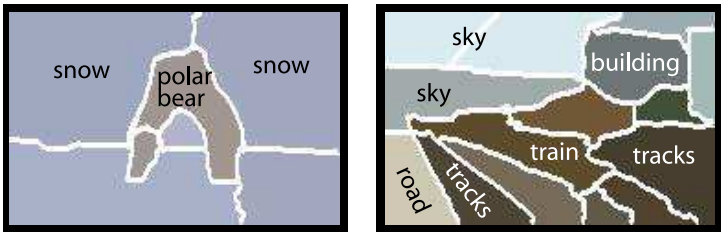

Figure 4.1: Examples of images which demonstrate that the importance of concepts has little or no relation to the area these concepts occupy. On the left, "polar bear" is at least pertinent as "snow" even though it takes up less area in the image. In the photograph on the right, "train" is most likely the focus of attention.

occurence of sky or posters on the wall. Ideally, when collecting user-annotated images for the purpose of evaluation, we should tag each word with a weight to specify its prominence in the scene. In practice, this is problematic because different users focus their attention on different concepts, not to mention the fact that it is a burdensome task.

In practice, we found that rewarding prediction accuracy over blobs — not objects — in a scene results in a reasonable performance measure, although we admit we have not explored the more robust evaluation options proposed by Borra and Sarkar [1997] and Barnard *et al.* [2003a]. The evaluation measure reports an error of 1 if the model annotation with the highest probability results in an incorrect patch annotation. The error is averaged over the number of patches in each image, and then again over the number of images in the data set. The prediction error is given by

$$\frac{1}{N}\sum_{n=1}^{N}\frac{1}{M_n}\sum_{j=1}^{M_n}\left(1-\delta\left(\tilde{a}_{nj}=a_{nj}^{(max)}\right)\right) \tag{4.1}$$

where $a_{nj}^{(max)}$ is the model alignment with the highest probability and $\tilde{a}_{nj}$ is the ground-truth annotation.

Ideally, we would like to match the model likelihood expression to any proposed evaluation metric, but the problem is that it would often involve knowing the assignments of the blobs, which of course are not provided in the training data.

4.2 Experiment setup

We train and compare the four proposed translation models on labeled images from the Corel image database and from our laboratory robot. We also examine the effect of two different segmentations on the performance of our models.

We composed three separate sets, denoted by *CorelB*, *CorelC* and *Robomedia*[1]. The first data set, *CorelB*, has 199 training images and 100 test images with 38 words in the training set. The *CorelB* data set contains a total of 504 annotated images, divided into training and test sets numbering 336 and 168 in size. The training set has a total of 55 distinct concepts. We built the *Robomedia* data set by having our autonomous robot José [Elinas *et al.*, 2002] roam around the lab taking pictures, and then having laboratory members create captions for the data using a consistent set of words. For evaluation purposes, we annotated the images by hand. The *Robomedia* data set is composed of 107 training images, 43 test images and 21 word tokens. The word frequencies in the labels and manual annotations for the *CorelB* and *Robomedia* data sets with are shown in Figures 4.6, 4.7 and 4.8. (As we explain later on in this section, the manual annotation frequencies differ slightly depending on the segmentation method.) For the *Robomedia* data, we show only the word frequencies for the *Grid* segmentation.

The Corel photographs tend to be very well composed with a specific subject and proper lighting conditions. These are certainly not realistic conditions, but we do not pretend that our object recognition models are robust to challenging situations. In contrast, the *Robomedia* data set suffers from poor lighting conditions, overexposure and poorly-composed images with ill-defined objects. Additionally, standard low-level features such as colour and texture are particularly ineffective in a laboratory environment. For example, chairs can come in a variety of shapes and colours, and "wall" refers to a vertical surface that has virtually no relation to colour, texture and position. Moreover, it is much more difficult to delineate specific concepts in a scene, even for humans — does a

[1]The experiment data is available at `http://www.cs.ubc.ca/~pcarbo`.

table include the legs, and where does one draw the line between shelves, drawers, cabinets and the objects contained in them? As a result, many of the image region annotations are inconsistent or left blank because they contain no obvious concept. While the *Robomedia* data set contains a small sample of images, the correspondence task is difficult due to the high level of noise in the data and the large number of objects in each scene. Object recognition on the *Corel* data sets is comparatively easy because the photos are captured to artificially delineate specific concepts. Colour and texture tend to be more informative in natural scenes as well.

The image regions are described using simple colour features. We also use vertical position for all the models except the the discrete translation models on the Corel data sets since the k-means clustering tends to be very poor. The number of clusters for the discrete translation models *dML1* and *dML1MRF* is set to approximately 5 times the number of words in the training set, and more specifically 200 for the *CorelB* set, 250 for *CorelC* and 100 for the *Robomedia* data set. The values of the priors for the hierarchical mixture model *gMAP1* are $\alpha = 0.8$, $\beta = 0.5$ and $\nu = F + 3$ where F is the number of features.

In our experiments, we consider two scenarios. In the first, we use Normalized Cuts [Shi and Malik, 1997] to segment the images into distinct patches. In the second scenario, we take on the object recognition task without the aid of a sophisticated segmentation algorithm, and instead construct a uniform grid of patches over the image. Examples of different segmentations are shown in in the anecdotal results in Figures 4.12, 4.13, 4.14 and 4.15. For the crude segmentation, the choice of grid size is important since our methods are not scale invariant. We used patches of height and width approximately $1/6$th the size of the image. We found that smaller patches introduced too much noise to the features and resulted in poor test performance, and larger patches contained too many objects at once. The two scenarios are denoted by *NCuts* and *Grid*.

One caveat regarding our evaluation procedure: the segmentation methods are not directly comparable because the manual annotation data is different for the Normalized Cuts uts and grid segmentations. To provide a ground truth, we annotated the segments in the images by hand. In some cases, there might be more than one annotation deemed correct because the region consists of several subjects. When a segment encompasses more than one concept, we give the model the benefit of the doubt. For example, according to our evaluation the annotations for both the grid and Normalized Cuts segmentations shown in Figure 4.2 are correct. However, from observation the grid segmentation provides a more precise object identification in the scene. As a result, evaluation can be unreliable when Normalized Cuts offers inferior segmentations. To address the role of segmentation in object recognition more accurately, we would like to build data sets with ground-truth annotations and segmentations. Fortunately, half the work has already been done [Martin *et al.*, 2001].

4.3 Results

A comparison of the model annotation error on the data sets *CorelB*, *CorelC* and *Robomedia* using the two segmentation methods is given in Figures 4.3, 4.4 and 4.5. The plots depict the mean and variance of the error over 12 runs of the EM algorithm corresponding to local minima of the log posterior (except in the case of the *dML1MRF* model). The red line at the top of each plot is the random bound on the translation error. The models that assume independence of objects in a scene took on the order of a few minutes to converge to a local point sample. Model *dML1MRF* usually took several hours to learn the potentials.

The first striking observation is that the contextual model shows consistently good results on the training sets but its performance fails to generalise to the test sets. The models that incorporate

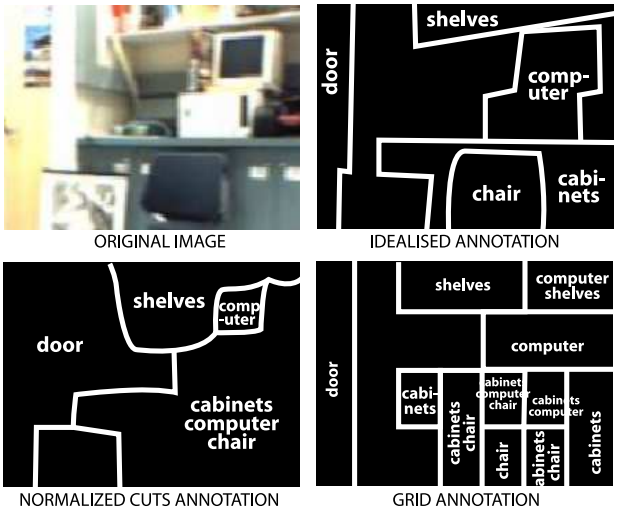

Figure 4.2: Correct annotations for Normalized Cuts, grid and manual segmentations. When there are multiple annotations in a single patch, any one of them is correct. Even when both are correct, the grid segmentation is usually more precise and, as a result, more closely approximates generic object recognition.

clustering into the learning step tend to generalise better than the models that require an initial clustering step to provide the blob discretisation. This suggests that a combination of *dML1MRF* and *gMAP1* — a model that learns both spatial relations and blob clusterings — would outperform the best of our current model annotations.

A surprising result is that the variance of model *dML1MRF* is not significantly greater than the other models, despite an increase in the number of parameters and lack of convergence guarantees on the learning algorithm. While this is surely an encouraging result, we cannot claim our approximate learning method is robust. *dML1MRF* fails drastically on the test sets and the *Robomedia* data using *NCuts*. Often it does not converge at all with the Normalized Cuts segmentations. When the approximate EM algorithm fails, it is because it cannot not find a unique positive root for the iterative scaling update (3.16) or (3.17). We can only claim that *dML1MRF* provides consistent quality annotations on the training sets when it converges to a local solution.

Interestingly, recognition using the grid segmentation tends to do better than the Normalized Cuts results. This suggests that we can achieve comparable results without an expensive segmentation step. It is also reasonable to expect that the *Grid* blob annotation is more than just a small improvement given that we require the predictions to be precise over smaller image regions, as discussed in the previous section. However, we are cautious not to make strong claims about the utility of sophisticated low-level segmentations in object recognition because we do not have a uniform evaluation framework and we have not examined different segmentation methods in sufficient variety and detail.

We show the precision on individual words for experiments *CorelB* and *Robomedia* in Figures 4.6, 4.7 and 4.8. The precision is averaged over the 12 trials. While not shown in the figures, we have noticed considerable variation among individual trials as to what words are predicted with high precision. For example, model *dML1MRF* with a grid segmentation predicts the word “train” with average success 0.396, although the precision on individual trials ranges from 0.102 to 0.862 in the

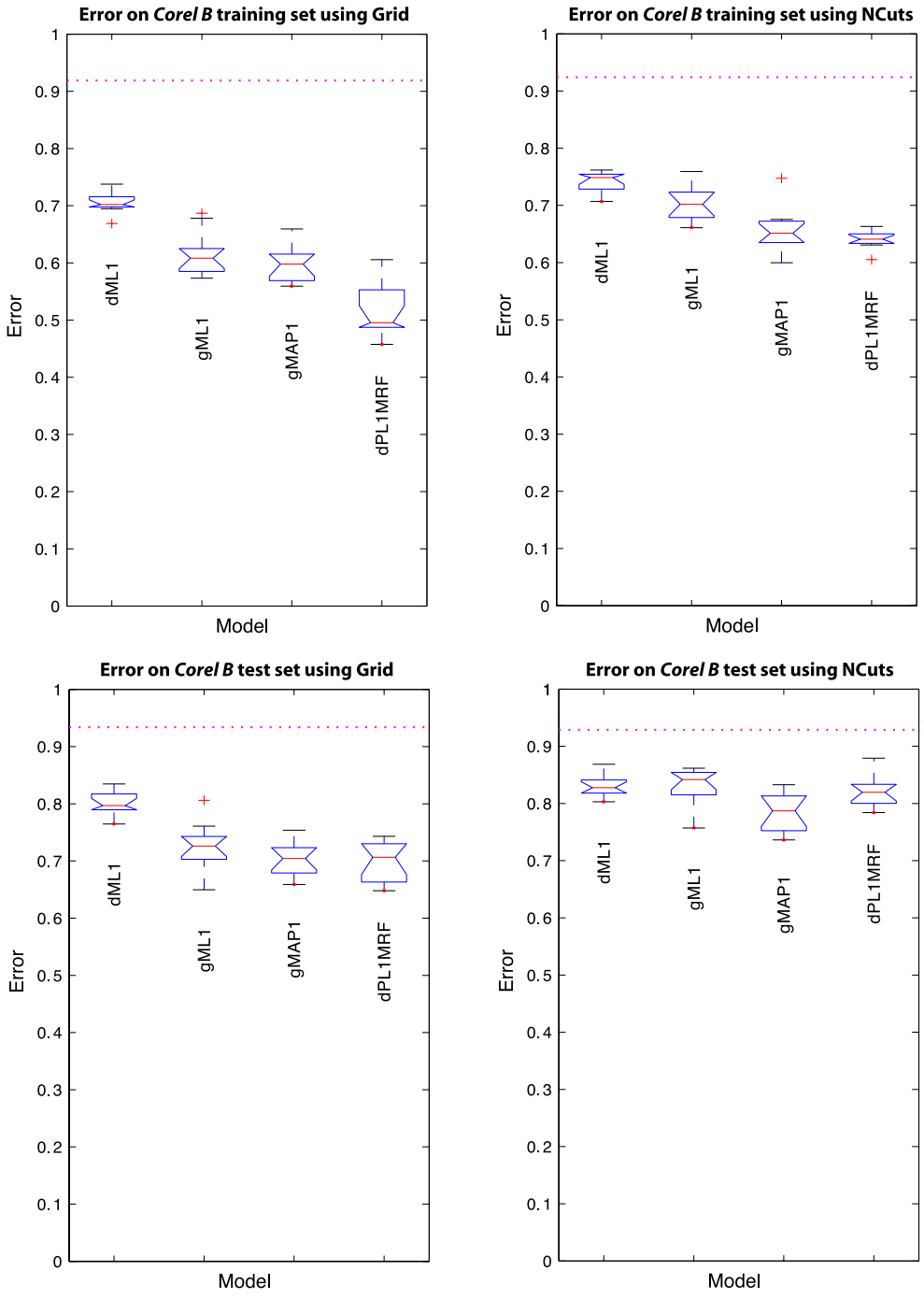

Figure 4.3: Prediction error of the four models on the *CorelB* training and test sets, displayed using a Box-and-Whisker plot. The middle line of a box represents the median. The central box represents the values from the 25 to 75 percentile, using the upper and lower statistical medians. The horizontal line extends from the minimum to the maximum value, excluding outside and far out values which are displayed as separate points. The dotted line at the top is the random prediction upper bound. On average, model *dML1MRF* produces the best blob annotations in the training sets but *gMAP1* generalises better to test sets.

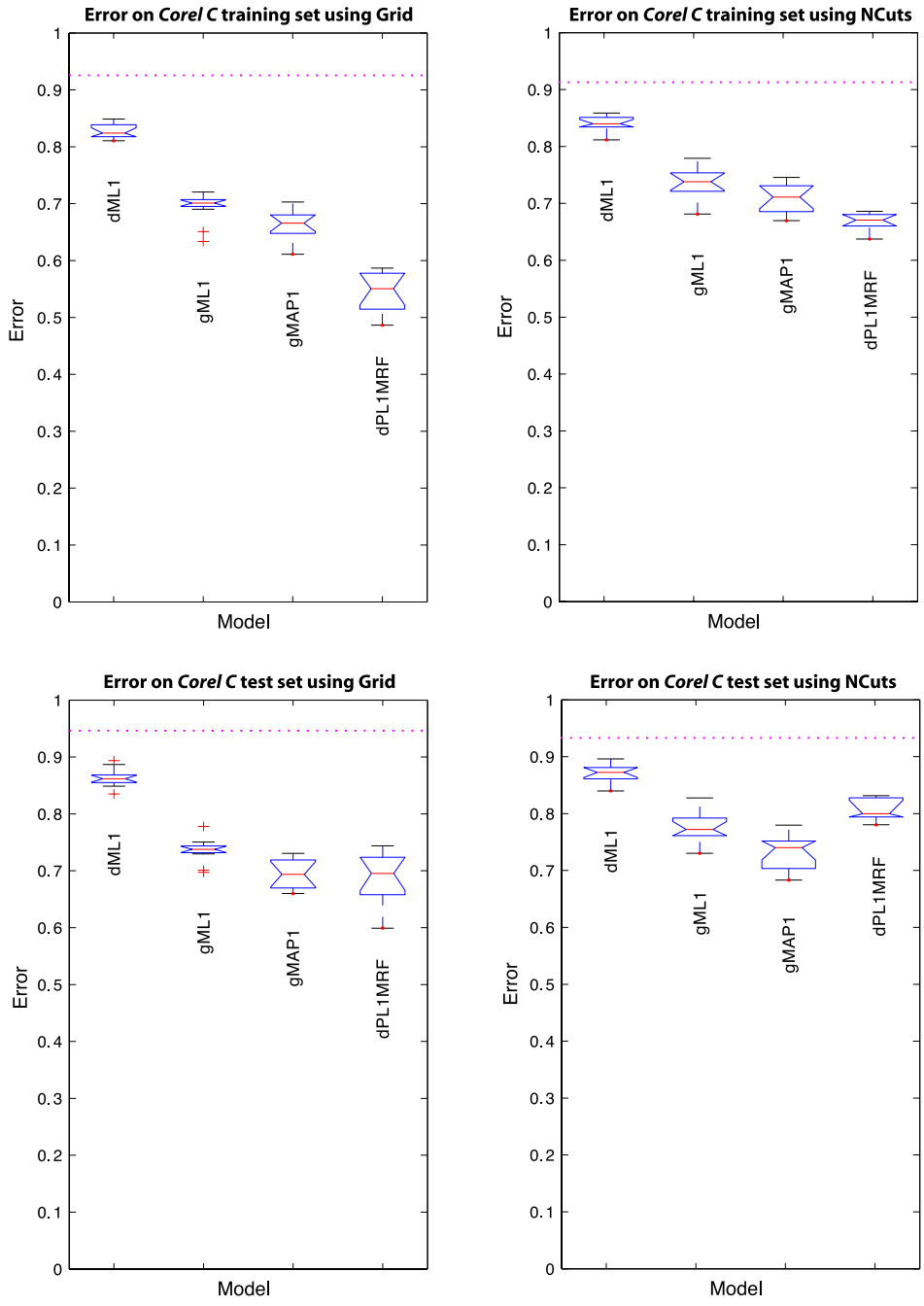

Figure 4.4: Prediction error of the four models on the *CorelC* training and test sets, displayed using a Box-and-Whisker plot. See the caption from Figure 4.3 for a full explanation of the plot. As for the *CorelB* set, model *dML1MRF* performs best on the training set but fails to generalise to the test set, unlike *gMAP1*.

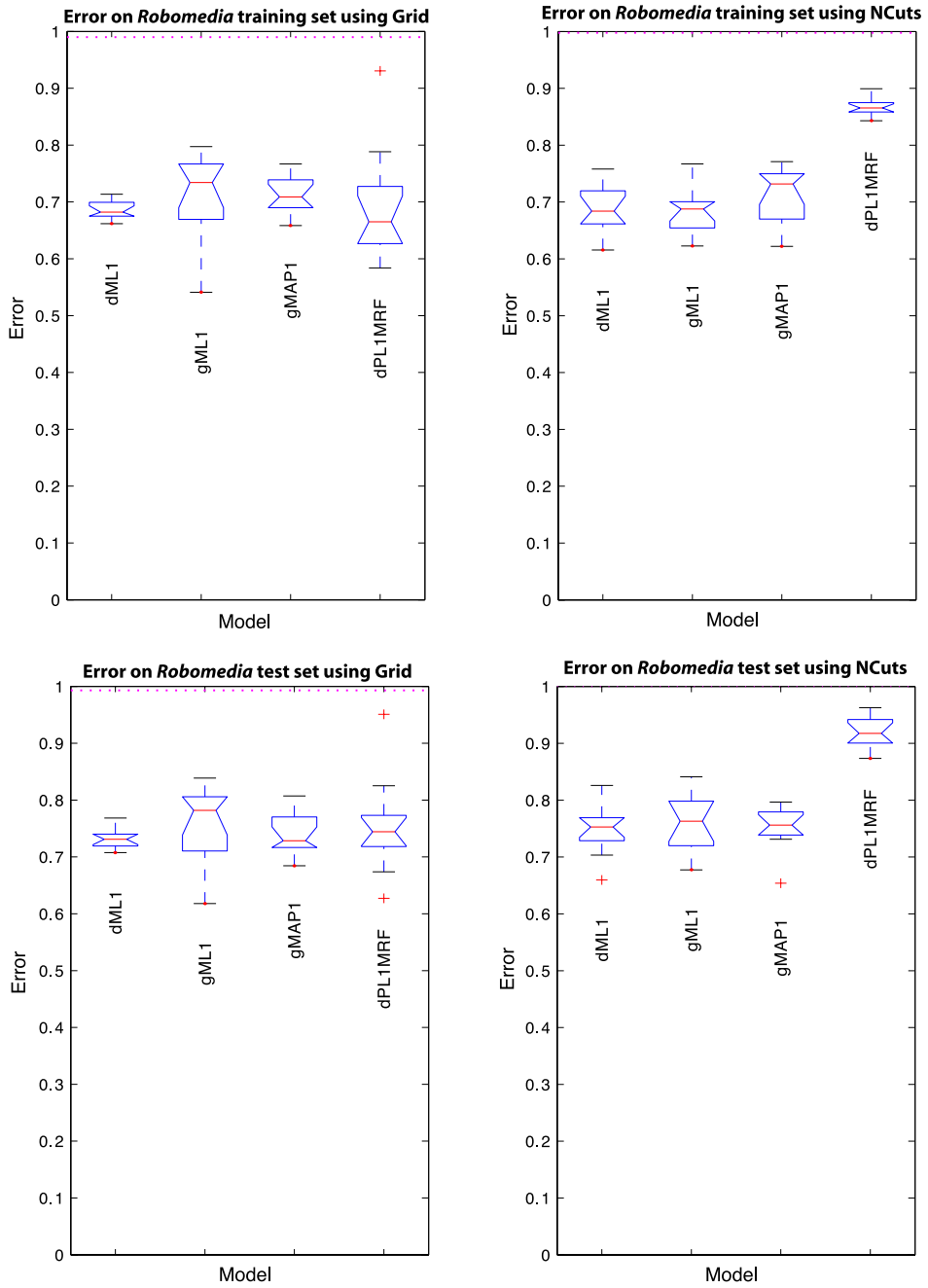

Figure 4.5: Prediction error of the four models on the *Robomedia* training and test sets, displayed using a Box-and-Whisker plot. See the caption from Figure 4.3 for a full explanation of the plot. There is no consistent winner on this data set.

training set. Significantly, the spatial context model tends to do better on words that cannot be described using simple colour features, such as “wolf”, “whale” and “building” but worse on relatively common and separable concepts such as “grass” and “water”. The models derive unexpectedly good performance on more abstract concepts that have little or no relation to low-level features, such as “floor”, “ceiling” and “wall” in the *Robomedia* data set. We suspect that the composition of the labeled data permits the models to find correspondences on poorly separable words; for instance, if the model is confident about all the other objects in the scene, then whatever is left over must be “floor” or “ceiling”.

Figure 4.9 shows the variable weighting parameter τ averaged over the trials on selected data sets. y is the horizontal position of the blob. The shrinkage model consistently places high importance on the means of the colours in the *CorelB* trials, which confirms our intuition. In the *Robomedia* images the colours have less weight, which makes sense because the objects in laboratory scenes observe significantly less variation in colour.

Figures 4.10 and 4.11 depict the potentials ψ for the *CorelB Grid* and *Robomedia Grid* experiments, respectively. Note that the tables are symmetric. The diagonal is dark because words appear most often next to themselves. The strong diagonal acts as a smoothing agent on the segmented scenes. Most of the high affinities are logical (e.g. “shuttle” and “astronaut”) but there are a few that defy common sense (e.g. “trees” and “trunk” have a weak affinity), probably due to incorrect associations between the blobs and words.

Selected annotations predicted by the *gMAP1* and *dML1MRF* models on the training sets are displayed in Figures 4.12 and 4.14. Test set annotations are shown in Figures 4.13 and 4.15. It is worth re-emphasising that these predictions are made without knowledge of the image labels. Also, it is important to remember that the predictions are probabilitistic, so the figures do not give an idea of the confidence the models have in their predictions. The trials using Normalized Cuts segmentations produce more visually appealing annotations compared to the rectangular grid, but this is misleading because the error measures consistently demonstrate good results with rectangular segments.

Predicition *CorelB* data set using *NCuts*

	LABEL %		ANNOTATION %‡		*gMAP1* PR.		*dML1MRF* PR.	
WORD	**TRAIN**	**TEST†**	**TRAIN**	**TEST**	**TRAIN**	**TEST**	**TRAIN**	**TEST**
airplane	0.060	0.054	0.067	0.066	0.186	0.131	0.263	0.198
astronaut	0.003	0.004	0.001	0.002	0.083	0.000	0.583	0.000
atm	n/a	0.004	n/a	0.005	n/a	0.000	n/a	0.000
bear	0.031	0.018	0.018	0.012	0.345	0.219	0.402	0.129
beluga	n/a	0.004	n/a	0.004	n/a	0.000	n/a	0.000
bill	0.019	0.018	0.040	0.024	0.317	0.359	0.254	0.029
bird	0.017	0.011	0.027	0.015	0.341	0.046	0.631	0.194
building	0.014	0.007	0.008	0.004	0.052	0.000	0.474	0.067
cheetah	0.012	0.018	0.014	0.020	0.571	0.064	0.928	0.139
cloud	0.048	0.039	0.043	0.032	0.300	0.236	0.283	0.192
coin	0.005	0.007	0.009	0.003	0.389	0.000	0.781	0.000
coral	0.005	n/a	0.008	n/a	0.392	n/a	0.662	n/a
crab	0.002	0.004	0.003	0.005	0.536	0.000	1.000	0.396
dolphin	0.014	0.004	0.008	0.003	0.298	0.458	0.714	0.306
earth	0.003	0.007	0.003	0.004	0.312	0.000	0.604	0.042
fish	0.007	n/a	0.003	n/a	0.479	n/a	0.546	n/a
flag	0.005	0.007	0.004	0.014	0.346	0.689	0.701	0.868
flowers	n/a	0.004	n/a	0.001	n/a	0.000	n/a	0.000
fox	0.010	0.018	0.014	0.017	0.524	0.104	0.644	0.042
goat	0.002	n/a	0.001	n/a	0.083	n/a	1.000	n/a
grass	0.128	0.118	0.145	0.133	0.398	0.361	0.149	0.135
hand	n/a	0.004	n/a	0.001	n/a	0.000	n/a	0.000
map	n/a	0.004	n/a	0.003	n/a	0.000	n/a	0.000
mountain	0.012	0.004	0.010	0.002	0.149	0.000	0.519	0.000
person	0.003	0.011	0.002	0.008	0.083	0.000	1.000	0.000
polarbear	0.021	0.025	0.015	0.013	0.222	0.054	0.715	0.321
rabbit	0.002	n/a	0.001	n/a	0.125	n/a	1.000	n/a
road	0.026	0.022	0.016	0.012	0.166	0.125	0.409	0.156
rock	0.024	0.036	0.022	0.039	0.130	0.042	0.327	0.041
sand	0.034	0.025	0.027	0.025	0.251	0.169	0.291	0.191
shuttle	0.007	0.007	0.009	0.008	0.165	0.106	0.172	0.000
sky	0.155	0.140	0.115	0.124	0.258	0.229	0.122	0.158
snow	0.038	0.065	0.067	0.108	0.325	0.302	0.297	0.288
space	0.007	0.011	0.003	0.008	0.146	0.028	0.187	0.097
tiger	0.021	0.036	0.024	0.043	0.666	0.306	0.765	0.294
tracks	0.024	0.018	0.011	0.011	0.133	0.118	0.421	0.188
train	0.026	0.022	0.030	0.017	0.244	0.206	0.288	0.079
trees	0.091	0.072	0.085	0.057	0.282	0.189	0.200	0.123
trunk	0.003	0.011	0.002	0.007	0.158	0.000	0.861	0.000
water	0.091	0.090	0.121	0.089	0.334	0.274	0.149	0.133
whale	0.007	0.007	0.006	0.009	0.104	0.014	0.622	0.000
wolf	0.009	0.039	0.007	0.043	0.083	0.020	0.524	0.044
zebra	0.012	0.011	0.011	0.006	0.676	0.206	0.881	0.386
Totals	**1.000**	**1.000**	**1.000**	**1.000**	**0.315**	**0.205**	**0.348**	**0.162**

Figure 4.6: The first four columns list the probability of finding a particular word in a label and manually-annotated image region in the *CorelB* training and test sets using the Normalized Cuts segmentation. The final four columns show the precision of translation models *gMAP1* and *dML1MRF* averaged over the 12 trials. Precision is defined as the probability the model's prediction is correct for a particular word and blob. Since precision is 1 minus the error of equation (4.1), the total precision on both the training and test sets matches the average performance shown in Figure 4.3. The variance in the precision on individual words is not presented in this table. Note that some words do not appear in both the training and test sets, hence the "n/a".
†The model predicts words *without* access to the test image labels, provided for only completeness.
‡We can use the manual annotations for evaluation purposes, but we underline the fact that an agent would not have access to the information presented in the ANNOTATION % column.

Predicition *CorelB* data set using *Grid*

	LABEL %		ANNOTATION %‡		*gMAP1* PR.		*dML1MRF* PR.	
WORD	**TRAIN**	**TEST†**	**TRAIN**	**TEST**	**TRAIN**	**TEST**	**TRAIN**	**TEST**
airplane	0.060	0.055	0.036	0.028	0.214	0.211	0.290	0.187
astronaut	0.003	0.003	0.001	0.002	0.225	0.019	0.000	0.135
atm	n/a	0.003	n/a	0.006	n/a	0.000	n/a	0.000
bear	0.031	0.017	0.021	0.013	0.303	0.205	0.452	0.272
beluga	n/a	0.003	n/a	0.005	n/a	0.000	n/a	0.000
bill	0.019	0.017	0.046	0.031	0.396	0.394	0.335	0.146
bird	0.017	0.010	0.009	0.004	0.139	0.000	0.556	0.458
building	0.014	0.007	0.006	0.002	0.291	0.100	0.408	0.137
cheetah	0.012	0.017	0.010	0.013	0.465	0.332	0.710	0.395
cloud	0.050	0.045	0.050	0.048	0.305	0.261	0.300	0.239
coin	0.005	0.007	0.008	0.008	0.616	0.136	0.767	0.017
coral	0.005	n/a	0.011	n/a	0.456	n/a	0.738	n/a
crab	0.002	0.003	0.001	0.002	0.703	0.533	1.000	0.833
dolphin	0.014	0.003	0.006	0.001	0.596	0.833	0.916	NaN
earth	0.003	0.007	0.004	0.002	0.247	0.000	0.732	0.142
fish	0.007	n/a	0.003	n/a	0.560	n/a	0.695	n/a
flag	0.005	0.007	0.004	0.008	0.495	0.786	0.888	0.890
flowers	n/a	0.003	n/a	0.003	n/a	0.000	n/a	0.000
fox	0.010	0.017	0.011	0.010	0.315	0.048	0.691	0.008
goat	0.003	n/a	0.001	n/a	0.415	n/a	0.994	n/a
grass	0.129	0.120	0.177	0.157	0.365	0.380	0.172	0.229
hand	n/a	0.003	n/a	0.002	n/a	0.000	n/a	0.000
map	n/a	0.003	n/a	0.003	n/a	0.000	n/a	0.000
mountain	0.012	0.003	0.007	0.000	0.180	0.000	0.671	0.057
person	0.003	0.014	0.001	0.004	0.267	0.000	1.000	0.000
polarbear	0.021	0.024	0.016	0.015	0.318	0.181	0.681	0.634
rabbit	0.002	n/a	0.001	n/a	0.142	n/a	1.000	n/a
road	0.026	0.024	0.016	0.008	0.351	0.149	0.526	0.213
rock	0.019	0.038	0.018	0.033	0.254	0.095	0.446	0.130
sand	0.034	0.024	0.023	0.024	0.288	0.260	0.330	0.185
shuttle	0.007	0.007	0.006	0.005	0.504	0.283	0.305	0.107
sky	0.156	0.137	0.172	0.173	0.226	0.200	0.190	0.208
snow	0.036	0.062	0.064	0.110	0.473	0.413	0.435	0.356
space	0.007	0.010	0.008	0.018	0.277	0.042	0.326	0.071
tiger	0.021	0.034	0.015	0.030	0.604	0.323	0.623	0.285
tracks	0.024	0.017	0.010	0.009	0.162	0.130	0.575	0.315
train	0.026	0.021	0.021	0.014	0.299	0.197	0.396	0.272
trees	0.095	0.076	0.075	0.069	0.389	0.234	0.227	0.134
trunk	0.003	0.010	0.001	0.006	0.068	0.000	0.910	0.000
water	0.091	0.089	0.120	0.095	0.320	0.193	0.212	0.137
whale	0.007	0.007	0.003	0.004	0.564	0.120	0.854	0.405
wolf	0.009	0.038	0.007	0.029	0.340	0.104	0.660	0.102
zebra	0.012	0.010	0.009	0.007	0.629	0.492	0.903	0.710
Totals	**1.000**	**1.000**	**1.000**	**1.000**	**0.315**	**0.205**	**0.418**	**0.245**

Figure 4.7: See caption 4.6 for a description of the table. Since precision is 1 minus the error of equation (4.1), the total precision on both the training and test sets matches the average performance shown in Figure 4.5.

†The model predicts words *without* access to the test image labels. We provide this information for completeness.

‡We can use the manual annotations for evaluation purposes, but we underline the fact that an agent would not have access to the information presented in the ANNOTATION % column.

Predicition *Robomedia* data set using *Grid*

	LABEL %		ANNOTATION %‡		*gMAP1* PR.		*dML1MRF* PR.	
WORD	**TRAIN**	**TEST†**	**TRAIN**	**TEST**	**TRAIN**	**TEST**	**TRAIN**	**TEST**
backpack	0.018	0.015	0.007	0.004	0.163	0.182	0.235	0.162
boxes	0.020	0.015	0.033	0.038	0.166	0.147	0.334	0.328
cabinets	0.075	0.077	0.105	0.112	0.543	0.527	0.300	0.251
ceiling	0.064	0.077	0.056	0.074	0.308	0.305	0.295	0.197
chair	0.137	0.134	0.111	0.103	0.167	0.130	0.173	0.129
computer	0.068	0.067	0.054	0.063	0.108	0.088	0.144	0.150
cooler	0.002	0.005	0.001	0.002	0.125	0.125	0.500	0.083
door	0.079	0.067	0.056	0.068	0.380	0.360	0.189	0.184
face	0.011	0.021	0.001	0.003	0.019	0.033	0.286	0.185
fan	0.018	0.021	0.009	0.011	0.205	0.174	0.522	0.439
fi lers	0.029	0.036	0.022	0.033	0.152	0.116	0.230	0.161
fbor	0.002	0.005	0.002	0.004	0.254	0.167	0.483	0.037
person	0.031	0.026	0.028	0.017	0.404	0.297	0.242	0.125
poster	0.031	0.046	0.019	0.038	0.303	0.260	0.229	0.244
robot	0.018	n/a	0.012	n/a	0.190	n/a	0.450	n/a
screen	0.037	0.057	0.042	0.051	0.373	0.312	0.534	0.468
shelves	0.075	0.088	0.111	0.129	0.141	0.160	0.294	0.243
table	0.040	0.021	0.035	0.029	0.133	0.044	0.296	0.106
tv	0.022	0.036	0.006	0.010	0.183	0.171	0.509	0.464
wall	0.106	0.098	0.118	0.102	0.131	0.122	0.201	0.193
whiteboard	0.119	0.088	0.172	0.109	0.250	0.231	0.155	0.142
Totals	**1.000**	**1.000**	**1.000**	**1.000**	**0.266**	**0.228**	**0.279**	**0.223**

Figure 4.8: See caption 4.6 for a description of the table. Since precision is 1 minus the error of equation (4.1), the total precision on both the training and test sets matches the average performance shown in Figure 4.3.
†The model predicts words *without* access to the test image labels. We provide this information for completeness.
‡We can use the manual annotations for evaluation purposes, but we underline the fact that an agent would not have access to the information presented in the ANNOTATION % column.

DATA SET	**y**	**mean CIE-Lab**			**stddev CIE-Lab**		
CorelB NCuts	0.444	0.491	0.278	0.406	0.285	0.275	0.360
CorelB Grid	0.214	0.499	0.430	0.530	0.418	0.291	0.395
Robomedia NCuts	0.270	0.247	0.114	0.218	0.219	0.166	0.250
Robomedia Grid	0.359	0.438	0.238	0.396	0.292	0.314	0.378

Figure 4.9: The feature weighting hyper-parameters τ averaged over the trials. A small τ_f implies low importance in the classification of concepts on dimension f.

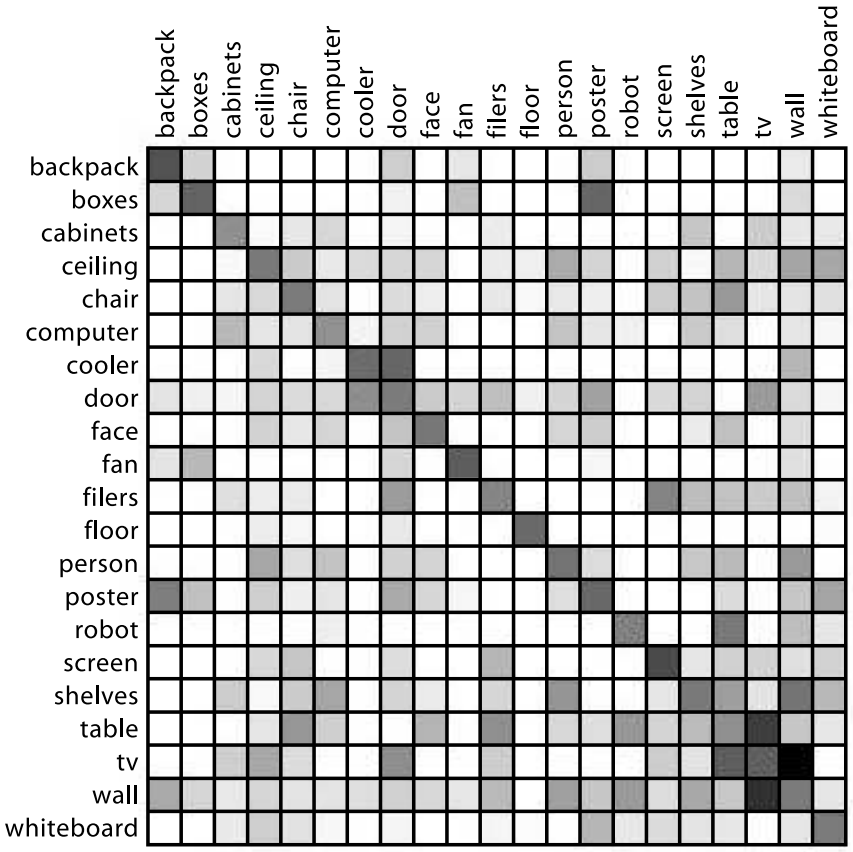

Figure 4.10: The matrix ψ averaged over trials on the *Robomedia* data using a grid segmentation. The darker squares indicates a strong neighbour relationship between two concepts. White indicates that the words were never observed next to each other during training.

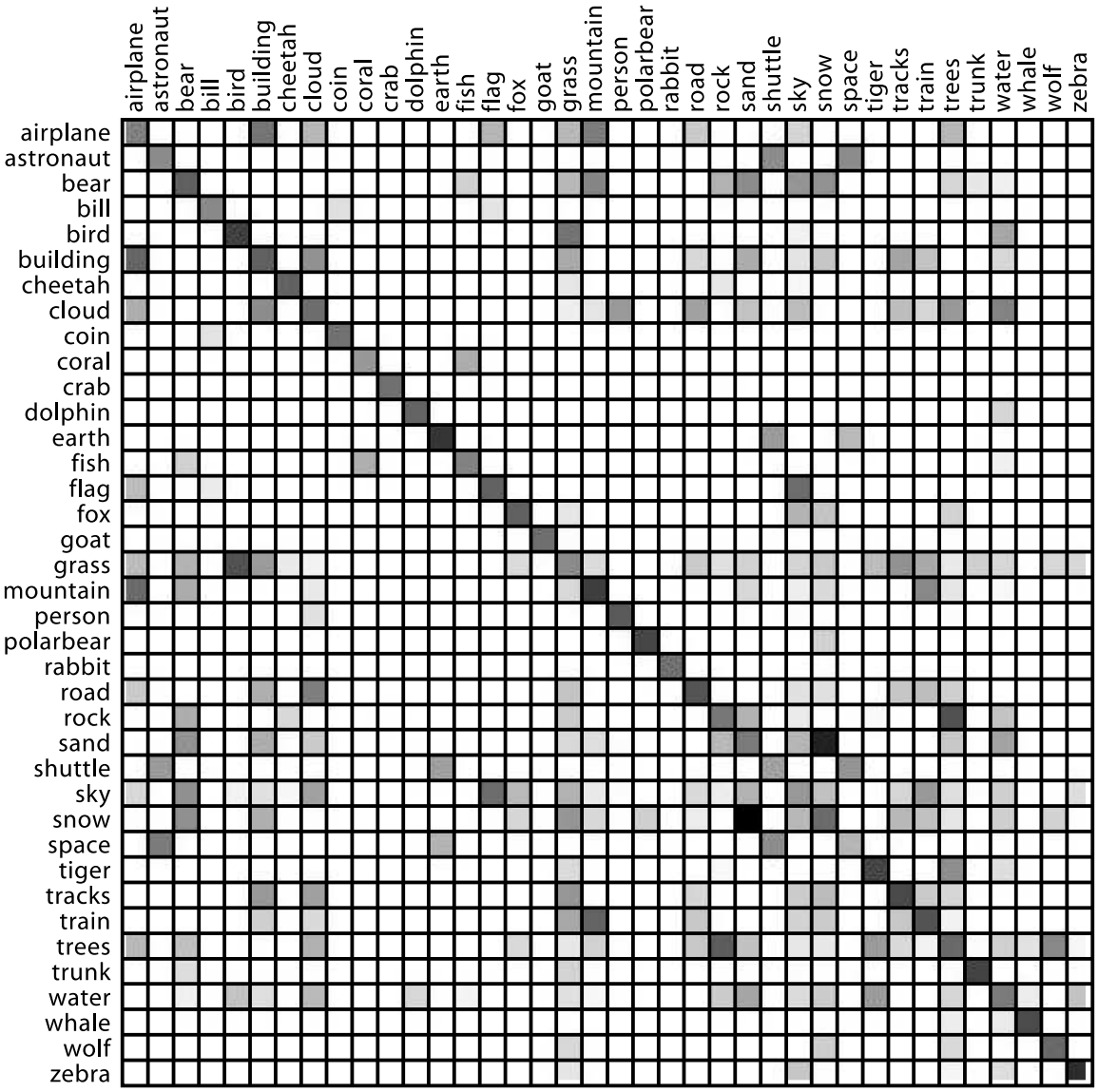

Figure 4.11: The matrix ψ averaged over *CorelB* trials using the grid segmentation. The darker squares indicates a strong neighbour relationship between two concepts. White indicates that the words were never observed next to each other during training.

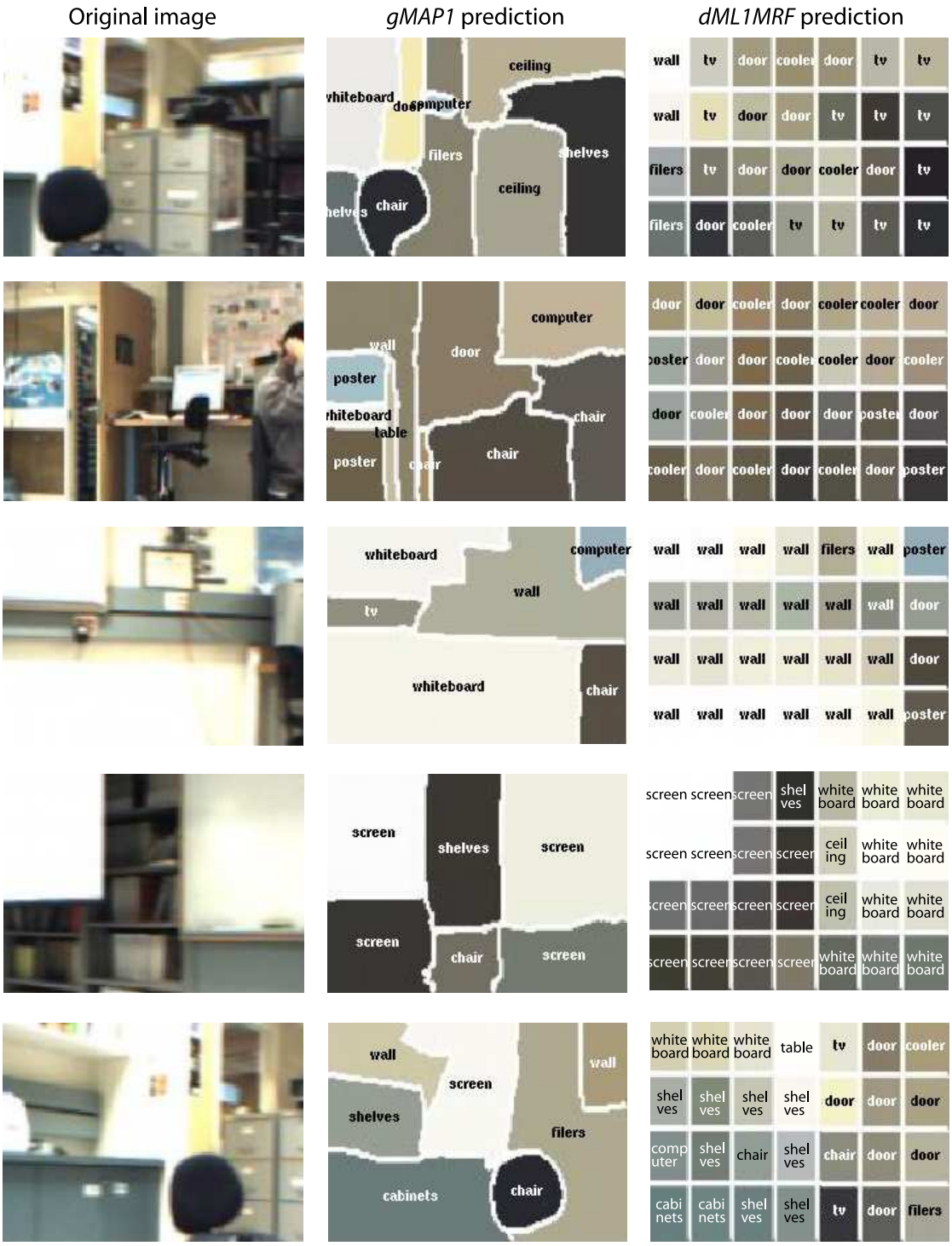

Figure 4.12: Selected model annotations on the *Robomedia* training set. The first column shows the original image. The second column shows the most likely predictions of the *gMAP1* model using the Normalized Cuts segmentation. The final column shows the predictions of the *dML1MRF* model given the grid segmentation. It is important to note that the model annotations are probabilistic, and for clarity we only display the classification with the highest probability. Also, the model does *not* have access to the labels when making these predictions.

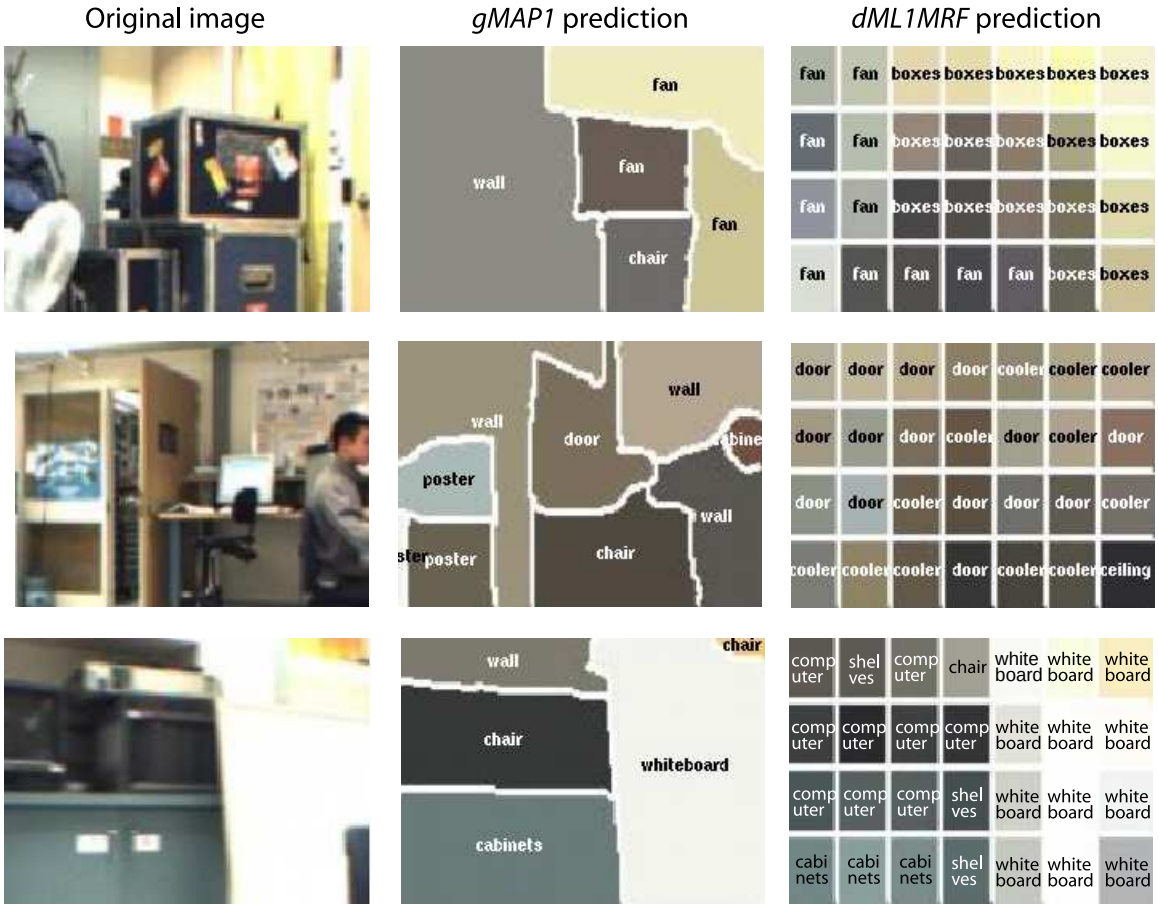

Figure 4.13: Selected model predictions on the *Robomedia* test set. See the caption in Figure 4.12 for more details.

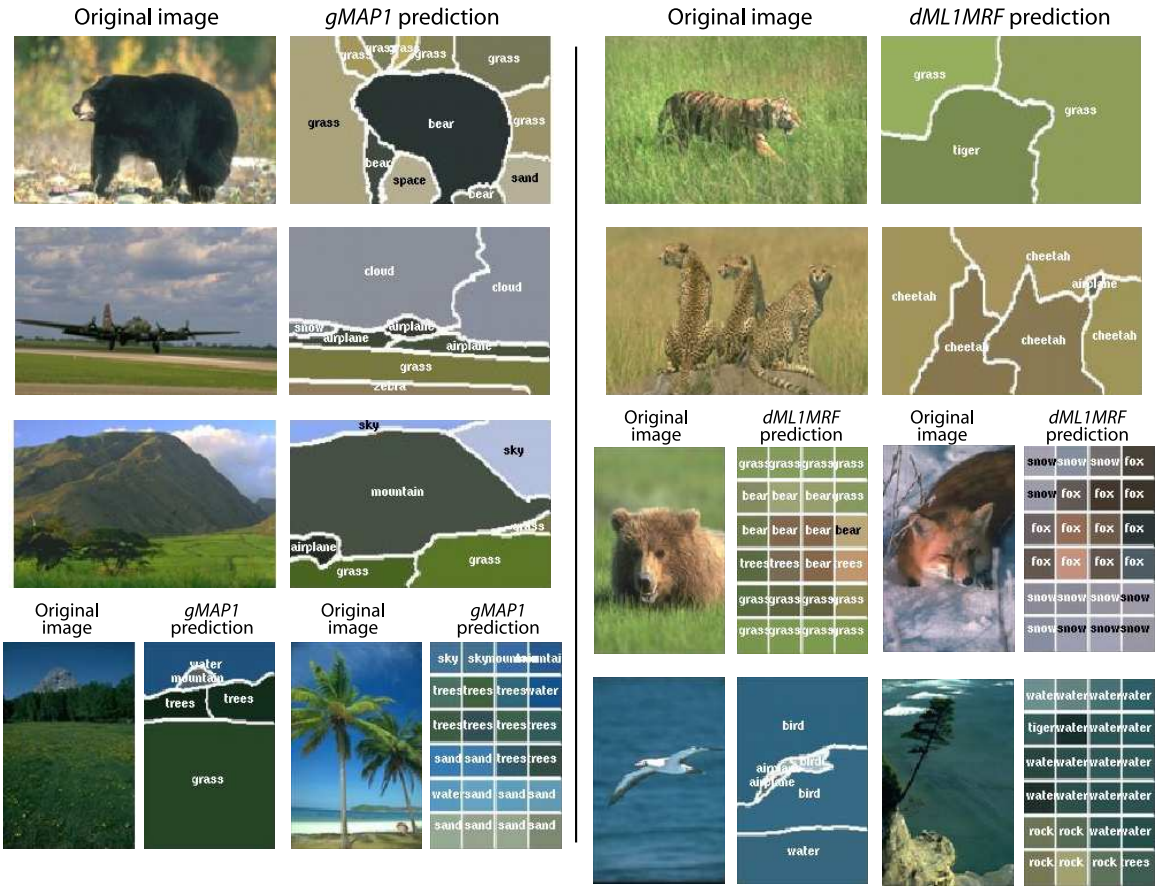

Figure 4.14: Selected model predictions on the *CorelB* training set. Note that for clarity we only display the classification with the highest probability, and the model does *not* have access to the labels when annotating the blobs in the scene.

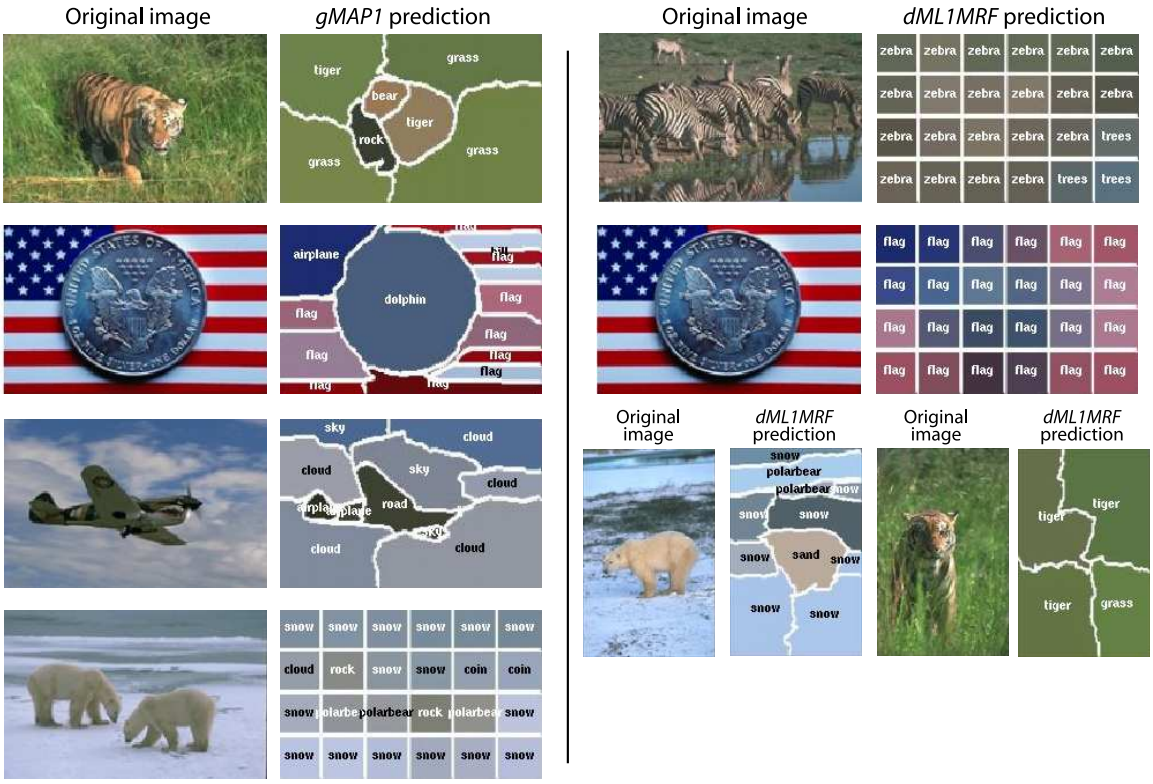

Figure 4.15: Selected model predictions on the *CorelB* test set. Note that for clarity we only display the classification with the highest probability, and the model does *not* have access to the labels when annotating the blobs in the scene.

Chapter 5

Discussion and Conclusions

We presented a set of models that learn to classify objects on weakly-annotated images that are collected without a heavy burden on the user.

We showed that assuming independence between objects reduces the ability to produce accurate scene annotations. We achieve nicer results and improved scalability on the *Corel* data with the independence assumption removed, since context helps classify objects when the blob features are ineffective. Poorly-classified words may be easier to label when paired with easily-separable concepts. Spatial context purges inconfident predictions, and thus acts as a smoothing agent for scene annotations. Moreover, the introduction of a Markov random field to the statistical model opens the door to richer representations. We would also like to investigate the value and computational feasability of hierarchical representations that incorporate spatial relations at different scales, as in [Freeman *et al.*, 2000]. We have not explored the influence of priors on the spatial potentials in much detail. It is clear that the choice of prior plays an important role, but at this point our learning techniques are too rudimentary to formulate a proper analysis.

We also demonstrated empirically that Gaussian translation rules are better than discrete translation rules because they facilitate the data association task. However, the set of features is inadequate for describing the objects and therefore linear classification models (*gML1* and *gMAP*) are infeasible in large domains. For instance, cars cannot be classified based on colour because they exhibit a wide variation in colour. As the model views more examples of cars, the Gaussian widens to eventually encompass the entire feature space. The discrete model poses an advantage in this respect because it permits more clusters than concepts, and thus a nonlinear classification. This consideration motivates an approach that combines the advantages of a non-linear classification with a mixture model, such as a hierarchical mixture model [Barnard *et al.*, 2003a] or a mixtures-of-experts [Jacobs *et al.*, 1991].

Feature weighting using shrinkage priors is an improvement over the standard mixture model since it stabilises the cluster covariances and prevents over-fitting on noise. Our experiments show that the shrinkage priors place low importance on the features we suspect have no relevance to the concepts they describe.

Normalized Cuts is widely considered to produce good segmentations of scenes. It is therefore surprising that experiments indicate crude segmentations work equally well or better for object recognition. Upon further consideration, our results are indeed sensible. We are attempting to achieve an optimal balance between loss of information through compression and adeptness in the data association problem through increasing the mutual information between blobs and labels. The Information Bottleneck frames the compression versus information tradeoff in an elegant manner [Slonim and Tishby, 1999]. The Normalized Cuts segmentation we use tends towards high compres-

sion and consequently fuses blobs that contain different objects. In turn, this introduces significant levels of noise to the classification data. A high level of compression is necessary for most models, but the spatial context model can cope with over-segmented scenes. Hence, *dML1MRF* performs much better with smaller segments even if they ignore object boundaries, as with the grid segmentations. Since model *dML1MRF* fuses blobs with high affinities, we claim it is a small step towards a model that learns both scene segmentations and annotations concurrently. One considerable obstacle in the development of such a model is that we need an evaluation scheme that uniformly evaluates the quality of segmentations combined with annotations.

We represent objects using a very simple set of colour features. Object classes overlap a great deal in low-dimension feature spaces, and thus poses an obstacle for the scalability of our models. We would like to include texture, although this poses new problems regarding texture scale. Shape is a very important component for object representation, but it is difficult to implement because we cannot rely on the segmentations to provide useful shape information. Our translation models do not require the scene to be segmented, so we can consider a blob representation using point features [Lowe, 1999; Mikolajczk and Schmid, 2003].

One important issue we did not address explicitly is on-line learning. Presently, we train our models assuming that all the images are collected at one time. Recent research shows that porting batch learning to an on-line process using EM does not pose significant challenges [Sato and Ishii, 2000; Brochu *et al*., 2003].

We have yet to answer a fundamental question regarding object recognition as machine translation: which objects are easy to annotate and which are difficult? In other words, how effective are the features and spatial context potentials in determining a proper classification boundary between the concepts? At this point we are not at a position to answer this question because it is confounded by the effect of the segmentation, variation over individual EM trials, and the quality of the data. The latter definitely rings true in our experiments. For example, we found that our models often associated the word “polar bear” with a pale blue patch because polar bears always occured next to blue ice in our data sets. Since the composition of the data factors into the success of our models, one could use active learning to implement a scheme for requesting images based on what is believed to be the most valuable for classification. This has yet to be explored for object recognition, but it has been applied to the related domain of image retrieval [Tong and Chang, 2001].

EM makes our results difficult to analyse because of the variation on individual trials. We can consider the learning process on a particular mixture component as a balancing act between finding a Gaussian that covers the space of blobs co-occuring with that concept, and making sure that this is done without hindering the prediction power on other concepts. Since the concepts tend to overlap a great deal, the point estimate at a local maximum of the likelihood surface corresponds to a preference towards the correct classification of certain concepts at the expense of others. This explains the high variation in the precision on individual words relative to the more moderate variance on overall performance. Computing a distribution over the parameters using Bayesian learning and MCMC would help us identify weaknesses in the data and permit a more rigorous analysis of our results. Ultimately, a model that segments and translates simultaneously, and learns using the Bayesian paradigm should offer more satisfying answers to the fundamental questions we raise on object recognition.

Bibliography

[Andrews *et al.*, 2002] Stuart Andrews, Ioannis Tsochantaridis, and Thomas Hofmann. Multiple instance learning with generalized support vector machines. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, 2002.

[Andrews *et al.*, 2003] Stuart Andrews, Ioannis Tsochantaridis, and Thomas Hofmann. Support vector machines for multiple instance learning. *Advances in Neural Information Processing Systems*, 15, 2003.

[Barnard and Forsyth, 2001a] Kobus Barnard and David A. Forsyth. Clustering art. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2001.

[Barnard and Forsyth, 2001b] Kobus Barnard and David A. Forsyth. Learning the semantics of words and pictures. In *Proceedings of the International Conference on Computer Vision*, 2001.

[Barnard *et al.*, 2003a] Kobus Barnard, Pinar Duygulu, David A. Forsyth, Nando de Freitas, David M. Blei, and Michael I. Jordan. Matching words and pictures. *Journal of Machine Learning Research*, 3:1107–1135, 2003.

[Barnard *et al.*, 2003b] Kobus Barnard, Pinar Duygulu, Raghavendra Guru, Prasad Gabbur, and David A. Forsyth. The Effects of segmentation and feature choice in a translation model of object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2003.

[Berger, 1997] Adam Berger. The Improved iterative scaling algorithm: a gentle introduction, December 1997. Carnegie Mellon University.

[Besag, 1986] Julian Besag. On the Statistical analysis of dirty pictures. *Journal of the Royal Statistical Society*, 48(3):259–302, 1986.

[Blei and Jordan, 2003] David Blei and Michael I. Jordan. Modeling annotated data. In *Proceedings of the international ACM SIGIR conference on research and development in information retrieval*, 2003.

[Borra and Sarkar, 1997] Sudhir Borra and Sudeep Sarkar. A Framework for performance characterization of intermediate-level grouping modules. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(11):1306–1312, November 1997.

[Brochu *et al.*, 2003] Eric Brochu, Nando de Freitas, and Kejie Bao. The Sound of an album cover: probabilistic multimedia and IR. In *Proceedings of the Workshop on Artificial Intelligence and Statistics*, 2003.

[Brown *et al.*, 1993] Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, and Robert L. Mercer. The Mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2):263–311, 1993.

[Cadez and Smyth, 1998] Igor Cadez and Padhraic Smyth. Parameter estimation for inhomogeneous markov random fields using pseudolikelihood, November 1998. University of California, Irvine.

[Carbonetto and de Freitas, 2003] Peter Carbonetto and Nando de Freitas. Why can't José read? The problem of learning semantic associations in a robot environment. In *Proceedings of the NAACL Human Language Technology Conference Workshop on Learning Word Meaning from Non-Linguistic Data*, 2003.

[Carbonetto *et al.*, 2003] Peter Carbonetto, Nando de Freitas, Paul Gustafson, and Natalie Thompson. Bayesian feature weighting for unsupervised learning, with application to object recognition. In *Proceedings of the Workshop on Artificial Intelligence and Statistics*, 2003.

[Duygulu *et al.*, 2002] Pinar Duygulu, Kobus Barnard, Nando de Freitas, and David A. Forsyth. Object recognition as machine translation: learning a lexicon for a fixed image vocabulary. In *Proceedings of the European Conference on Computer Vision*, 2002.

[Duygulu, 2003] Pinar Duygulu. *Translating images to words: a novel approach for object recognition.* PhD thesis, Graduate School of Natural and Applied Applied Sciences at the Middle East Technical University, February 2003.

[Edwards *et al.*, 2003] Jaety Edwards, Ryan White, and David A. Forsyth. Words and pictures in the news. In *Proceedings of the NAACL Human Language Technology Conference Workshop on Learning Word Meaning from Non-Linguistic Data*, 2003.

[Elinas *et al.*, 2002] Pantelis Elinas, Jesse Hoey, Darrel Lahey, Jefferson D. Montgomery, Don Murray, Stephen Se, and Jim J. Little. Waiting with José, a vision-based mobile robot. In *Proceedings of the International Conference on Robotics and Automation*, 2002.

[Fergus *et al.*, 2003] Rob Fergus, Pietro Perona, and Andrew Zisserman. Object class recognition by unsupervised scale-invariant learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2003.

[Forsyth, 2003] David A. Forsyth. *Computer Vision: A Modern Appoach*. Prentice Hall, 2003.

[Freeman *et al.*, 2000] William T. Freeman, Egon C. Pasztor, and Owen T. Carmichael. Learning low-level vision. *International Journal of Computer Vision*, 40(1):23–47, October 2000.

[Huang and Zabih, 1998] Jing Huang and Ramin Zabih. Combining color and spatial information for content-based image retrieval. In *Proceedings of the European Conference on Digital Libraries*, 1998.

[Jacobs *et al.*, 1991] Robert A. Jacobs, Michael I. Jordan, Geoffrey E. Hinton, and Stephen J. Nowlan. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991.

[Jordan, 2003] Michael I. Jordan. Introduction to graphical models, June 2003. Unpublished.

[Lowe, 1999] David G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision*, pages 1150–1157, 1999.

[Marr, 1982] David Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. W.H. Freeman and Company, 1982.

[Martin *et al.*, 2001] David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A Database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings of the International Conference on Computer Vision*, 2001.

[Mézard and Zecchina, 2002] Marc Mézard and Riccardo Zecchina. The Random k-satisfiability problem: from an analytic solution to an efficient algorithm. *Physical Review*, 66(056126), November 2002.

[Mikolajczk and Schmid, 2003] Krystian Mikolajczk and Cordelia Schmid. A Performance evaluation of local descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2003.

[Murphy *et al.*, 1999] Kevin Murphy, Yair Weiss, and Michael I. Jordan. Loopy belief propagation for approximate inference: an empirical study. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, volume 15. Morgan Kaufmann Publishers, 1999.

[Neal and Hinton, 1998] Radford Neal and Geoffrey Hinton. A View of the EM algorithm that justifies incremental, sparse and other variants. In Michael I. Jordan, editor, *Learning in Graphical Models*, 1998.

[Paget, 1999] Rupert D. Paget. *Nonparametric Markov random field models for natural texture images*. PhD thesis, University of Queensland, February 1999.

[Sato and Ishii, 2000] Masa-aki Sato and Shin Ishii. On-line EM algorithm for the Normalized Gaussian Network. *Neural Computation*, 12(2):407–432, 2000.

[Seymour, 1993] Lynne Seymour. *Parameter estimation and model selection in image analysis using Gibbs-Markov random fields*. PhD thesis, University of North Carolina, Chapel Hill, 1993.

[Shi and Malik, 1997] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1997.

[Slonim and Tishby, 1999] Noam Slonim and Naftali Tishby. Agglomerative Information Bottleneck. In *Advances in Neural Information Processing Systems*, 1999.

[Smith and Chang, 1996] John R. Smith and Shih-Fu Chang. VisualSEEk: a fully automated content-based image query system. In *ACM Multimedia*, 1996.

[Teh and Welling, 2002] Yee Whye Teh and Max Welling. The Unified propagation and scaling algorithm. *Advances in Neural Information Processing Systems*, 14, 2002.

[Teh, 2003] Yee Whye Teh. *Bethe Free energy and contrastive divergence approximations for undirected graphical models*. PhD thesis, University of Toronto, February 2003.

[Tham, 2002] Shien-Shin Tham. *Markov Chain Monte Carlo for sparse Bayesian regression and classification*. PhD thesis, University of Melbourne, August 2002.

[Tieu and Viola, 2000] Kinh Tieu and Paul Viola. Boosting image retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2000.

[Tong and Chang, 2001] Simon Tong and Edward Chang. Support vector machine active learning for image retrieval. In *ACM Multimedia*, 2001.

[Yedidia *et al.*, 2001] Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. Generalized belief propagation. *Advances in Neural Information Processing Systems*, 13:689–695, 2001.

[Yedidia *et al.*, 2002] Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. Constructing free energy approximations and generalized belief propagation algorithms. Technical report, August 2002. Mitsubishi Electric Research Laboratories.

Appendix A

Exponential family probability distributions

Here is a list of exponential probability density functions used in the translation models. Variable x be defined in the Euclidean space $\mathbb{R}^k$ for the normal distribution, $x \in \mathbb{R}$ for the inverse-Gamma distribution and $x \in \mathbb{R}^{k \times k}$ for the inverse-Wishart distribution, where k is the number of features.

Name	**Density function**	**Parameters**
Normal	$\mathcal{N}(x \mid \mu, \Sigma) = \lvert 2\pi\Sigma \rvert^{-1/2}$ $\times \exp\big(-0.5(x-\mu)^T \Sigma^{-1}(x-\mu)\big)$	mean μ, $k \times k$ positive semi-definite covariance Σ
Inverse Gamma	$\mathcal{I}g(x \mid \alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{-(\alpha+1)} e^{-\beta/x}$	scale $a > 0$, shape $b > 0$
Inverse Wishart	$\mathcal{IW}_k(x \mid \nu, \Sigma) =$ $(2^{\nu k/2} \pi^{k(k-1)/4} \prod_{i=1}^{k} \Gamma(0.5(\nu+1-i)))^{-1}$ $\times \lvert\Sigma\rvert^{\nu/2} \lvert x \rvert^{-0.5(\nu+k+1)} \exp\big(-0.5 \operatorname{trace}(\Sigma x^{-1})\big)$	$\nu > k+1$, symmetic positive definite $k \times k$ matrix Σ

Appendix B

EM derivations

B.1 Model *dML1*

To derive the E Step equation for the latent variables a_{nj} given the obesrved variables and the current parameter values, we apply Bayes' theorem.

$$\begin{aligned}\widetilde{p}(a_{nj}=i) &= p(a_{nj}=i \mid b_{nj}, w_{ni}, t)\\ &\propto p(b_{nj} \mid a_{nj}=i, w_{ni}, t)\, p(a_{nj}=i \mid w_{ni}, t)\\ &= t(b_{nj} \mid w_{ni})\end{aligned}$$

We require all the probabilities $p(a_{nj}=i)$ for all i sum to unity, so the E Step becomes

$$\widetilde{p}(a_{nj}=i) = \frac{t(b_{nj} \mid w_{ni})}{\sum_{k=1}^{L_n} t(b_{nj} \mid w_{nk})} \tag{B.1}$$

Next we derive the M Step equation for parameter t. From equation 3.1, the complete log likelihood of model *dML1* is

$$\begin{aligned}\widetilde{\ell}_c(\widetilde{p}, \theta) &= \mathbb{E}_{\widetilde{p}} \log p(a, b \mid w, \theta)\\ &= \sum_{n=1}^{M}\sum_{j=1}^{M_n}\sum_{i=1}^{L_n} \widetilde{p}(a_{nj}=i) \log t(b_{nj} \mid w_{ni})\end{aligned} \tag{B.2}$$

and the incomplete log likelihood is

$$\widetilde{\ell}(\widetilde{p}, \theta) = \sum_{a} \widetilde{\ell}_c(\widetilde{p}, \theta) = \sum_{n=1}^{N}\sum_{j=1}^{L_n}\sum_{i=1}^{L_n} \log t(b_{nj} \mid w_{ni})$$

Next, we formulate the Lagrangian which takes into account the constraints on the parameters, namely that the translation probabilities $t(b_j \mid w_i)$ must sum to one for all blob tokens j. The Lagrangian is the complete log likelihood (B.2) with constraints tacked on the end

$$\widetilde{\ell}_c^{\star}(\widetilde{p}, \theta) = \sum_{n=1}^{M}\sum_{j=1}^{M_n}\sum_{i=1}^{L_n} \widetilde{p}(a_{nj}=i) \log t(b_{nj} \mid w_{ni}) + \sum_{i=1}^{W} \lambda_i \left(1 - \sum_{j=1}^{B} t(b_j \mid w_i)\right) \tag{B.3}$$

λ_i is the Lagrange multiplier associated with word token i. The derivative of the Lagrangian $\widetilde{\ell}_c^{\star}(\widetilde{p},\theta)$ with respect to the single translation table entry $t(b^\star|w^\star)$ is

$$\frac{\partial \widetilde{\ell}_c^{\star}}{\partial t(b^\star|w^\star)} = \frac{1}{t(b^\star|w^\star)} \sum_{n=1}^{N}\sum_{j=1}^{M_n}\sum_{i=1}^{L_n} \delta(b_{nj}=b^\star)\,\delta(w_{ni}=w^\star)\,\widetilde{p}(a_{nj}=i) - \lambda^\star \tag{B.4}$$

where $\delta(x=x^\star)$ is the Delta-dirac function and is defined to be equal to 1 if $x=x^\star$; otherwise, $\delta(x=x^\star)=0$. Setting (B.4) to 0 and summing over all blobs since $\sum_{j=1}^{B} t(b_j\,|\,w_i)=1$ as previously specified, we end up with

$$\begin{aligned}\lambda^\star &= \sum_{k=1}^{B}\sum_{n=1}^{N}\sum_{j=1}^{M_n}\sum_{i=1}^{L_n} \delta(b_{nj}=b_k)\,\delta(w_{ni}=w^\star)\,\widetilde{p}(a_{nj}=i)\\ &= \sum_{n=1}^{N}\sum_{j=1}^{M_n}\sum_{i=1}^{L_n} \delta(w_{ni}=w^\star)\,\widetilde{p}(a_{nj}=i)\end{aligned} \tag{B.5}$$

Plugging (B.5) back into (B.4) and setting the equation to 0, we get the M Step update

$$t^{(new)}(b^\star|w^\star) = \frac{\sum_{n=1}^{N}\sum_{j=1}^{M_n}\sum_{i=1}^{L_n}\delta(b_{nj}=b^\star)\,\delta(w_{ni}=w^\star)\,\widetilde{p}(a_{nj}=i)}{\sum_{n=1}^{N}\sum_{j=1}^{M_n}\sum_{i=1}^{L_n}\delta(w_{ni}=w^\star)\,\widetilde{p}(a_{nj}=i)} \tag{B.6}$$

B.2 Model *gML1*

The E Step derivation follows the same logic presented in B.1.

$$\begin{aligned}\widetilde{p}(a_{nj}=i) &= p(a_{nj}=i\,|\,b_{nj},w_{ni},t)\\ &= \frac{\sum_{c=1}^{W}\delta(w_{ni}=c)\mathcal{N}(b_{nj}\,|\,\mu_c,\Sigma_c)}{\sum_{k=1}^{L_n}\sum_{d=1}^{W}\delta(w_{nk}=d)\mathcal{N}(b_{nj}\,|\,\mu_d,\Sigma_d)}\end{aligned} \tag{B.7}$$

Next we derive the M Step for the parameters μ_c and Σ_c associated with word cluster c. Taking the log of equation (3.4) and omitting constant terms, we get

$$\begin{aligned}\widetilde{\ell}_c(\widetilde{p},\theta) &= \mathbb{E}_{\widetilde{p}}\log p(a,b\,|\,w,\theta)\\ &= \sum_{n=1}^{N}\sum_{j=1}^{M_n}\sum_{i=1}^{L_n}\sum_{c=1}^{C}\delta_c(w_{ni})\,\widetilde{p}(a_{nj}=i)\left[-\frac{1}{2}\log|\Sigma_c| - \frac{1}{2}(b_{nj}-\mu_c)^T\Sigma_c^{-1}(b_{nj}-\mu_c)\right]\end{aligned} \tag{B.8}$$

We place no additional Lagrangian constraints on the model parameters, so the complete log likelihood is equivalent to the Lagrangian $\widetilde{\ell}_c^{\star}(\widetilde{p},\theta)$. Using (B.8), the partial derivatives of $\widetilde{\ell}_c^{\star}(\widetilde{p},\theta)$ with respect to the model parameters μ_c and Σ_c are

$$\frac{\partial \widetilde{\ell}_c^{\star}}{\partial \mu_c} = \sum_{n=1}^{N}\sum_{j=1}^{M_n}\sum_{i=1}^{L_n}\delta(w_{ni}=c)\,\widetilde{p}(a_{nj}=i)(b_{nj}-\mu_c)\Sigma_c^{-1} \tag{B.9}$$

$$\frac{\partial \widetilde{\ell}_c^{\star}}{\partial \Sigma_c} = \sum_{n=1}^{N}\sum_{j=1}^{M_n}\sum_{i=1}^{L_n}\delta(w_{ni}=c)\,\widetilde{p}(a_{nj}=i)\left[\frac{1}{2}(b_{nj}-\mu_c)^T\Sigma_c^{-2}(b_{nj}-\mu_c) - \frac{1}{2}\Sigma_c^{-1}\right] \tag{B.10}$$

Setting equations (B.9) and (B.10) to zero so as to obtain a stationary point of the log posterior and rearranging the terms, we get the following M Step update equations:

$$\mu_c^{(new)} = \frac{\sum_{n=1}^{N}\sum_{j=1}^{M_n}\sum_{i=1}^{L_n}\delta(w_{ni}=c)\,\widetilde{p}(a_{nj}=i)\,b_{nj}}{\sum_{n=1}^{N}\sum_{j=1}^{M_n}\sum_{i=1}^{L_n}\delta(w_{ni}=c)\,\widetilde{p}(a_{nj}=i)} \tag{B.11}$$

$$\Sigma_c^{(new)} = \frac{\sum_{n=1}^{N}\sum_{j=1}^{M_n}\sum_{i=1}^{L_n}\delta(w_{ni}=c)\,\widetilde{p}(a_{nj}=i)\,(b_{nj}-\mu_c)^T(b_{nj}-\mu_c)}{\sum_{n=1}^{N}\sum_{j=1}^{M_n}\sum_{i=1}^{L_n}\delta(w_{ni}=c)\,\widetilde{p}(a_{nj}=i)} \tag{B.12}$$

B.3 Model *gMAP1*

In this section we derive the M Step equations for model *gMAP1*, just as we did for previous models. The E Step remains the same as before (B.7). The log of complete posterior (3.8), disregarding constant terms, is given by

$$\begin{aligned}\widetilde{\ell}_c(\widetilde{p},\theta) = &\sum_{n=1}^{N}\sum_{j=1}^{M_n}\sum_{i=1}^{L_n}\sum_{c=1}^{W}\delta(w_{ni}=c)\,\widetilde{p}(a_{nj}=i)\left[-\tfrac{1}{2}\log|2\pi\Sigma_c| + \tfrac{1}{2}(b_{nj}-\mu_c)^T\Sigma_c^{-1}(b_{nj}-\mu_c)\right]\\ &+\sum_{c=1}^{W}\Big[-\tfrac{1}{2}(\nu+F+1)\log|2\pi\Sigma_c| - \tfrac{1}{2}\mathrm{trace}\left(\nu S\Sigma_c^{-1}\right) - \tfrac{1}{2}\log\left|2\pi T^2\right|\\ &\qquad -\tfrac{1}{2}(\mu_c-\mu^*)\left(T^2\right)^{-1}(\mu_c-\mu^*)\Big] + \sum_{f=1}^{F}\left[-(\alpha+1)\log\tau_f^2 - \beta\left(\tau_f^2\right)^{-1}\right]\end{aligned} \tag{B.13}$$

There are no Lagrangian constants, so $\widetilde{\ell}_c^{\star}(\widetilde{p},\theta) = \widetilde{\ell}_c(\widetilde{p},\theta)$. The partial derivatives of equation (B.13) with respect to the parameters μ_c, Σ_c and τ_f^2 are

$$\frac{\partial\widetilde{\ell}_c^{\star}}{\partial\mu_c} = \sum_{n=1}^{N}\sum_{j=1}^{M_n}\sum_{i=1}^{L_n}\delta(w_{ni}=c)\,\widetilde{p}(a_{nj}=i)\,\left[(b_{nj}-\mu_c)\Sigma_c^{-1} - (\mu_c-\mu^*)\left(T^2\right)^{-1}\right] \tag{B.14}$$

$$\begin{aligned}\frac{\partial\widetilde{\ell}_c^{\star}}{\partial\Sigma_c} = &\sum_{n=1}^{N}\sum_{j=1}^{M_n}\sum_{i=1}^{L_n}\delta(w_{ni}=c)\,\widetilde{p}(a_{nj}=i)\left[(b_{nj}-\mu_c)^T\Sigma_c^{-2}(b_{nj}-\mu_c) - \Sigma_c^{-1}\right]\\ &-\tfrac{1}{2}(\nu+F+1)\Sigma_c^{-1} + \tfrac{1}{2}\nu S\Sigma_c^{-2}\end{aligned} \tag{B.15}$$

$$\frac{\partial\widetilde{\ell}_c^{\star}}{\partial\tau_f^2} = \sum_{c=1}^{W}\left[-\tfrac{1}{2}\left(\tau_f^2\right)^{-1} + \tfrac{1}{2}(\mu_{c,f}-\mu_f^*)^T\left(\tau_f^2\right)^{-2}(\mu_{c,f}-\mu_f^*)\right] - (\alpha+1)\left(\tau_f^2\right)^{-1} + \beta\left(\tau_f^2\right)^{-2} \tag{B.16}$$

We denote the fth component of the mean μ_c by $\mu_{c,f}$ and equivalently μ_f^* for the hyperprior μ^*. The zero derivatives are stationary points of the log posterior. Setting the partial derivatives in equations (B.14), (B.15) and (B.16) to zero and rearranging terms, the result is the M Step updates for the

unknown parameters.

$$\mu_c^{(new)} = \frac{\sum_{n=1}^{N}\sum_{j=1}^{M_n}\sum_{i=1}^{L_n}\delta(w_{ni}=c)\,\widetilde{p}(a_{nj}=i)\,b_{nj}\Sigma_c^{-1} + \left(T^2\right)^{-1}\mu^*}{\sum_{n=1}^{N}\sum_{j=1}^{M_n}\sum_{i=1}^{L_n}\delta(w_{ni}=c)\,\widetilde{p}(a_{nj}=i) + \left(T^2\right)^{-1}} \tag{B.17}$$

$$\Sigma_c^{(new)} = \frac{\sum_{n=1}^{N}\sum_{j=1}^{M_n}\sum_{i=1}^{L_n}\delta(w_{ni}=c)\,\widetilde{p}(a_{nj}=i)\,(b_{nj}-\mu_c)^T(b_{nj}-\mu_c) + \nu S}{\sum_{n=1}^{N}\sum_{j=1}^{M_n}\sum_{i=1}^{L_n}\delta(w_{ni}=c)\,\widetilde{p}(a_{nj}=i) + \nu + F + 1} \tag{B.18}$$

$$\tau_f^{2(new)} = \frac{\sum_{c=1}^{W}(\mu_{c,f}-\mu_f^*)^T(\mu_{c,f}-\mu_f^*) + 2\beta}{W + 2\alpha + 2} \tag{B.19}$$

Using the matrix inversion lemma[1], we can rewrite the update step (B.17) as

$$\mu_c^{(new)} = T^2 A_c^{-1}\sum_{n=1}^{N}\sum_{j=1}^{M_n}\sum_{i=1}^{L_n}\delta(w_{ni}=c)\,\widetilde{p}(a_{nj}=i)\,b_{nj} + \Sigma_c A_c\,\mu^* \tag{B.20}$$

where we define

$$A_c \triangleq \sum_{n=1}^{N}\sum_{j=1}^{M_n}\sum_{i=1}^{L_n}\delta(w_{ni}=c)\,\widetilde{p}(a_{nj}=i)\,T^2 + \Sigma_c \tag{B.21}$$

In this expression, we don't have to invert the diagonal matrix T^2. This permits elements of T^2 to go to zero, allowing for feature selection.

[1]The matrix inversion lemma states that $(A+BCD)^{-1} = A^{-1} - A^{-1}B(C^{-1}+DA^{-1}B)^{-1}DA^{-1}$ for matrices A, B, C and D.