numéro d'ordre : 3508

THESE

présentée à

L'UNIVERSITÉ DES SCIENCES ET TECHNOLOGIES DE LILLE

pour obtenir le titre de

DOCTEUR EN INFORMATIQUE

par

Huicheng ZHENG

Modèles de maximum d'entropie pour la détection de la peau : application au filtrage de l'internet

Thèse soutenue publiquement le 8 novembre 2004 devant le jury :

Président :	Jean-Louis BON	Professeur, Laboratoire de Mathématiques Appliquées, Université de Lille 1
Rapporteurs :	Anuj SRIVASTAVA	Associate Professor, Department of Statistics, Florida State University
	Françoise PRÊTEUX	Professeur, Institut National des Télécommunications
Examinateur :	Jean-Marc GEIB	Professeur, LIFL, Université de Lille 1
Directeur :	Mohamed DAOUDI	Professeur, Laboratoire d'Informatique, Université de Tours
Co-Directeur :	Bruno JEDYNAK	Maître de conférences, Laboratoire de Mathématiques Appliquées, Université de Lille 1

UNIVERSITÉ DES SCIENCES ET TECHNOLOGIES DE LILLE

Laboratoire d'Informatique Fondamentale de Lille
UPRESA 8022 – UFR d'IEEA – bât. M3
cité scientifique – 59655 Villeneuve d'Ascq cedex
tel. : +33 (0)3 20 43 47 24 – fax. : +33 (0)3 20 43 65 66

ENIC Telecom Lille 1
rue Guglielmo Marconi
cité scientifique – 59658 Villeneuve d'Ascq cedex
tel. : +33 (0)3 20 33 55 77 – fax. : +33 (0)3 20 33 55 99

numéro d'ordre : 3508

THESE

présentée à

L'UNIVERSITÉ DES SCIENCES ET TECHNOLOGIES DE LILLE

pour obtenir le titre de

DOCTEUR EN INFORMATIQUE

par

Huicheng ZHENG

Maximum entropy modeling for skin detection : with an application to Internet filtering

Thèse soutenue publiquement le 8 novembre 2004 devant le jury :

UNIVERSITÉ DES SCIENCES ET TECHNOLOGIES DE LILLE

Laboratoire d'Informatique Fondamentale de Lille
UPRESA 8022 – UFR d'IEEA – bât. M3
cité scientifique – 59655 Villeneuve d'Ascq cedex
tel. : +33 (0)3 20 43 47 24 – fax. : +33 (0)3 20 43 65 66

ENIC Telecom Lille 1
rue Guglielmo Marconi
cité scientifique – 59658 Villeneuve d'Ascq cedex
tel. : +33 (0)3 20 33 55 77 – fax. : +33 (0)3 20 33 55 99

To my wife, Fang

Nothing in life is to be feared. It is only to be understood.

Marie Curie

Acknowledgements

This thesis was prepared in the Computer Science Laboratory (LIFL CNRS UMR 8022) of the University of Lille 1. It represents several years of work which could not have been done without significant interactions with and help from many people that I would like to acknowledge here.

First of all, I would like to thank my advisors Mohamed Daoudi and Bruno Jedynak. I still remember the first day I came to France and met Mohamed for the first time. During the course of my Ph.D., he not only showed me the right direction in doing research, but also always pushed me to go further in research and was always ready to discuss with me. I got to know Bruno when I began my research on skin detection. It has been a great pleasure to have Bruno's guidance since he has always an overall grasp of the situation and a remarkable mathematical insight into the problems. Once more I would like to thank both of my advisors for giving me the chance to realize this research, for their excellent advising work and for showing me their confidence on my work.

I would also like to thank Anuj Srivastava, Associate Professor at Department of Statistics in Florida State University, U.S.A., and Françoise Prêteux, Professor at INT, for their interest in my work and for accepting to write the reports of my thesis. It should be sure that their comments will make me reflect the work that has been done, and will help me to improve the quality of my thesis.

I would like to thank Jean-Louis Bon, Professor at the Applied Mathematics Laboratory of University of Lille 1, for the interest in my work, and for giving me the honor to be the president of my defense committee.

I would also like to thank Jean-Marc Geib, Professor and head of Computer Science Laboratory (LIFL) in University of Lille 1, for accepting to be the examiner of my thesis.

I would like to thank all the members in MIIRE team. My research work could not have been done so smoothly without their uncountable generous help throughout the course of my Ph.D.. Jean-Philippe has been showing me his generosity, humour and skills at Linux since I came. Thanks to Saïd, Tarik, chafik, Tierry and Sylvain for their generous help. I would also like to thank Didier for sharing code of Gibbs sampling, who left me with the impression of his enthusiasm and talent on computer. Special thanks to Hongmei, for giving me so much advice that I benefited from and for the good memory.

Thanks to the people in Computer Science and Networks Department, Jean-François, Christophe, Chabane, Gilles, Jacques and Ahmed, for the good memory in the same department. Thanks to the guys at Communication System Department on the same floor, Matthieu, Rodrique, Hassan and François. Thanks to the people in SIR, Martine, Tovo, Rodolphe and Donatien, for the quality of their service. Thank you to Dean and Amaria in International department. I would also like to thank all the personnel of ENIC Telecom Lille 1 for their known or unknown work from which I have been benefiting. Thanks to the Center for Imaging Science of the Johns Hopkins University for their support and help during my research in the U.S.A..

This work was partially supported by European Community Internet Action Plan 2117/27572-POESIA (www.poesia-filter.org) and by CNRS under the project MathSTIC.

Thanks to all my friends for their help and for the precious memory they left me with.

Finally, I would like to thank my parents, parents-in-law and family for their love and support throughout the years.

Résumé

La détection de la peau dans les images en couleur joue un rôle très important pour de nombreux problèmes de reconnaissance de formes. Citons la détection de visages ou encore la détection de corps humain dans une image. La résolution partielle de ces problèmes permet de faire des progrès dans les applications suivantes: le filtrage de l'internet, afin, par exemple, de signaler les images pornographiques, l'indexation par le contenu de catalogues d'images, ou encore la visioconférence, en permettant le suivi des visages dans des séquences d'images.

Nous disposons d'une large collection d'images qui ont été étiquetées manuellement. Ainsi, chaque pixel de chaque image de cette collection admet un label binaire: "peau" ou "non-peau".

Nous considérons successivement trois modèles probabilistes de complexité croissante pour la détection de la peau. Chaque modèle est un modèle de maximum d'entropie sur la moyenne. Tout d'abord, les contraintes ne portent que sur les lois marginales des couleurs pour chaque pixels. Puis, nous ajoutons des contraintes sur les lois jointes des labels binaires — "peau" ou "non-peau" — des pixels voisins. Le modèle obtenu est alors un champs de Markov. Dans un troisième temps, nous ajoutons des contraintes sur les couleurs des pixels voisins. Le modèle obtenu est à nouveau un champs de Markov mais contenant un très grand nombre de paramètres.

Afin, aussi bien d'estimer les paramètres de ces modèles que d'effectuer de l'inférence, c'est à dire, étant donné une image couleur, calculer la probabilité pour chaque pixel qu'il corresponde à de la peau, nous proposons d'approximer, localement, le graphe associé aux pixels par un arbre. On dispose alors de l'algorithme "iterative scaling" pour l'estimation des paramètres et de l'algorithme "belief propagation" pour l'inférence.

Nous avons effectué de nombreuses études expérimentales afin d'évaluer les performances respectives des différents modèles, en particulier en modifiant la taille et la géométrie des arbres.

Dans le cas du projet européen Poesia, nous avons utilisé notre détecteur de peau en entrée d'un système de classification utlisant la méthode des réseaux neuronaux pour bloquer les pages webs indésirable pour les enfants. Nous avons obtenu des résultats extrêmement encourageants.

Mots clefs : maximum d'entropie, détection de la peau, champs de Markov, belief propagation, filtrage de l'internet

Abstract

Skin detection consists in detecting human skin pixels from a color image. It plays an important role in various applications such as face detection, searching and filtering image content on the web, video segmentation and face/head tracking, ... However, skin detection is not an easy task, especially in the case of web images. First, skin colors may vary from person to person. Furthermore, since web images are captured under various conditions, they are subject to all kinds of noise and distortion.

In this thesis, we consider a sequence of three models for skin detection built from a large collection of labelled images. Each model is a maximum entropy model with respect to constraints concerning marginal distributions. Our models are nested. The first model is well known by practitioners. Pixels are considered as independent. It is referred to as the Baseline model in the thesis. The second model is a hidden Markov model. It includes constraints that force smoothness of the solution. The third model is a first order model. The constraints are imposed upon the two-site joint marginal probabilities. Parameter estimation as well as optimization cannot be tackled without approximations. We use tree approximations of the pixel lattice. Within it, the models are represented in terms of marginal probabilities on single sites or neighboring sites, which can be empirically estimated from the training data.

For skin probability inference given input color image, we first experimented with Gibbs sampling method. The experimental results show improved performance of skin detection compared to that of the Baseline model at the cost of higher computational load. For near-real-time performance we then replace Gibbs sampling with belief propagation algorithm, which permits to obtain exact and fast solution for skin probability at pixel locations on tree graphs. We show that the performance of belief propagation is comparable to that of Gibbs sampling.

In the European project Poesia, we apply our skin detection scheme to blocking adult images from Internet. All features are extracted from the skin detection output. The performance is quite impressive for such simple features, which is an evidence that skin detection plays a very important role in adult image detection.

Key words : skin detection, maximum entropy modeling, Markov random fields, belief propagation, Internet filtering.

Contents

List of Figures

List of Tables

List of Algorithms

Résumé (Chapitre 1)

La détection de la peau consiste à détecter les pixels de peau dans une image couleur pour obtenir une image binaire. Elle joue un rôle très important dans différentes applications telles que la détection du visage, la segmentation vidéo et le filtrage d'images adultes.

Cependant, c'est une tâche difficile à réalier à cause de la variation de la couleur de la peau et la diversité des conditions de prise de vue (lumière, bruit, . . .). Nous proposons dans cette thèse d'apprendre ces variations par l'utilisation du principe de maximum d'entropie (MaxEnt). Dans ce chapitre, nous introduisons le principe de MaxEnt pour la modélisation de la peau. Ce principe nous permet d'unifier tous les modèles statistiques que nous proposons dans cette thèse.

Nous construisons plusieurs modèles statistiques. Le premier modèle suppose l'indépendance statistique entre les pixels de peau. Les deux autres modèles sont des champs de Markov. Ils modélisent les dépendances entre pixels voisins. Le premier modèle peut être résolu analytiquement et il ne possède aucun paramètre à estimer. En revanche, les deux autres modèles possèdent des paramètres à estimer.

Pour l'estimation de ces paramètres nous proposons différentes approches. La première consiste à exploiter le concept de "Julesz Ensemble" qui permet de simuler la texture de peau. La seconde consiste à approximer localement le graphe des pixels par un arbre et de calculer la loi jointe sur cet arbre sous une forme analytique. Il n'y a alors plus de paramètres à estimer.

Une fois les paramètres estimés, on peut donc passer à l'étape d'inférence. Deux méthodes sont étudiées, l'une est basée sur l' "échantillonnage de Gibbs" et l'autre sur l'algorithme de "belief propagation".

Les résultats de la détection de la peau des chapitres précédents nous serviront pour construire un système de filtrage d'images adultes.

Chapter 1

Introduction

1.1 Skin detection

Skin detection consists in detecting human skin pixels from a color image. The system output is a binary image defined on the same pixel grid as the input image with one value for skin and the other value for the background. Fig 1.1 shows a color image and the output of skin detection.

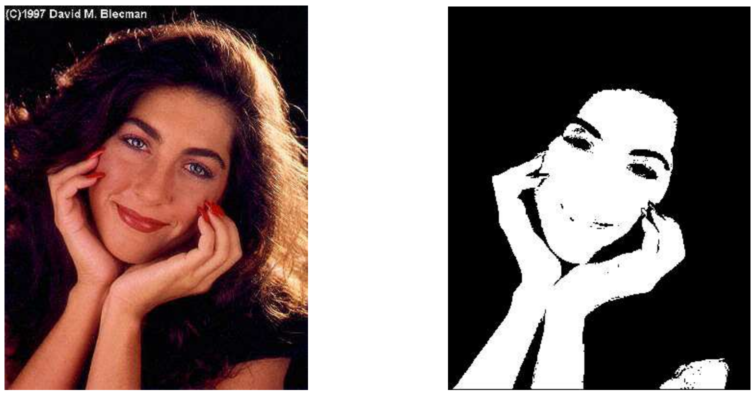


Figure 1.1: Color image with skin and the result of skin detection

Skin detection plays an important role in various applications such as face detection [TSFA00] [KPS03] [HAMJ02], searching and filtering image content on the web [WLWF98b] [WLWF98a], video segmentation [SSA04] and face/head tracking [Bir98] [JPCSV99].

However skin detection is not an easy task, especially in the case of web images. First, skin colors may vary from person to person. As shown in Fig 1.2, there are Caucasians, Africans and

Asians, etc. Furthermore, since web images are captured with devices of different characteristics under various conditions, they are subject to all kinds of noise and distortion. Fig 1.3 shows some examples.

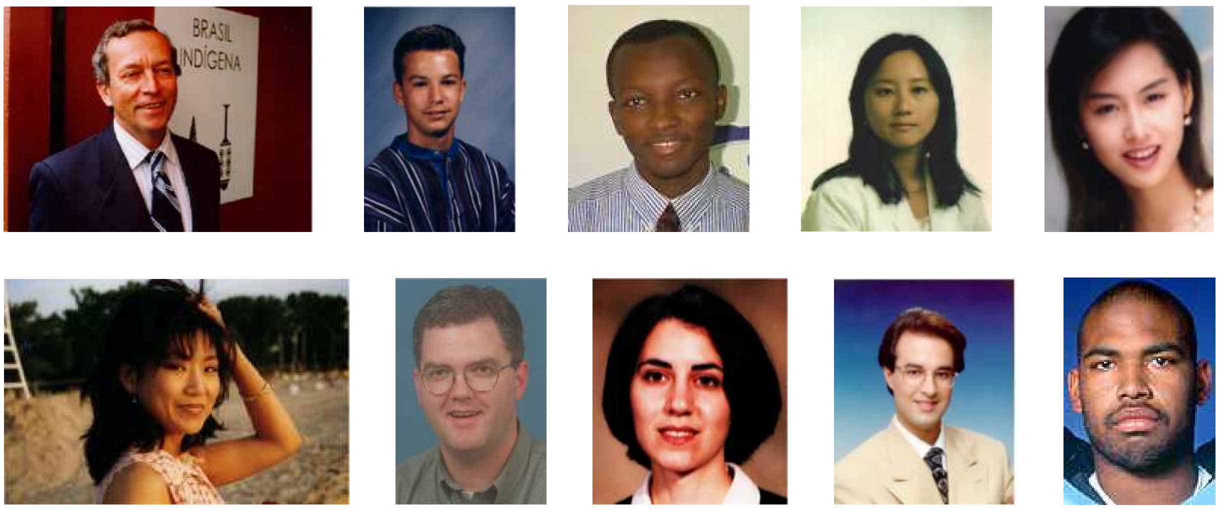


Figure 1.2: Skin colors vary from person to person.

A skin detection system is never perfect and different users use different criteria for evaluation. General appearance of the skin-zones detected, or other global criteria might be important for further processing. For quantitative evaluation, we will use *false positive rate* (FP) and *true positive rate* (TP). False positive rate is the proportion of non-skin pixels classified as skin and true positive rate is the proportion of skin pixels classified as skin. The user might wish to combine these two indicators his own way depending on the kind of error he is more willing to afford. Hence we propose a system where the output is not binary but a floating number between zero and one, the larger the value, the larger the belief for a skin pixel. The user can then apply a threshold to obtain a binary image. Error rates for all possible thresholding are summarized in the Receiver Operating Characteristic (ROC) curve.

1.2 The state of the art

Research has been performed on the detection of human skin pixels in color images and on the discrimination between skin pixels and "non-skin" pixels by use of various color models. There are mainly two lines of research in this area, first, how to model skin/non-skin so that we can discriminate skin pixels from non-skin ones, and second, which color space to choose for good

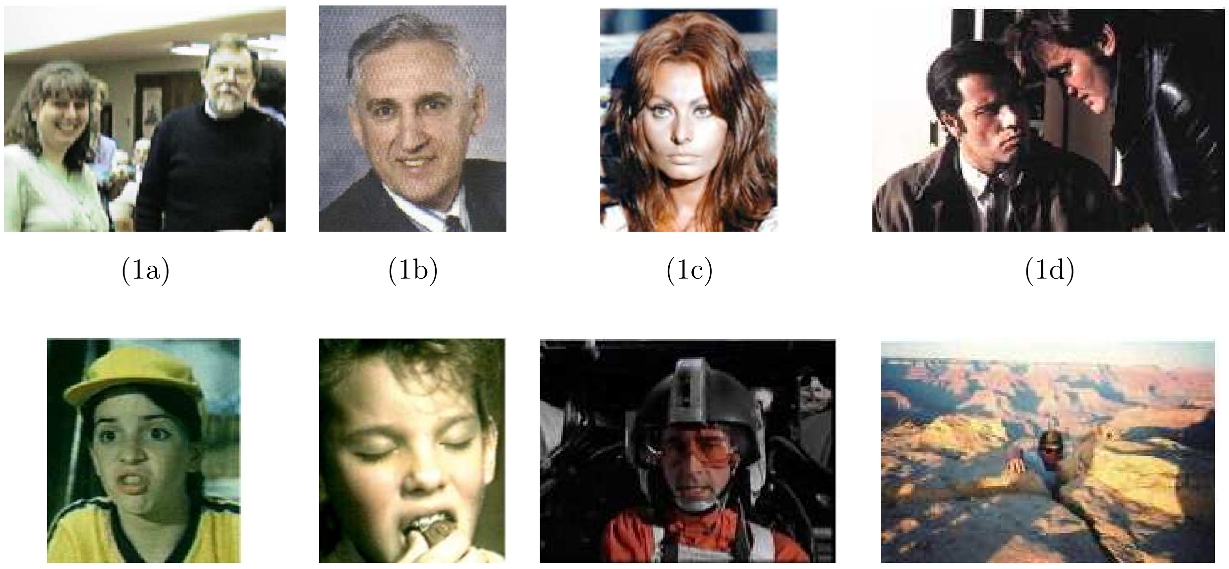


Figure 1.3: Web images are captured under very different conditions. Images (1a) and (1b) are blurred. Images (1c), (1d) and (2b) have too strong or too poor illumination on skin regions. Image (2d) shows skin regions in the shadow and mixed with the background. Image (2a) shows the color bias perhaps due to the capturing device. The pilot in (2c) wears a colorful mask on his face. All these situations creat difficulties for skin detection.

classification performance.

1.2.1 Skin modeling

There is some survey work on skin modeling such as [VSA03] [ZSQ99] [TSFA00] which summarizes research in this area. The goal of skin modeling is to build a decision rule which seperates skin pixels from non-skin ones. Following Vezhnevets et al. [VSA03], we split the work in this area into two main fields: non-parametric modeling and parametric modeling.

Non-parametric skin modeling

The research in this line aims to estimate the distribution of skin colors without explictly giving functions in parametric formulae. The result of these methods is sometimes referred to as construction of Skin Probability Map (SPM) [VSA03] [BM00] [Gom02] —assigning a probability value to each point of a discretized color space.

Bayesian models based on histograms

In [JR02] [CB00] the authors model the skin and non-skin colors through histograms. They quantize the color space C to a number of bins $c \in C$ and count the number of color pixels in each bin $N_{\text{skin}}(c)$ for skin class as well as $N_{\neg\text{skin}}(c)$ for non-skin class. Finally they normalize each bin to get the discrete conditional skin/non-skin color distribution $p(c|\text{skin})/p(c|\neg\text{skin})$. Letting N_{skin} denote the number of skin pixels and $N_{\neg\text{skin}}$ the number of non-skin pixels in the training set, we have

$$\begin{aligned} p(c|\text{skin}) &= \frac{N_{\text{skin}}(c)}{N_{\text{skin}}} && (1.1) \\ p(c|\neg\text{skin}) &= \frac{N_{\neg\text{skin}}(c)}{N_{\neg\text{skin}}} && (1.2) \end{aligned}$$

as well as the priors:

$$\begin{aligned} p(\text{skin}) &= \frac{N_{\text{skin}}}{N_{\text{skin}} + N_{\neg\text{skin}}} && (1.3) \\ p(\neg\text{skin}) &= \frac{N_{\neg\text{skin}}(c)}{N_{\text{skin}} + N_{\neg\text{skin}}} = 1 - p(\text{skin}) && (1.4) \end{aligned}$$

Then the Bayesian formula is employed to estimate the skin/non-skin probability according to the color of a given pixel:

$$\begin{aligned} p(\text{skin}|c) &= \frac{p(c|\text{skin})p(\text{skin})}{p(c|\text{skin})p(\text{skin}) + p(c|\neg\text{skin})p(\neg\text{skin})} && (1.5) \\ p(\neg\text{skin}|c) &= 1 - p(\text{skin}|c) && (1.6) \end{aligned}$$

The two-class decision is made upon a properly selected threshold Θ, $0 < \Theta < 1$. The pixel is said to be a skin pixel if $p(\text{skin}|c) > \Theta$, or a non-skin pixel if $p(\text{skin}|c) \leq \Theta$.

Self organizing map (SOM)

Devised by Kohonen in the early 80's, SOM is now one of the most popular and widely used types of unsupervised artificial neural networks [VSA03]. Brown et al. [BCL01] applied it to the detection of skin pixels. The basic SOM consists of a 2-dimensional lattice L of neurons. Each neuron $n_i \in L$ is associated with a codebook vector $v_i \in \mathbb{R}^m$ which is initialized randomly at the beginning. In [BCL01] the authors choose 2-dimensional color space, therefore $m = 2$ in their work. The lattice could be either rectangular or hexagonal. In Fig. 1.4 we show examples of both lattices as well as neighborhood of the center nodes.

In order to train a SOM, we present the training vectors sequentially to all neurons in the lattice. Each time an input vector v is sent into the SOM, a winning neuron v_w is determined according to

$$\|v_w - v\| \leq \|v_i - v\|, \forall i \in I \qquad (1.7)$$

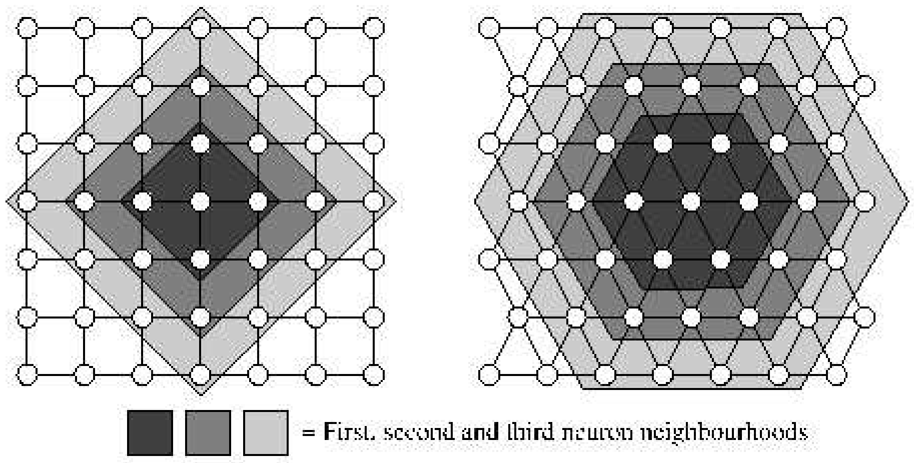


Figure 1.4: The SOM lattices with neighborhood systems [BCL01]. L: the rectangular SOM lattice; R: the hexagonal SOM lattice.

in which I is the ensemble of all the indices of neurons in the lattice. The Euclidean or Manhattan distance is generally chosen as the metric. The neurons in the neighborhood then adjust their codebook vectors according to a learning function. As training progresses, the learning rate and the size of the affected neighborhood are both decreased. The lattice gradually forms a topologically ordered mapping (or feature map) of the training data.

If necessary, a calibration phase then takes place, where labelled training data is sequentially presented to the SOM. The data label and the index of the winning neuron are recorded each time. Each neuron is then assigned the label which it takes the most. For classification, each input data just takes the label of the winning neuron.

Brown et al. [BCL01] choose the hexagonal lattice and the size of the lattice is about $16-256$. They showed that the performance of SOM is marginally better than Gaussian mixture model, while inferior to the histograms-based method in [JR02]. The good side is that it consumes less resources than histograms-based methods and could be implemented in cheap and fast SOM hardware.

Several face detection and tracking algorithms [CWY95] [ZSQ99] use a normalized lookup table(LUT) for skin pixel detection. What is in the table is in fact the conditional probability $p(c|\text{skin})$ or the likelihood of the color corresponding to skin.

Advantages of non-parametric methods are 1) they are generally fast in both training and testing; 2) the methods do not care about the shapes of the underlying distributions of the training data, thus they are less constrained in choosing working color spaces. However these models generally need large amount of storage space and lack the ability to interpolate or generalize the training data. Supposing we choose RGB color space $C = \{0, \cdots, 255\}^3$, we need 2^{24} bins for storing each skin/non-skin histogram. Jones&Regh [JR02] proposed to reduce the number of bins in the histograms by resampling the color space. Their experiments have shown that 32^3 gives almost the best performance among other tested sampling rates.

Parametric skin modeling

Parametric skin models permit to generalize the distributions. They aim to fit the distributions with some specific parameterized functions. Thus parametric models need far less storage than non-parametric models. They give more insight into the true shapes or regularity of the distributions and allow to analyze them conveniently afterwards. They have also the ability to interpolate the training data when it is sparse. Different functions could be applied according to the specific problems. The most used ones on skin/non-skin modeling are introduced below.

Single Gaussian

Single Gaussian models skin color distribution with a Gaussian probability density function (pdf):

$$p(c|\text{skin}) = \frac{1}{2\pi|\Sigma_{\text{skin}}|^{1/2}} \exp\left(-\frac{1}{2}(c-\mu_{\text{skin}})^T \Sigma_{\text{skin}}^{-1}(c-\mu_{\text{skin}})\right) \tag{1.8}$$

where μ_{skin} is the expectation and Σ_{skin} the covariance matrix of the skin color vectors. They can be estimated from the training samples as follows:

$$\mu_{\text{skin}} = \frac{1}{N_{\text{skin}}} \sum_{c \in C} N_{\text{skin}}(c)c \tag{1.9}$$

$$\Sigma_{\text{skin}} = \frac{1}{N_{\text{skin}} - 1} \sum_{c \in C} N_{\text{skin}}(c)(c-\mu_{\text{skin}})(c-\mu_{\text{skin}})^T \tag{1.10}$$

$p(c|\text{skin})$ can then be used as the likelihood of c belonging to skin. Or we can just generate another model for non-skin class as well and use the Bayesian formula to get $p(\text{skin}|c)$. Single Gaussian modeling was employed in [HAMJ02] and [ST98].

Mixture of Gaussians

Mixture of Gaussians is an extension of the single Gaussian. It has ability to represent more complex distributions than those representable with single Gaussian models. The pdf under mixture of Gaussians is represented as:

$$p(c|\text{skin}) = \sum_{i=1}^{k} w_i p_i(c|\text{skin}) \tag{1.11}$$

where p_i are the Gaussian kernels defined in (1.8), each of which is itself a Gaussian distribution, k is the number of Gaussian kernels, w_i are the weights of corresponding kernels which sum up to 1. We can build the mixture model for non-skin class similarly. Expectation Maximization (EM) algorithm is generally applied to estimate the unknown parameters (w_i, μ_i, Σ_i) as in [JR98] [TSFA00]. k should be chosen properly so that the model can explain the training data well and does not over-fit the data.

Elliptical boundary model

Proposed by Lee and Yoo [LY02], elliptical boundary model is claimed to be superior to the single Gaussian model and the mixture of Gaussians based on the experiments performed on the Compaq database [JR02]. The motivation of this model is the skewness of the skin distribution from the Gaussian distribution. In order to account for the true shape of the skin distribution, which is approximately an ellipse from the observation of the training samples, Lee and Yoo proposed to separate skin and non-skin color regions by an elliptical boundary. The model is defined as follows:

$$\Phi(c) = (c - \phi)^T \Sigma_{\text{skin}}^{-1} (c - \phi) \tag{1.12}$$

where ϕ and Σ_{skin} are to be estimated from the training skin pixel set. Before parameter estimation, the outliers are removed from the training set first, which are $0 - 5\%$ of the training skin color samples with low frequency and are assumed to be noise and negligible data. Parameters are estimated by

$$\phi = \frac{1}{|C_{\text{skin}}|} \sum_{c \in C_{\text{skin}}} c \tag{1.13}$$

$$\Sigma_{\text{skin}} = \frac{1}{N_{\text{skin}}} \sum_{c \in C_{\text{skin}}} N_{\text{skin}}(c)(c - \mu_{\text{skin}})(c - \mu_{\text{skin}})^T \tag{1.14}$$

where $|C_{\text{skin}}|$ is the number of elements in the skin color set C_{skin}, $C_{\text{skin}} \subseteq C$. μ_{skin} is the expectation of the training skin color vectors as defined in (1.9). The decision rule is simply to compare $\Phi(c)$ with a chosen threshold θ: c is skin if $\Phi(c) < \theta$ and non-skin if not. The authors tested this model on 6 chrominance color spaces. The experiments conducted by the authors reveal that this model is better than the single Gaussian and mixture of Gaussians in terms of skin/non-skin pixel classification rates, and it is fast in classification. A little drawback of this model is that it can only give the binary decision, which means it lacks the continuous information provided by the probability density function.

Parametric models have the ability of generalization. However the goodness of fit is much dependent on the shapes of the distributions, and therefore on the colorspaces chosen. Furthermore their training and testing are much slower since these involve parameter estimation procedure such as the EM algorithm and evaluation of the relatively complex model functions such as (1.8), (1.11) and (1.12).

Other skin models

In specific applications such as video tracking, skin detection is only a very primitive step and should be fast. Such requirement gives rise to simple explicit skin region definitions, such as: (R, G, B) is classified as skin if all the following conditions are satisfied:

$$\begin{cases} R > 95, G > 40, B > 20 \\ \max\{R, G, B\} - \min\{R, G, B\} > 15 \\ |R - G| > 15, R > G, R > B \end{cases} \tag{1.15}$$

Such schemes were implemented in [KPS03] [FFB96] [JPCSV99]. The main difficulties faced by the practitioner are the empirical choice of a proper color space and adequate decision rules. Recently some machine learning algorithms for the automatic construction of the proper color space as well as the decision rule is proposed [GM02].

Another line of research takes place in the dynamic modeling. That is, the model itself is adaptable to the changing conditions (person, camera, lighting, background). Such models are mainly used in special applications such as face tracking [OBP96] [SMHL00] [YA98].

1.2.2 Color spaces for skin modeling

Color is perceived by human being as a combination of so-called tristimuli R (red), G (green) and B (blue). They are usually called the three primary colors. Some other color representations (spaces) can be derived through linear/non-linear combinations of R, G and B. Different color spaces have been proposed for various purposes [CJSW01]. Many of these spaces have been applied to the problem of skin color modeling.

RGB

RGB color space originated from CRT displaying applications. Color is represented as a combination of three colored rays (red, green and blue) in this color space. A primary principle of colorimetry is that any color could be represented by such a combination and that this combination is unique. It is one of the most commonly used color space for image displaying and processing. However there exists high correlation between these three channels, e.g., when the illumination changes, all three channels will change accordingly. Furthermore it is not perceptually uniform across the color range. This color space is implemented in [BM00] [JR02].

Normalized RGB

In light of the high correlation inside RGB color space, researchers have been looking for methods to decorrelate the channels or to decrease the dimensionality of the color space. A very simple way is to normalize the three components in the RGB representation according to:

$$r = \frac{R}{R+G+B}, g = \frac{G}{R+G+B}, b = \frac{B}{R+G+B} \tag{1.16}$$

The color space defined by the above transform is called *normalized RGB* color space. Since we have $r + g + b = 1$, one of the components is in fact redundant. We can simply represent the color with the couple (r, g). The dimensionality is reduced to only 2 in this way. Now the representation is independent of the brightness of the color. This, together with the simplicity of transformation attracted many researchers [BCL01] [ZSQ99] [SMHL00]. The drawback of the normalized RGB color space comes from the nonlinear transformation from the RGB color space, i.e., the representation is very noisy under low intensities.

HSV (Hue Saturation Value)

HSV color space is created by Alvy Ray Smith [Smi78] in 1978. There are many variants based on similar ideas, e.g., HSL (Hue Saturation Lightness) and HSI (Hue Saturation Intensity). This class of color spaces stem from artists' idea of tint, saturation and tone. They try to describe color intuitively. *Hue* defines the color type (such as red, blue, or yellow), *saturation* measures the colorfulness relative to its brightness [Poy97]. We use the following formulae to transform the color (R, G, B) to (H, S, V) [Wika]:

$$H = \begin{cases} (0 + \frac{G-B}{MAX-MIN}) \times 60, & \text{if} R = MAX \\ (2 + \frac{B-R}{MAX-MIN}) \times 60, & \text{if} G = MAX \\ (4 + \frac{R-G}{MAX-MIN}) \times 60, & \text{if} B = MAX \end{cases} \tag{1.17}$$

$$S = \frac{MAX - MIN}{MAX} \tag{1.18}$$

$$V = MAX \tag{1.19}$$

where R, G and B are normalized to $[0, 1]$. $MAX = \max(R, G, B)$ and $MIN = \min(R, G, B)$. The HSV color space can be visualized as Fig. 1.5. The resulting H varies from 0 to 360, indicating the angle in degrees around the color circle where the hue is located. S and V vary from 0 to 1, with 0 being the least amount and 1 being the greatest amount of saturation or value, respectively [Wika]. Note that H is undefined when $MAX = MIN$ (i.e., $S = 0$) and S is undefined when $MAX = 0$ (i.e., $V = 0$).

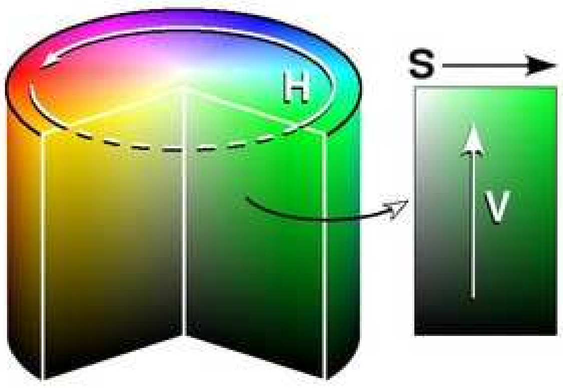


Figure 1.5: Visualization of HSV color space (taken from [Wika]). *Hue* varies along the outer circumference of a cylinder. *Saturation* varies with distance from the center. *Value* varies from bottom to top.

The intuitiveness and the seperation of chrominance and illuminance make HSV color space popular in color image segmentation as well as in skin color image segmentation [ZSQ99] [SSA04] [Bir98] [JPCSV99]. One of the disadvantages of HSV color space is the existence of singularity in H and S components. A slight change of R, G or B will cause a jump in the tranformed values near the singular points. The polar coordinate system of Hue-Saturation spaces, resulting in cyclic nature of the color space, makes it inconvenient for parametric skin color modeling that need tight cluster of skin colors for best performance [VSA03].

Other color spaces

There are some other less popular used color spaces for skin detection, e.g., TSL (Tint Saturation Lightness) in [TSFA00], YC_bC_r in [HAMJ02] and [CB00], CIE in [ZSQ99] and [YA98], RGB ratios and YIQ in [BM00], etc.

Lots of strategies in selecting color spaces aim to drop the illuminance component. This is attractive for training data generalization. Low-dimensional color spaces are easier to handle than the original 3D spaces. It is especially useful when the training data is sparse or when the researcher wants to define the skin color region empirically [KPS03] [FFB96] [JPCSV99]. The choice of color spaces is critical for dimension reduction. Gomez and Morales [GM02] suggested a way to find both suitable color space and simple decision rules with machine learning algorithms.

The performance of parametric skin color modeling depends heavily on the choice of proper color spaces. This is visible in [TSFA00] and [LY02]. On the other hand, the non-parametric methods, given enough training data, do not show much difference when choosing different color spaces as has been revealed in [ZSQ99] [BCL01] and [ATD01]. Albiol et al. [ATD01] gave a theoretical proof that if the transformation between two color spaces are invertible, then performance of the optimum skin detection scheme under either color space is the same. The authors also perfromed experiments to show that the separability of the skin and non-skin classes is independent of the color spaces chosen.

1.2.3 Performance

Vezhnevets et al. [VSA03] gave an evaluation of some state-of-the-art skin detection methods. According to their report, we have Table 1.1. As shown in this table, Bayesian modeling based on RGB histograms gives the best performance in terms of skin/non-skin pixel classification (false positive rate under the same true positive rate). The performance of Bayesian modeling based on RGB histograms is followed by that of the mixture of Gaussians in RGB. The other methods are left behind.

The Bayesian modeling based on RGB histograms proposed by Jones and Rehg [JR02] is in fact a statistically independent model in the sense that they assume no relationship among neighboring pixels. Thus they classify the pixel one by one, not taking into account the neighboring influence. However, the skin regions are not totally random, they actually tend to form regular shapes, the neighboring pixels, especially skin pixels, are much correlated. Better modeling could be to model the covariation coded in neighboring pixels. Maximum entropy modeling (MaxEnt) [Jay] is a method for inferring models from a data set. It can incorporate whatever statistical properties that could be observed from the data into a mathematical model. It opens the possibility for us to design models that take into account the statistics existing among neighboring pixels. We shall see this in the following chapter.

1.3 Thesis overview

The remainder of this thesis is organized on a chapter by chapter basis in the following manner:

Chapter 2: From independent model to Markov random fields

This chapter introduces the principle of maximum entropy, which is the basis of our skin modeling. It unifies all the skin models built in the thesis. We will build a sequence of three models in this chapter. The first one is called the Baseline model, which is well known to the practitioners and also implemented in [JR02]. It is an independent model and thus the simplest

Table 1.1: Performance of different skin detectors reported by the authors

Method	TP	FP
Bayesian modeling based on RGB histograms		
[JR02]	80%	8.5%
[JR02]	90%	14.2%
[BM00]	93.4%	19.8%
SOM in TS		
[BCL01]	78%	32%
Single Gaussian in C_bC_r		
[LY02]	90%	33.3%
Mixture of Gaussians in RGB		
[JR02]	80%	$\sim 9.5\%$
[JR02]	90%	$\sim 15.5\%$
Mixture of Gaussians in IQ		
[LY02]	90%	30%
Elliptical boundary model in CIE		
[LY02]	90%	20.9%
Thresholding of I axis in YIQ		
[BM00]	94.7%	30.2%

one of all the models we built. The other two models are Markov random fields (MRF) built by introducing appropriate constraints under the maximum entropy framework. Our first MRF model is a hidden Markov model (HMM), and the second is called the first-order model (FOM). The Baseline model can be solved analytically and exactly in terms of the empirically estimated statistics. The MRF models, however, can only be expressed in parametric forms for now. The parameter estimation issues will be tackled in the next chapter.

Chapter 3: Parameter estimation

This chapter aims to solve the unknown parameters in the two Markov models. Several parameter estimation methods are explored. The first method is to build a set of equations from the local characteristics of the MRF. We can then estimate the local characteristics by sampling from the MRF according to its constraints and counting the local characteristics or by directly counting the local characteristics from the training data. This works for the HMM but not for the FOM, since for FOM the equation system becomes huge and intractable, not to say we do not have enough training data to estimate the parameters without overfitting. The problem is solved by tree approximations for FOM since the MRF takes an analytical formula on tree graphs in terms of the empirically estimated probabilites from the training set. However, direct empirical estimation of the FOM parameters from the training set will cause over-fitting since the orginal parameter space is really huge even for small images. Two methods are studied to get around the difficulty. The first method is to decrease the dimensionality by assuming the independence of one-site marginal conditioned on the skin/non-skin label and the gradient of neighboring colors conditioned on neighboring labels. The other method is more natural. It estimates the high-dimensional parameters with the maximum entropy principle on the basis of some low-dimensional empirically estimable parameters from the training set. Now the MRF models are solved and ready for classification tasks.

Chapter 4: Gibbs sampling

This chapter deals with pixel classification through Gibbs sampling. The MRFs are equivalent to Gibbs fields. The well-established sampling theory tells us that we can sample these fields through the inherent local characteristics of the MRFs. Given an input color image, we show how we sample from the label image space through the Markov chain Monte Carlo method. The skin probabilities on pixel locations are then easily computed from the sequence of samples. Some experimental results are shown in this chapter, which reveal that the FOM is superior to the HMM and in turn to the Baseline model. However the sampling procedure generally takes a long time and depends seriously on the starting configuration. We have to look for a faster inference algorithm to meet the time requirement for real environment. This is the issue of Chapter 5.

Chapter 5: Belief propagation on tree graphs

We introduce the belief propagation (BP) algorithm in this chapter. This method aims to minimize the Bethe free energy, which is an approximation to the real free energy when the underlying graphs are not trees. This algorithm is well expressed in graphical models. Thus we begin with the introduction of general graphical models. Of particular interest is the pairwise MRF. The fixed points of the BP algorithm are the stationary points of the Bethe free energy. Two sorts of tree approximations, Bethe tree approximation and 4-star tree approximation, are introduced and exprimented on. We show that the BP algorithm dramatically improved the speed of skin probability estimation at pixel locations at similar pixel classification performance to that of Gibbs sampling.

Chapter 6: Blocking adult images

In this chapter we apply the skin detector to adult image detection. We extract several simple features from the skin probability images. We train a two-layer perceptron on these features.

The experiments on a large image database show stimulating performance despite the simplicity of our features based only on skin probability images. This model has been incorporated into the Poesia system, which is an open-source platform for Internet content scanning. The whole architecture of Poesia system is introduced in this chapter. We also show that combination of several images on the same web pages improved the overall performance of pornographic web page detection.

Chapter 7: Conclusion and perspectives

This chapter concludes the thesis and points out the furture direction of the related work.

Résumé (Chapitre 2)

Nous considerons dans ce chapitre trois modèles de maximum d'entropie respectant différentes contraintes sur les lois jointes des pixels voisins.

Le principe de maximum d'entropie consiste à trouver la loi de probabilité qui maximise l'entropie sous certaines contraintes statistiques. Ces contraintes qui sont exprimées sous forme d'espérance sont calculées à partir de la base d'apprentissage. Ce principe nous dit que la meilleur décision à prendre est celle la plus uniforme. Cette uniformité est calculée grâce à l'entropie.

Le premier modèle impose des contraintes sur un pixel. La solution obtenue est le modèle de base qui suppose l'indépendance des pixels de peau. Ce modèle ne tient pas compte de l'information de voisinage. C'est pour cela que nous introduisons deux autres modèles dit champs de Markov aléatoire.

A l'exception du premier modèle, les autres modèles nécessitent l'estimation des paramètres.

Chapter 2

From independent model to Markov random fields

We consider a sequence of three maximum entropy models with respect to various constraints concerning marginal distributions. The first model imposes constraints on one-pixel marginals. The solution is a baseline model in which pixels are considered independent. This model is well known by practitioners [JR02]. The baseline model is certainly too loose and does not take into account the fact that skin zones are not purely random but are made of large regions with regular shapes. Hence, in the second model, we add constraints on the marginals of neighboring labels, which will hopefully smooth the solution. Finally, the joint probabilities of neighboring colors and skin labels are included in building the third model. We expect that the increased knowledge about the statistics of the color images and skin label images would help discriminate skin pixels from non-skin ones.

2.1 Maximum entropy principle

The maximum entropy concept has a long history. Adopting the least complex hypothesis possible appears early in the Bible [BPP96]. Laplace might justly be considered the father of maximum entropy, having enunciated the underlying theme 200 years ago in his "Principle of Insufficient Reason": when one has no information to distinguish between the probability of two events, the best strategy is to consider them equally likely [GS85]. The principle of maximum entropy can thus be seen as a generalization of the "Principle of Insufficient Reason". This strategy was first proposed as a general inference procedure by Jaynes [Jay57a] [Jay57b].

Suppose we are about to estimate a distribution subject to some constraints, or say, some known facts about the statistics of the distribution. These constraints could be some statistics we observed from a lot of samples from the underlying distribution. That is, we constrain our solution to those distributions conforming with our observations. The solutions could be enormous if we do not have any prior information. Then, which distribution should we choose? Maximum entropy principle solves this problem by the policy of honesty, that is, frankly acknowledging the full extent of its ignorance by taking into account all possibilities allowed by the knowledge [Jay]. We would not rule out any possibilities except the constraints explictly tell us to do so. We would accept the solution that is in line with the constraints, and otherwise as uniform as possible. Under maximum entropy principle, the uniformity of a distribution is measured by the information entropy [Sha48]. For a discrete random variable X with distribution p, its entropy is defined as:

$$\mathcal{H}(X) = -\sum_{x\in\mathbf{X}} p(x)\ln p(x) \tag{2.1}$$

where $\mathbf{X}$ is the space of the random variable X. We will use the convention that $0\ln 0 = 0$, which is easily justified by continuity since $\lim_{x\to 0} x\ln x = 0$. Thus adding terms of zero probability does not change the entropy. Note that entropy is a function of the distribution of X. It does not depend on the actual values taken by the random variable X, but only on the probabilities. We shall denote the expectation by E. Thus if X has the distribution p, then the expected value of the random variable $f(X)$ is written as:

$$E_p[f(X)] = \sum_{x\in\mathbf{X}} p(x)f(x) \tag{2.2}$$

or simply as $E[f(X)]$ when the probability mass function is understood from the context. The entropy of X can also be interpreted as the expected value of $-\ln p(X)$, where X is drawn according to probability mass function $p(x)$. Thus

$$\mathcal{H}(X) = E_p[-\ln p(X)] \tag{2.3}$$

It is very easy to verify that the entropy of a random variable X is always non-negative, i.e., $\mathcal{H}(X) > 0$ always holds. Figure 2.1 shows the entropy curve $\mathcal{H}(X)$ of a binary random variable X with respect to its probability mass function $\Pr(X)$. X can take two discrete values 0 or 1. We can see that the entropy $\mathcal{H}(X)$ gets maximized when $\Pr(X = 0) = \Pr(X = 1) = 0.5$, and it starts to decrease when $\Pr(X = 0)$ and $\Pr(X = 1)$ get apart from each other. $\mathcal{H}(X)$ has the minimum value 0 when X assumes one value 0 or 1 with probability 1.

For an arbitrary discrete random variable X, we can verify that $\mathcal{H}(X)$ reaches its maximum when X is uniformly distributed. It follows naturally that under certain constraints, the distribution with the maximum entropy would also be the most uniform one among all distributions

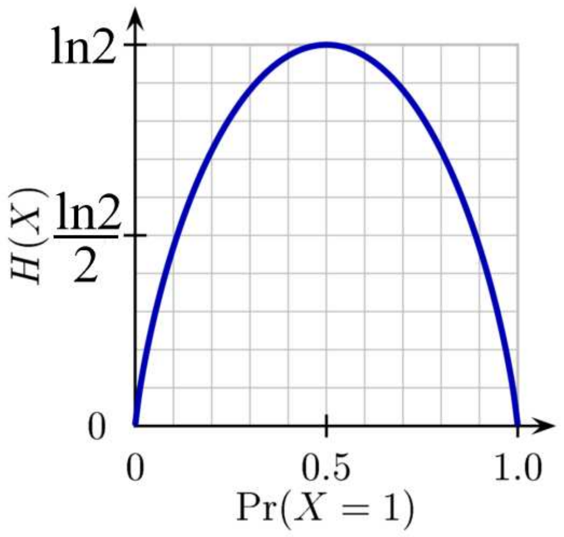


Figure 2.1: Entropy of a binary random variable X with respect to its probability mass function $\Pr(X)$. The horizontal axis is the probability that $X = 1$. The vertical axis is the entropy of X (The picture is originally from Wikepedia [Wikb] for entropy of base 2, and modified by the author with the natural base.)

subject to the defined constraints. Jaynes [Jay] justified the reason for choosing entropy as the measurement of uniformity.

[Jay] gives a full account of maximum entropy modeling (MaxEnt). The following introduction of MaxEnt in discrete case basically follows the idea in that classical text, with some efforts to make it brief and explore some different point of view. Suppose a random variable X can take on n different discrete values $(x_1, x_2, \cdots, x_n)$ with the distribution p. We measure a line of $m, m < n$, features $f_k(X), k = 1, \cdots, m$, from X. The expectation of the corresponding measured values of these features are $F_k(X), k = 1, \cdots, m$. The question is: what is the maximum entropy solution to p while these expections of features are satisfied? The question could be formulated as: maximizing the entropy $\mathcal{H}_p(X)$ subject to the following constraints,

$$\begin{cases} \sum_i p(x_i) &= 1 \\ E[f_k(X)] &= \sum_i p(x_i) f_k(x_i) = F_k(X) \end{cases} \tag{2.4}$$

We can introduce as many Lagrange multipliers as the constraints posed on the maximization problem

$$L(p) = \mathcal{H}_p(X) + (\lambda_0 - 1) \sum_i p(x_i) + \sum_k \lambda_k \left(\sum_i p(x_i) f_k(x_i) - F_k(X) \right) \tag{2.5}$$

Take the partial derivative, we get, for $i = 1, \cdots, n$,

$$\begin{aligned}\frac{\partial L(p)}{\partial p(x_i)} &= \frac{\partial \mathcal{H}_p(X)}{\partial p(x_i)} + (\lambda_0 - 1) + \sum_k \lambda_k f_k(x_i) \\ &= 0 \end{aligned} \tag{2.6}$$

The solution is then the following:

$$p(x_i) = \exp(\lambda_0 + \sum_{k=1}^{m} \lambda_k f_k(x_i)) \tag{2.7}$$

or equivalently:

$$p(x_i) = \frac{1}{Z}\exp(\sum_{k=1}^{m} \lambda_k f_k(x_i)) \tag{2.8}$$

if we define

$$Z = \exp(-\lambda_0) \tag{2.9}$$

Since p is a distribution, that is, $\sum_i p(x_i) = 1$, we can easily find

$$Z = \sum_i \exp(\sum_{k=1}^{m} \lambda_k f_k(x_i)) \tag{2.10}$$

Z is called *partition function* or *normalization function.*

The maximum entropy we get is then:

$$\mathcal{H}_{max}(X) = -\sum_i p(x_i) \ln p(x_i) = \ln Z - \sum_{k=1}^{m} \lambda_k F_k(X) \tag{2.11}$$

For now we have got a maximum entropy solution where the partial derivatives are defined. The question is, what we get is a global maximum or unfortunately only a local one? Furthermore, is it possible that we could find a higher entropy somewhere the partial derivatives are not defined, for example, at a cusp (discontinuity of slope)? The variational method alone we have just implemented does not answer these questions. In the following we will give a rigorous proof that the solution we just found does give the maximum entropy among all possible distributions that satisfy the constraints (2.4).

Before we prove (2.7) is a global maximum entropy solution, we shall introduce the concept of *Kullback-Leibler (KL) distance.* It is a measure of distance between two probability mass function $q(x)$ and $r(x)$. The definition is as follows:

$$D(q\|r) = \sum_{x\in\mathbf{X}} q(x) \ln \frac{q(x)}{r(x)} \tag{2.12}$$

Sometimes it is referred to as *KL divergence, cross entropy* or *relative entropy*. We can easily show that the KL distance is always nonnegative and is zero if and only if $q = r$. We know $\ln x \leq (x-1), x > 0$, with equality if and only if $x = 1$. Then $D(q\|r) = \sum_x q(x) \ln \frac{q(x)}{r(x)} = -\sum_x q(x) \ln \frac{r(x)}{q(x)} \geq -\sum_x q(x)(\frac{r(x)}{q(x)} - 1) = 0$, with equality if and only if $q(x) = r(x), \forall x \in \mathbf{X}$. Even though KL distance is not a true distance since it is not symetric and it does not satisfy the triangle inequality, it is still useful to consider it as a measure of "distance" between two distributions [CT91].

In order to prove (2.7) is a global maximum entropy solution subject to the constraints defined in (2.4) we need a lemma.

Lemma 1 *Supposing $r(x)$ is an arbitrary distribution subject to the constraints (2.4), p is the maximum entropy solution defined in (2.8) and subject to the constraints (2.4). Then $\mathcal{H}_p(X) - \mathcal{H}_r(X) = D(r\|p)$.*

Proof: Since $r(x)$ is inside (2.4), i.e., $\forall k = 1, \cdots, m$:

$$\sum_i r(x_i) f_k(x_i) = F_k(X)$$

From (2.11), we have:

$$\begin{aligned}\mathcal{H}_p(X) &= \ln Z - \sum_{k=1}^{m} \lambda_k F_k(X) = \sum_i r(x_i)(\ln Z - \sum_{k=1}^{m} \lambda_k f_k(x_i)) \\ &= -\sum_i r(x_i) \ln p(x_i)\end{aligned}$$

It follows from the definition of KL distance that $\mathcal{H}_p(X) - \mathcal{H}_r(X) = D(r\|p)$. Proof concluded.

Since we know the KL distance $D(r\|p)$ is non-negative with zero if and only if $r = p$, it follows naturally from Lemma 1 that among all the distributions r subject to constraints (2.4), the one $r = p$ has the maximum entropy defined in (2.11). This is a rigorous proof of our maximum entropy solution. It convinces us that as long as we could find a solution with the form (2.8) and conforming to the constraints (2.4) at the same time, it must have the maximum entropy among all distributions subject to these constraints. We now show that if the maximum entropy solution exists, then it must be unique. Suppose we have found one maximum entropy distributions p with the form (2.8) subject to the constraints (2.4), which has the maximum entropy $\mathcal{H}_p(X) = \mathcal{H}_{\max}$. Now a distribution p^* is given. We are told that p^* has also the same maximum entropy $\mathcal{H}_{p^*}(X) = \mathcal{H}_{\max}$ among all distributions subject to the constraints (2.4). Now we shall show that $p^* = p$ must hold. We learned from Lemma 1 that $D(p^*\|p) = \mathcal{H}_p(X) - \mathcal{H}_{p^*}(X) = 0$. This implies $p^* = p$. Proof concluded.

Maximum entropy formalism for the continuous case

Shannon's entropy though defined for a discrete random variable can be extended to the continuous case [Tan]. Now let X be a continuous random variable with pdf p, then its entropy can be defined by

$$\mathcal{H}(X) = -\int p(x)\ln p(x)dx \tag{2.13}$$

whenever it exists. It is also called *differential entropy* in literature. Unlike the entropy of a discrete random variable, the entropy of a continuous random variable may be infinitely large, negative or positive [Ash65]. For a continous random variable $X \in [a, b]$, the entropy is maximized when p is a uniform distribution [Mit].

The constraints for a continuous random variable X are defined as:

$$\begin{cases} \int_a^b p(x)dx & = & 1 \\ E[f_k(X)] & = & \int_a^b p(x)f_k(x)dx = F_k(X), \ \forall k = 1, \cdots, m. \end{cases} \tag{2.14}$$

As the discrete case, we find the solution

$$p(x) = \exp(\lambda_0 + \sum_{k=1}^{m} \lambda_k f_k(x)) \tag{2.15}$$

A complete description and derivation of these equations can be found in [Sel85].

The principle of maximum entropy is useful whenever our information is testable, i.e., we can determine whether or not a given distribution is consistent with the information at hand. In practice the maximum entropy modeling works as follows:

1. choose relevant features. We specialize in colors and skinness for skin detection;
2. compute their histograms on the training set;
3. write down the maximum entropy model within the ones that have the feature histograms as observed on the training set;
4. estimate the parameters of the model;
5. use the model for classification.

This plan has been successfully completed for several tasks related to speech recognition and language processing [BPP96]. When working with images, the graph underlying the model is the pixel lattice. It has many nodes and many loops. Task 4 is much more difficult. A break through appeared with the work in [ZWM98] on texture simulation where 1, 2, 3 and 4 were performed for images and 5 replaced by simulation.

2.2 Notations

In order to tackle skin detection with MaxEnt, we need to fix some notations for easier description and understanding of the solution procedure. The set of pixels of an image is S. The color of a pixel $s \in S$ is x_s. It is a 3 dimensional vector, each component being usually coded on one octet. $C = \{0, \ldots, 255\}^3$ denotes the RGB color space. The "skinness" of a pixel s, is y_s with $y_s = 1$ if s is a skin pixel and $y_s = 0$ if not. The color image, which is the vector of color pixels, is x and the binary image made up of the y_s's is denoted by y.

Let's assume for a moment that we knew the joint probability distribution $p(x, y)$ of the vector (x, y), then Bayesian analysis tells us that, whatever cost function the user might think of, all that is needed is the posterior distribution $p(y|x)$. From the user's point of view, the useful information is contained in the one pixel marginal of the posterior, that is, for each pixel, the quantity $p(Y_s = 1|x)$, quantifying the belief for skinness at pixel s given the full color image. Hence we explore a system where the output is a gray-scale image with the gray scales proportional to the skin probabilities at pixel locations.

In practice the model $p(x, y)$ is unknown. Instead, we have the Compaq Database. It is a collection of samples

$$\{(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})\}$$

where for each $1 \leq i \leq n$, $x^{(i)}$ is a color image and $y^{(i)}$ is the associated binary skinness image. We assume that the samples are independent of each other with distribution $p(x, y)$. The collection of samples is referred later as the training data. Probabilities are estimated by using classical empirical estimators and are denoted with the letter q.

In what follows, we build models for $p(x, y)$, the joint probability distribution of color and skinness images, using MaxEnt.

2.3 Baseline model

First, we build a model that respects the one pixel marginal observed in the Compaq Database. That is, consider the set of probability distributions $p(x, y)$ that verify:

$$\mathcal{C}_0 : \forall s \in S, \forall x_s \in C, \forall y_s \in \{0, 1\}, p(x_s, y_s) = q(x_s, y_s) \tag{2.16}$$

In (2.16), the quantity on the right side of the equal sign is the proportion of pixels with color x_s and label y_s in the training data. The MaxEnt solution under $\mathcal{C}_0$ is the independent model:

$$p(x, y) = \prod_{s \in S} q(x_s, y_s) \tag{2.17}$$

The proof is postponed to Appendix B. Using Bayes formulae, one then obtains:

$$p(y|x) = \prod_{s \in S} q(y_s|x_s) \tag{2.18}$$

We call the model in (2.18) the Baseline model. It is the most commonly used model in the literature [TDA98]. It is in fact a realization of the Bayesian modeling based on RGB histograms in [JR02], as we have introduced in the first chapter.

Each term of the product on the right side of (2.18) can be computed using probabilities estimated on the training data as follows using Bayes formula:

$$q(y_s|x_s) = \frac{1}{q(x_s)} q(x_s|y_s) q(y_s) \tag{2.19}$$

with

$$q(x_s) = \sum_{y_s=0}^{1} q(x_s|y_s) q(y_s)$$

Evaluation of the quantities in (2.19) is based on two 3 dimensional histograms, $q(x_s|y_s = 1)$ and $q(x_s|y_s = 0)$ describing the one pixel color distribution for skin and non-skin pixels respectively. Several authors have tried to get a parametric expression for these histograms as Gaussian distribution or mixtures of Gaussian distributions [JR02] [TSFA00]. Our experience is that the Compaq Database is large enough so that crude histograms made with 512 color value per bin uniformly distributed do not over-fit. Each histogram is then made of 32^3 bins.

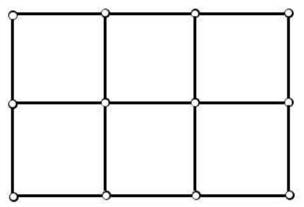
2.4 Markov random fields

The other two MaxEnt models are Markov random fields. Before proceeding to the specific models, we give a brief introduction to Markov random fields and related concepts. The introduction follows the classical book [Win03], with some notations adjusted for the convention of this thesis. Let S be a finite set of vertices/sites. A *neighborhood system* $\mathcal{V}$ is a collection of subsets of S indexed by a vertex s: $\mathcal{V} = \{\mathcal{V}_s \subset S, s \in S\}$ such that

a) $\forall s \in S, s \notin \mathcal{V}_s$

b) $\forall s, t \in S, s \in \mathcal{V}_t \Rightarrow t \in \mathcal{V}_s$

One can then check that $(S, \mathcal{V})$ is a non oriented graph. The vertices $t \in \mathcal{V}_s$ are called neighbors of s. We shall write $s \sim t$ instead of $t \in \mathcal{V}_s$ when convenient. For example, we can define the lattice $S = \{(i,j); 1 \leq i \leq n_1, 1 \leq j \leq n_2\}$, and $\mathcal{V}_{(i,j)} = \{(u,v) \in S; 0 \leq (u-i)^2 + (v-j)^2 \leq a\}$.


(a) 4-neighborhood system ($n_1 = 3, n_2 = 4, a = 1$)

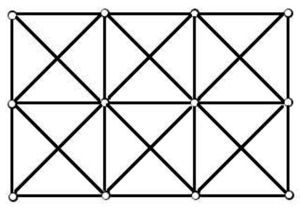
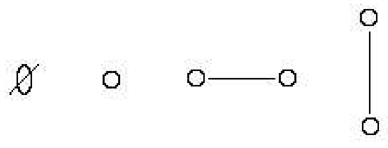

(b) 8-neighborhood system ($n_1 = 3, n_2 = 4, a = 2$)

Figure 2.2: Examples of neighborhood systems. $S = \{(i,j); 1 \leq i \leq n_1, 1 \leq j \leq n_2\}$, $\mathcal{V}_{(i,j)} = \{(u,v) \in S; 0 \leq (u-i)^2 + (v-j)^2 \leq a\}$, $\mathcal{V} = \{\mathcal{V}_{(i,j)}; (i,j) \in S\}$. Circles represent the sites. Links between circles represent their neighborhood relationship

Then if we let $\mathcal{V} = \{\mathcal{V}_{(i,j)}; (i,j) \in S\}$, $(S, \mathcal{V})$ is a neighborhood system. Figure 2.2 shows two such neighborhood systems corresponding to different a.

A *clique* $\mathfrak{c}$ is a subset of S such that: $s \sim t, \forall s,t \in \mathfrak{c}$. Figure 2.3 shows the cliques of the 4-neighborhood system and the 8-neighborhood system defined in Fig. 2.2. The set of cliques will be denoted by $\mathfrak{C}$.


(a) Cliques of 4-neighborhood system

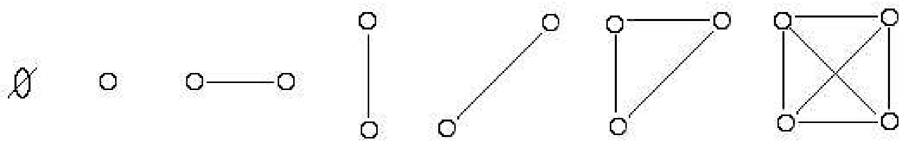

(b) Cliques of 8-neighborhood system

Figure 2.3: Examples of cliques corresponding to the neighborhood systems in Fig. 2.2. $\emptyset$ denotes the empty set

Let $X_s \in \mathbf{X}_s = \{1, \ldots, K\}$ be a random variable on the site $s \in S$. Then $X = (X_s)_{s \in S}$ is a stochastic process defined over the set S and taking a finite number of values. The space of *configurations* $x = (x_s)_{s \in S}$ is denoted by $\mathbf{X} = \{1, .., K\}^S$. So $X \in \mathbf{X}$. X is called a *random field* if $p(X = x) > 0$, $\forall x \in \mathbf{X}$. For $A \subseteq S$, let $x_A = (x_s)_{s \in A}$ denote a configuration on the subset A of sites and $\mathbf{X}_A = \prod_{s \in A} \mathbf{X}_s$ the space of all configurations on A. We shall write $A \backslash B = \{e \in A; e \notin B\}$ for two sets A and B. The following conditional probatilities

$$p(x_A | x_{S \backslash A}), x_A \in \mathbf{X}_A, x_{S \backslash A} \in \mathbf{X}_{S \backslash A} \tag{2.20}$$

are called *local characteristics*. It is the probability that the configuration on A is x_A given the configuration $x_{S \backslash A}$ on the rest of the world.

X is a *Markov Random Field* (MRF) with respect to $(S, \mathcal{V})$ if and only if $\forall x \in \mathbf{X}$, $\forall s \in S$

$$\begin{cases} p(X = x) & > & 0, \text{ and} \\ p(X_s = x_s | X_t = x_t; t \neq s) & = & p(X_s = x_s | X_t = x_t; t \in \mathcal{V}_s) \end{cases} \tag{2.21}$$

In words, (2.21) says that in order to guess the value of X at s, it is equivalent to know the values of the entire field except at s or to know the values at the neighboring locations of s. It could be extended to arbitrary subsets A of S as the following formula says:

$$p(x_A | x_{S \backslash A}) = p(x_A | x_{\mathcal{V}_A}), x_A \in \mathbf{X}_A, x_{S \backslash A} \in \mathbf{X}_{S \backslash A} \tag{2.22}$$

where $\mathcal{V}_A = (\bigcup_{s \in A} \mathcal{V}_s) \backslash A$ is called the *neighbors of* A.

Remark that any random process X such that $p(X = x) > 0, \forall x \in \mathbf{X}$, is a MRF with respect to $(S, \mathcal{V})$ if we define $\mathcal{V}_s = S \backslash \{s\}$.

Of particular interest are the so-called pairwise MRFs. These are the simplest and most used MRFs in computer vision. Their distribution is given by

$$\pi(x) = p(X_s = x_s; s \in S) = \frac{1}{Z} \prod_{s \sim t} \psi_{st}(x_s, x_t) \prod_{s \in S} \phi_s(x_s) \tag{2.23}$$

where the product over $s \sim t$ means the product over all the couples of mutual neighbor vertices and Z is a normalizing constant. We can verify that any random field with the form (2.23) satisfies the MRF definition (2.21). In the model defined by (2.23), $\phi_s(x_s)$ is often called "evidence" for x_s, $\psi_{st}(x_s, x_t)$ is the "compatibility" function of x_s and x_t.

2.5 Hidden Markov model

The Baseline model is certainly too loose and one might hope to get better detection results by constraining it to a model that takes into account the fact that skin zones are not purely random but are made of large regions with regular shapes. Hence, we fix the marginals of y for all the neighboring pixel couples. We use 4-neighbor system for simplicity in all that follows. For 2 neighboring pixels s and t, the expected proportion of times that we observe $(y_s = a, y_t = b)$ should be $q(a, b)$ for $a = 0, 1$ and $b = 0, 1$, the corresponding quantities measured on the training set. We assume that the model is isotropic, aggregating the cases where s and t are in vertical position to the cases where s and t are in horizontal position. Hence let us define the following constraints:

$$\mathcal{D} : \forall s \sim t \in S \times S, \quad \begin{aligned} &p(y_s = 0, y_t = 0) = q(0, 0) \text{ and} \\ &p(y_s = 1, y_t = 1) = q(1, 1) \end{aligned} \tag{2.24}$$

The MaxEnt model under $\mathcal{C}_0 \cap \mathcal{D}$ is then the following Gibbs distribution:

$$p(x, y) \propto \prod_{s \in S} q(x_s|y_s) \exp\left(\sum_{s \sim t} (a_0(1 - y_s)(1 - y_t) + a_1 y_s y_t)\right) \tag{2.25}$$

Here and thereafter, the sign $\propto$ means equality up to a normalization function. a_0 and a_1 are constants that must be set up such that the constraints are satisfied. The proof is in Appendix B. From (2.25) one then obtains the following model:

$$p(y|x) \propto p(y) \prod_{s \in S} q(x_s|y_s) \tag{2.26}$$

with

$$p(y) = \frac{1}{Z(a_0, a_1)} \exp\left(\sum_{s \sim t} (a_0(1 - y_s)(1 - y_t) + a_1 y_s y_t)\right) \tag{2.27}$$

where $Z(a_0, a_1)$ is the normalization function. It is defined by the following formula:

$$Z(a_0, a_1) = \sum_y \left(\exp\left(\sum_{s \sim t} (a_0(1 - y_s)(1 - y_t) + a_1 y_s y_t)\right)\right) \tag{2.28}$$

The model in equation (2.27) is known as a special case of a Potts model, see [Win95] and [CJ83]. We can verify that $p(x|y) = \prod_{s \in S} q(x_s|y_s) = \prod_{s \in S} p(x_s|y_s)$, so it is a HMM if we consider y to be the hidden layer.

2.6 First-order model

The Baseline model was built in order to mimic the one pixel marginal of the joint distribution of color and skinness as observed on the database. Then, in building the HMM we added constraints on the prior skinness distribution in order to smooth the model. Now, we constrain once more the MaxEnt model by imposing the two-pixel marginal, that is $p(x_s, x_t, y_s, y_t)$, for 4-neighbor $s \sim t$, to match those observed in the training data. Hence we define the following constraints:

$$\begin{aligned}\mathcal{C}_1 :\quad & \forall s \sim t \in S \times S, \forall x_s \in C, \forall x_t \in C, \forall y_s \in \{0,1\}, \forall y_t \in \{0,1\}, \\ & p(x_s, x_t, y_s, y_t) = q(x_s, x_t, y_s, y_t)\end{aligned} \tag{2.29}$$

The quantity $q(x_s, x_t, y_s, y_t)$ is the expected proportion of times we observe the values (x_s, x_t, y_s, y_t) for a couple of neighboring pixels, regardless of the orientation of the pixels s and t in the training set.

Clearly, $\mathcal{C}_1 \subset (\mathcal{C}_0 \cap \mathcal{D}) \subset \mathcal{C}_0$. The solution to the MaxEnt problem under $\mathcal{C}_1$ is then, see Appendix B, the following Gibbs distribution:

$$p(x, y) \propto \exp\left(\sum_{s\sim t} \lambda(s, t, x_s, x_t, y_s, y_t)\right) \tag{2.30}$$

where $\lambda(s, t, x_s, x_t, y_s, y_t)$ are parameters that should be set up to satisfy the constraints. From (2.30), one gets

$$p(y|x) \propto \exp\left(\sum_{s\sim t} \lambda(s, t, x_s, x_t, y_s, y_t)\right) \tag{2.31}$$

It contains a huge number of parameters for ordinary-sized images.

2.7 Discussion

This chapter introduced the principle of maximum entropy. It is a fundamental philosophy for inferring the probabilistic model according to some observed statistics of the samples, which are the so-called constraints. The maximum entropy principle states that the best decision is to accept the model in line with the constraints and otherwise as uniform as possible. And the uniformity of the model is measured by the information entropy.

With the guidance of the maximum entropy principle, we built a sequence of three statistical models for the distributions of the skin label images conditioned on given color images. The first model, called the Baseline model in this thesis, has been well established in the literature through other means. However we showed that under certain assumption (constraints), we can derive this model through the maximum entropy framework. It justifies the power of this fundamental principle, and gives more insight into the well-known simplest statistical skin model.

The Baseline model is an independent model that does not take into account the relationship between neighboring pixels. The observation of the image database tells us that the skin pixels are not totally independent. Instead, they tend to appear together and form some regions. A natural step would be to model the distribution with MRFs. Again we get to MRFs naturally through the maximum entropy principle. That is, we have a good reason to choose the Markov random fields—it is the natural solution under certain constraints according to the maximum entropy principle. Thus, the independent model as well as the Markov random fields models are all unified into the maximum entropy framework. Our first MRF model is a hidden Markov model, which puts constraints on the joint probabilities of the pairs of neighboring skin labels besides the constraints introduced in the Baseline model. In the second MRF model, we go further to include the joint probabilities of the pairs of neighboring colors and skin labels. This model is called the first-order model in this thesis.

Except the Baseline model, which can be solved analytically, all the models that we built are in parametric forms. We shall tackle the issue of parameter estimation in the next chapter.

Résumé (Chapitre 3)

L'estimation des paramètres dans le context de MaxEnt est un problème difficile, spécialement quand l'estimateur de maximum de vraisemblance ne peut pas être calculé.

Différentes méthodes ont été proposées [Bes86] [Zha92] [WD95] [You98]. Dans ce chapitre nous proposons deux méthodes pour l'estimation de ces paramètres. La première exploite le concept de "Julesz ensembles" permettant de simuler la texture de la peau. La seconde méthode approxime localement le graphe de pixel par des arbres. Cette approximation nous fournit une solution analytique à MaxEnt sans paramètres à estimer.

Nous avons pu montrer que les paramètres estimés par les deux méthodes sont sensiblement identiques pour un modèle simple. Pour un modèle plus complexe, avec beaucoup plus de paramètres, l'approximation à l'aide des arbres reste viable alors que la méthode alternative ne l'est plus.

Quel type d'arbre choisir ? nous présentons les arbres de Bethe ainsi que les arbres "étoilés".

Chapter 3

Parameter estimation

Parameter estimation in the context of MaxEnt is still an active research subject, especially in situations where even the likelihood function cannot be computed for a given value of the parameters. This is the case here since the partition function cannot be evaluated even for very small size images. One line of research consists in approximating the model in order to obtain a formula where the partition function no longer appears: Pseudo-likelihood [Bes86] [DF00], mean field methods [Zha92] [CFP02], as well as Bethe tree models [WD95] are among them. Another possibility is to use stochastic gradient as in [You98].

3.1 Parameter estimation through local characteristics

Here we explore a related method based on the concept of Julesz ensembles defined in [WZL00]. We learn from this work that one can sample an image from the model defined in (2.27) without knowing the parameters a_0 and a_1. For large lattices, the model defined in (2.27) will concentrate on the set of images y's that respects the constraints $p_y(0,0) = q(0,0)$ and $p_y(1,1) = q(1,1)$, where

$$\begin{cases} p_y(0,0) & = & N_y(0,0)/N_y(.,.) \\ p_y(1,1) & = & N_y(1,1)/N_y(.,.) \end{cases} \tag{3.1}$$

with $N_y(.,.)$ being the number of cliques in the label image y, $N_y(0,0)$ being the number of cliques having labels $(0,0)$, and $N_y(1,1)$ being the number of cliques having labels $(1,1)$. $p_y(0,0)$ and $p_y(1,1)$ are in fact the pair-wise marginal probabilities of the samples y. The above principle is saying that for large lattices, a special set of images will receive most of the probabilities, in which each image respects the constraints itself. For the infinite lattice, this set of images will eventually receive the probability 1. Each image in the same set will have the same probabilities.

So if we sample on the infinite lattice from the model defined in (2.27) that respect the constraints $\mathcal{D}$, we will always get the set of images each of which respects the constraints $\mathcal{D}$ itself. Even though this is true only in the asymptotic case of an infinite image but we will apply the result for a large image, say 512×512 pixels. In a second step, we use this sample image in order to estimate the parameters a_0 and a_1. This is done using the quantity $p(Y_s = 1|y_{(s)})$ which is the probability to observe the label 1 at pixel s given all the other values y_t, $\forall t \in S$ and $t \neq s$. For the model in (2.27), this quantity can be easily analytically computed as

$$p(Y_s = 1|y_{(s)}) = \phi((a_1 + a_0)n_s(1) - 4a_0) \tag{3.2}$$

where $\phi(x) = (1+e^{-x})^{-1}$ is the sigmoïd (also known as logistic) function and $n_s(1)$ is the number of neighbors of s that take the label 1. The proof is as follows. Recall the formula in (2.27). Now we have:

$$\begin{aligned} p(Y_s = 1|y_{(s)}) &= \frac{p(Y_s = 1, y_{(s)})}{\sum_{y_s} p(y_s, y_{(s)})} \\ &= \frac{\exp\left(\sum_{t\in\mathcal{V}_s} a_1 y_t\right)}{\exp\left(\sum_{t\in\mathcal{V}_s} a_1 y_t\right) + \exp\left(\sum_{t\in\mathcal{V}_s} a_0(1-y_t)\right)} \\ &= \frac{1}{1+\exp\left(\sum_{t\in\mathcal{V}_s}(a_0 - (a_1+a_0)y_t)\right)} \\ &= \frac{1}{1+\exp\left(4a_0 - (a_1+a_0)n_s(1)\right)} \\ &= \phi((a_1+a_0)n_s(1) - 4a_0) \end{aligned}$$

This sum can take only five different values since it is a function of $n_s(1)$ with $n_s(1) = 0, 1, 2, 3, 4$. For each possible value of $n_s(1)$, one quantity $p(Y_s = 1|y_{(s)})$ can be estimated from the sample image. This will lead to five linearly independent equations from which parameters a_0 and a_1 can be estimated. Now, returning to how to obtain a sample from the model in (2.27). The key idea which originated in statistical physics [ML79], is that the MaxEnt model which we are looking for is, in an appropriate asymptotic meaning, the uniform distribution over the set of images that respect the constraints $\mathcal{D}$. Now, in the absence of phase transition, sampling from this set can be achieved numerically using simulated annealing, see [GG84].

To sample from the uniform distribution of the set of images that verify the constraints $p_y(0,0) = q(0,0)$ and $p_y(1,1) = q(1,1)$, we construct the following energy function:

$$\begin{aligned} H(y) &= \left(\frac{1}{N_y(.,.)}\sum_{s\sim t}(1-y_s)(1-y_t) - q(0,0)\right)^2 \\ &\quad + \left(\frac{1}{N_y(.,.)}\sum_{s\sim t} y_s y_t - q(1,1)\right)^2 \end{aligned}$$

Remark that

$$\begin{cases} \frac{1}{N_y(.,.)} \sum_{s\sim t}(1-y_s)(1-y_t) & = & p_y(0,0) \\ \frac{1}{N_y(.,.)} \sum_{s\sim t} y_s y_t & = & p_y(1,1) \end{cases} \tag{3.3}$$

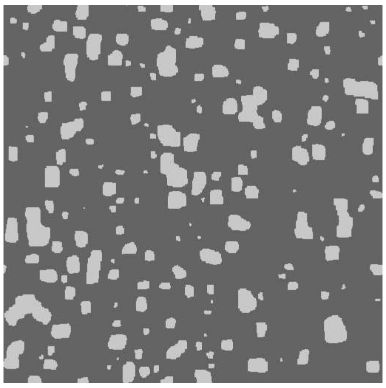
$H(y)$ is always non-negative. And when $H(y)$ gets minimized, that is, when $H(y) = 0$, we will have $p_y(0,0) = q(0,0)$ and $p_y(1,1) = q(1,1)$. We can apply the Metropolis algorithm [Win03] in order to sample from the associated Gibbs distribution, and to minimize the energy. It produces a sequence of images $y^{(0)}, y^{(1)}, y^{(2)}, \ldots$. At step n, we select $y^{(n)}$ as follows. Consider y', the image obtained by fliping the value at s of $y^{(n-1)}$

- If $H(y') \leq H(y^{(n-1)})$ then $y^n \leftarrow y'$
- If $H(y') > H(y^{(n-1)})$ then
  - * $y^{(n)} \leftarrow y'$ with probability
$$\exp(\beta(H(y) - H(y')))$$
  - * $y^{(n)} \leftarrow y^{(n-1)}$ with probability
$$1 - \exp(\beta(H(y) - H(y')))$$

If all the pixels are visited a sufficient number of times and if β increases logarithmically towards infinity (temperature decreases towards zero), then the sequence $y^{(n)}$ converges to a sample from the uniform distribution over the images that verify the constraints in (2.24), or say, over the images with the minimal energy.

Figure 3.1 shows a 512×512 sample of the prior model defined in equation (2.27). The quantities $Pr(Y_s = y_s, Y_t = y_t)$, for neighboring pixels s and t are presented in the array below, first, as estimated from the training data, and secondly, as estimated from the image in figure 3.1.

We see that the constraints are nearly respected. Parameter estimation from this image leads to the numerical values: $a0 = 3.76$ and $a1 = 3.94$. This image is a sample from the prior, hence one can qualitatively appreciate how well it models skin regions. Notice that vertical and horizontal borders are preferred. This is a bias of the neighborhood system. Choosing 8 neighbors could improve it at the expense of computational load.



	database values	image values
$Pr(Y_s = 0, Y_t = 0)$	0.828	0.827991
$Pr(Y_s = 1, Y_t = 1)$	0.159	0.151646

Figure 3.1: **Top:** a sample image from the prior distribution used in the Hidden Markov Model. **Bottom:** probabilities estimated from the training set and from the image on the top.

3.2 Parameter estimation through tree approximations

The above-mentioned method works for the HMM, where there are only two parameters to estimate. For the FOM in (2.31), however, there are enormous parameters, and it is impractical to build that huge equation system then solve it. Remark that all the models we built with MaxEnt are MRFs. We learned from [Pea88] that any MRF on tree graphs can be expressed analytically in terms of the marginal probabilities on single sites and pairs of sites. If we can approximate the pixel lattice with some trees, then we can solve the parameter estimation issue through the well-known results about MRFs on tree graphs. In what follows we begin with the introduction of MRFs on tree graphs.

3.2.1 Markov random fields on tree graphs

In the very special case where the underlying graph is loop free, for example a tree, then the functions (also called potentials) ψ_{st} and ϕ_s in (2.23) can be expressed as simple functions of the marginals of $\pi(.)$ and moreover when expressed in such a way, the normalizing constant Z equals to one [Pea88]. Specifically,

$$\pi(x) = \prod_{s \sim t} \frac{p_{st}(x_s, x_t)}{p_s(x_s)p_t(x_t)} \prod_{s \in S} p_s(x_s) \tag{3.4}$$

where $p_s(x_s)$ is the marginal distribution of $\pi(.)$ for vertex s and $p_{st}(x_s, x_t)$ is the marginal distribution of $\pi(.)$ for neighbor vertices s and t. That is

$$p_s(x_s) = \sum_{x_t; t\in S; t\neq s} \pi(x) \tag{3.5}$$

$$p_{st}(x_s, x_t) = \sum_{x_u; u\in S; u\neq s,t} \pi(x) \tag{3.6}$$

$$1 = \sum_{x_u; u\in S} \pi(x) \tag{3.7}$$

Proof: (3.4) is true for $S = \{s\}$. And we can easily check that:

$$\sum_x \pi(x) = \sum_{x_s} p_s(x_s) = 1$$

Assume (3.4) and $\sum_x \pi(x) = 1$ are true for any tree graph $(S, \mathcal{V})$ where $|S| = n$, $n \geq 1$. Now suppose we are given a tree graph $(S', \mathcal{V}')$ in which $|S'| = n + 1$. We select any leaf v from the graph, the rest of the graph $(S, \mathcal{V})$ where $S = S' \backslash \{v\}$ is a tree graph of n nodes. We denote as u the node connected with v. Then we have

$$\begin{aligned}
\pi(x) &= \pi(x_S, x_v) = \pi(x_v|x_S)\pi(x_S) = \pi(x_v|x_u)\pi(x_S) \\
&= \frac{p_{uv}(x_u, x_v)}{p_u(x_u)} \prod_{s\sim t, s,t\in S} \frac{p_{st}(x_s, x_t)}{p_s(x_s)p_t(x_t)} \prod_{s\in S} p_s(x_s) \\
&= \prod_{s\sim t, s,t\in S'} \frac{p_{st}(x_s, x_t)}{p_s(x_s)p_t(x_t)} \prod_{s\in S'} p_s(x_s)
\end{aligned}$$

Moreover,

$$\begin{aligned}
\sum_x \pi(x) &= \sum_{x_S} \left(\pi(x_S) \sum_{x_v} \frac{p_{uv}(x_u, x_v)}{p_u(x_u)} \right) \\
&= \sum_{x_S} \pi(x_S) = 1
\end{aligned}$$

So (3.4) and $\sum_x \pi(x) = 1$ are true for any tree graph $(S, \mathcal{V})$ where $|S| \geq 1$. Proof concluded.

Since $\prod_{s\sim t} p_s(x_s)p_t(x_t) = \prod_{s\in S} p_s^{n_s}(x_s)$, where n_s is the number of neighbors of site s, we can put the formula (3.4) in another way:

$$\pi(x) = \prod_{s\sim t} p_{st}(x_s, x_t) \prod_{s\in S} p_s^{1-n_s}(x_s) \tag{3.8}$$

3.2.2 Maximum entropy models on tree graphs

The FOM defined in (2.31) is a Markov random field on the non-oriented pixel graph. Let us assume for now that this graph was a tree: that is a connected graph without loops. Then, the Maxent solution under $\mathcal{C}_1$ would be

$$p(x,y) = \prod_{s\sim t} \frac{q(x_s,x_t,y_s,y_t)}{q(x_s,y_s)q(x_t,y_t)} \prod_{s\in S} q(x_s,y_s) \tag{3.9}$$

The proof is as follows: we have proved in the last section that any pairwise MRF on a tree graph can be written

$$p(z) = \prod_{s\sim t} \frac{p(z_s,z_t)}{p(z_s)p(z_t)} \prod_{s\in S} p(z_s) \tag{3.10}$$

where $p(z_s)$ is the one-site marginal of p and $p(z_s,z_t)$ is it's two-site marginal.

Applying this result to $z=(x,y)$ and replacing p with q on the right side permits to obtain the model in equation (3.9). By construction it is in $\mathcal{C}_1$. Moreover it has the same form as the one in equation (2.31) which concludes the proof.

The conditional model given the color image x is then:

$$p(y|x) \propto \prod_{s\sim t} \frac{q(x_s,x_t,y_s,y_t)}{q(x_s,y_s)q(x_t,y_t)} \prod_{s\in S} q(x_s,y_s) \tag{3.11}$$

It is called *tree first-order model* in this thesis and denoted by TFOM.

Assume conditional independence, that is

$$q(x_s,x_t|y_s,y_t) = q(x_s|y_s)q(x_t|y_t) \tag{3.12}$$

The obtained model is then an HMM model, as in equation (2.26). But now we can write $p(y)$ explicitly as follows:

$$p(y) = \prod_{s\sim t} \frac{q(y_s,y_t)}{q(y_s)q(y_t)} \prod_{s\in S} q(y_s) = \prod_{s\sim t} q(y_s,y_t) \prod_{s\in S} q^{1-n_s}(y_s) \tag{3.13}$$

The conditional model of y given x is then:

$$p(y|x) \propto \prod_{s\in S} q(x_s|y_s)p(y) \tag{3.14}$$

It is called *tree hidden Markov model* in this thesis and denoted by THMM.

Hence, tree approximation gives another way to estimate parameters a_0 and a_1. For the interior pixels, which have exactly 4 neighbors, we can calculate the local characteristics $p(Y_s = 1|y_{(s)})$ according to the distribution (3.13). The computed quantity is:

$$p(Y_s=1|y_{(s)}) = \phi\left(\left(\ln\frac{q(1,1)}{q(0,1)} - \ln\frac{q(1,0)}{q(0,0)}\right)n_s(1) + 4\ln\frac{q(1,0)}{q(0,0)} - 3\ln\frac{q(1)}{q(0)}\right) \tag{3.15}$$

Since this must be true for all the values of $n_s(1)$, comparing it with (3.2), we must have

$$\begin{cases} -4a_0 & = & 4\ln\frac{q(1,0)}{q(0,0)} - 3\ln\frac{q(1)}{q(0)} \\ a_1 + a_0 & = & \ln\frac{q(1,1)}{q(0,1)} - \ln\frac{q(1,0)}{q(0,0)} \end{cases} \tag{3.16}$$

Obtained values are $a_0 = 3.94$ and $a_1 = 4$, which are close to the values obtained in Section 3.1.

Bethe tree

Bethe tree has been introduced in computer vision as a way of approximating estimators in Markov random field models in [WD95]. We shall revisit this work in connection with maximum entropy models. The key idea is to provide a tree that approximates locally the pixel lattice. More precisely, for each pixel s, we consider a sequence of trees $\mathcal{T}_1^{(s)}, \mathcal{T}_2^{(s)}, \ldots$ of increasing depth. The construction is as follows: the root node of the tree is associated with s. For each neighbor t of s in the pixel-graph, a child node indexed by t is added to the root node. This defines $\mathcal{T}_1^{(s)}$. Subsequently, for each u, neighbor of a neighbor of s, (excluding s itself), a grandchild node indexed by u is added to the appropriate child node. This defines $\mathcal{T}_2^{(s)}$, and so on, see [WD95] for a detailed account. An important remark is that a single pixel might lead to several different nodes in the tree! For example $\mathcal{T}_2^{(s)}$ is built with s, the neighbors of s and the neighbors of these. Using 4-neighbors, and assuming that s is not in the border of the image, this makes up 13 pixels, but the associated tree has 17 nodes, 4 pixels being replicated twice each, see figure 3.2. If the original graph is acyclic, then the Bethe tree representation is exactly the same graph. Otherwise it is only an approximation.

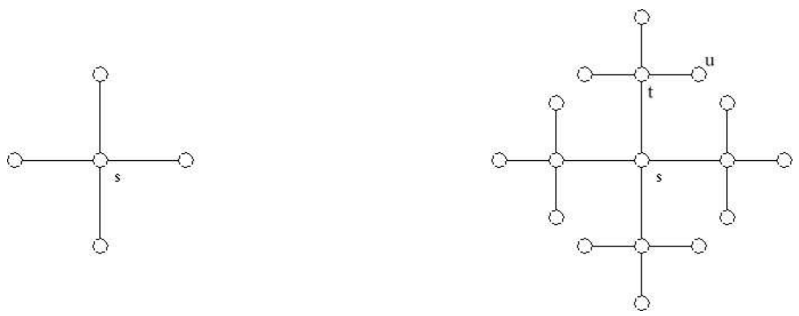


Figure 3.2: **Left:** a Bethe tree of depth 1 rooted at s. **Right:** a Bethe tree of depth 2 rooted at s, where t is an example of neighbors of s and u is an example of neighbors of t (excluding s itself)

4-Star tree

The 4-star tree of depth 1 rooted at s, $\mathcal{T}_1^s$, is built by adding to s its 4 neighbors in the pixel lattice. For the 4-star tree with one more depth, we add one node u for each leaf t of $\mathcal{T}_1^s$, following the direction $s \to t$. We repeat this process until the tree reaches the depth needed. Figure 3.3 shows the construction of 4-star trees.

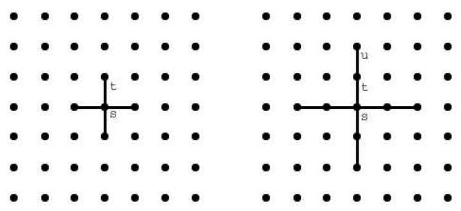


Figure 3.3: **Left:** a 4-star tree of depth 1. **right:** a 4-star tree of depth 2.

3.3 Implementing the tree first-order model

The quantities $q(x_s, x_t, y_s, y_t)$ on the right side of (3.9) cannot be directly extracted from the database without drastic over-fitting since the histograms involved have a support of high dimension. So some kind of dimension reduction is necessary. We have derived two methods to reduce the dimensions of the histograms as we will see in the following.

3.3.1 Gradient independence assumption

A promising dimension reduction procedure is the following approximation:

$$\begin{aligned} q(x_s, x_t, y_s, y_t) &= q(x_s, x_t|y_s, y_t)q(y_s, y_t) \\ q(x_s, x_t|y_s, y_t) &\approx q(x_s|y_s)q(x_t - x_s|y_s, y_t) \end{aligned} \tag{3.17}$$

That is, we assume that the color gradient at s, measured by the quantity $x_t - x_s$, is, given the labels at s and t, independent of the actual color x_s. Evaluation of the right side of the sign $\approx$ requires to compute 6 histograms with a support of dimension 3 only. We use 32^3 bins of 512 colors each. The MaxEnt model based on the approximation (3.17) shall be referred to as TFOM1 later on.

3.3.2 Maximum entropy estimation

Another possibility is to estimate $q(x_s, x_t, y_s, y_t)$ with MaxEnt under the constraints of one-site marginal and gradient distribution. This can be done off-line and will not introduce any extra online load to skin detection. To be more precise, denoting by $p_q(x_s, x_t, y_s, y_t)$ the two-site joint distribution that we would like to estimate, we define the following constraints:

$$\mathcal{C}^* : \forall x_s \in C, \forall x_t \in C, \forall y_s \in \{0,1\}, \forall y_t \in \{0,1\},$$

$$\begin{cases} p_q(x_s, y_s) & = & q(x_s, y_s) \\ p_q(x_t, y_t) & = & q(x_t, y_t) \\ p_q(x_t - x_s, y_s, y_t) & = & q(x_t - x_s, y_s, y_t) \end{cases} \tag{3.18}$$

The entropy of the joint distribution is defined as:

$$\mathcal{H}(p_q) = -\sum_{x_s, x_t, y_s, y_t} p_q(x_s, x_t, y_s, y_t) \ln p_q(x_s, x_t, y_s, y_t)$$

Our goal is to find the MaxEnt solution

$$p_q^*(x_s, x_t, y_s, y_t) = \text{argmax}_{p_q} \mathcal{H}(p_q)$$

subject to the constraints $\mathcal{C}^*$. Using Lagrange multipliers, then the MaxEnt solution is

$$p_q^*(x_s, x_t, y_s, y_t) = \mathcal{P}_\lambda \cap \mathcal{C}^*$$

in which

$$\mathcal{P}_\lambda = \{\frac{1}{Z_\lambda} \exp\left(\lambda(x_s, y_s) + \lambda(x_t, y_t) + \lambda(x_t - x_s, y_s, y_t)\right)\} \tag{3.19}$$

where

$$Z_\lambda = \sum_{x_s, x_t, y_s, y_t} \exp\left(\lambda(x_s, y_s) + \lambda(x_t, y_t) + \lambda(x_t - x_s, y_s, y_t)\right)$$

is a normalizing constant.

The MaxEnt model $p_q^*(x_s, x_t, y_s, y_t)$ is equivalent to a model $p_\lambda^*(x_s, x_t, y_s, y_t) \in \mathcal{P}_\lambda$ that maximizes the likelihood of the training data [BPP96]. However, the parameters λ^* that maximize the likelihood cannot be found analytically. Instead, we have to resort to numerical methods. From the perspective of numerical optimization, the likelihood function is well behaved since it is smooth and convex-$\cap$ in λ [BPP96]. An optimization method specifically tailored to the maximum entropy problem is the generalized iterative scaling algorithm of Darroch and Ratcliff [DR72]. See Appendix C for details of this algorithm. In the algorithm, we first initialize the parameters λ's. Then the iterative scaling process follows. For each iteration that updates the parameters, the likelihood function will climb a little towards the maximum value. We stop the iterative scaling when such updating will introduce very little gain in the likelihood.

The generalized iterative scaling algorithm for our solution of λ^* can be summarized as follows:

1. Initialize λ

$$\forall x_s \in C, \forall x_t \in C, \forall y_s \in \{0,1\}, \forall y_t \in \{0,1\},$$

$$\lambda_0(x_s, y_s) = \lambda_0(x_t, y_t) = \lambda_0(x_t - x_s, y_s, y_t) = 0$$

2. Update λ

(1) Calculate $Z_\lambda = \sum_{x_s,x_t,y_s,y_t} \exp\left(\lambda(x_s, y_s) + \lambda(x_t, y_t) + \lambda(x_t - x_s, y_s, y_t)\right)$

(2) Calculate all $\Delta\lambda$

$$\begin{aligned} \Delta\lambda(x_s, y_s) &= \frac{1}{3} \ln \frac{q(x_s, y_s)}{p(x_s, y_s)} \\ \Delta\lambda(x_t, y_t) &= \frac{1}{3} \ln \frac{q(x_t, y_t)}{p(x_t, y_t)} \\ \Delta\lambda(x_t - x_s, y_s, y_t) &= \frac{1}{3} \ln \frac{q(x_t - x_s, y_s, y_t)}{p(x_t - x_s, y_s, y_t)} \end{aligned}$$

where,

$$\begin{aligned} p(x_s, y_s) &= \sum_{x_t, y_t} \left(\frac{1}{Z_\lambda} \exp\left(\lambda(x_s, y_s) + \lambda(x_t, y_t) + \lambda(x_t - x_s, y_s, y_t)\right) \right) \\ p(x_t, y_t) &= \sum_{x_s, y_s} \left(\frac{1}{Z_\lambda} \exp\left(\lambda(x_s, y_s) + \lambda(x_t, y_t) + \lambda(x_t - x_s, y_s, y_t)\right) \right) \\ p(x_t - x_s, y_s, y_t) &= \sum_{x'_s, x'_t, x'_t - x'_s = x_t - x_s} \left(\frac{1}{Z_\lambda} \exp\left(\lambda(x'_s, y_s) + \lambda(x'_t, y_t) + \lambda(x'_t - x'_s, y_s, y_t)\right) \right) \end{aligned}$$

(3) Update λ according to: $\lambda \Leftarrow \lambda + \Delta\lambda$

3. Goto 2. if λ have not converged.

This algorithm can still be accelarated since there are some duplications of calculation. $\exp\left(\lambda(x_s, y_s) + \lambda(x_t, y_t) + \lambda(x_t - x_s, y_s, y_t)\right)$ are needed by Z_λ, $\Delta\lambda(x_s, y_s)$, $\Delta\lambda(x_t, y_t)$ and $\Delta\lambda(x_t - x_s, y_s, y_t)$. In the calculation of $\Delta\lambda(x_t - x_s, y_s, y_t)$, the summation ranges over x'_s, x'_t that have the required gradients $x'_t - x'_s = x_t - x_s$, this implies another subscript searching process. Supposing (x_s, x_t, y_s, y_t) can take n values, then Z_λ, $\{\Delta\lambda(x_s, y_s)\}$, $\{\Delta\lambda(x_t, y_t)\}$ and $\{\Delta\lambda(x_t - x_s, y_s, y_t)\}$ need about n exponential operations respectively, which does not take into account the computational load for the searching process in the calculation of $\Delta\lambda(x_t - x_s, y_s, y_t)$ yet. The total number of exponential operations involved for each iteration is about $4n$. We can avoid the replication of the exponential operations by parallel computation of the marginal array $p(x_s, y_s), p(x_t, y_t)$ and $p(x_t - x_s, y_s, y_t)$ and the normalizing constant Z_λ. Here is the accelerated algorithm:

1. Initialize λ

$$\forall x_s \in C, \forall x_t \in C, \forall y_s \in \{0,1\}, \forall y_t \in \{0,1\},$$

$$\lambda_0(x_s, y_s) = \lambda_0(x_t, y_t) = \lambda_0(x_t - x_s, y_s, y_t) = 0.0$$

2. Update λ

(1) Calculate marginals and Z_λ

(a) Initialize marginals and Z_λ to 0.

(b) For each value (x_s, x_t, y_s, y_t), we calculate

$$g(x_s, x_t, y_s, y_t) = \exp\left(\lambda(x_s, y_s) + \lambda(x_t, y_t) + \lambda(x_t - x_s, y_s, y_t)\right)$$

once and update the normalizing constant and marginal arrays,

$$Z_\lambda \Leftarrow Z_\lambda + g(x_s, x_t, y_s, y_t)$$

$$p(x_s, y_s) \Leftarrow p(x_s, y_s) + g(x_s, x_t, y_s, y_t)$$

$$p(x_t, y_t) \Leftarrow p(x_t, y_t) + g(x_s, x_t, y_s, y_t)$$

$$p(x_t - x_s, y_s, y_t) \Leftarrow p(x_t - x_s, y_s, y_t) + g(x_s, x_t, y_s, y_t)$$

(c) For all the marginals

$$p(x_s, y_s) \Leftarrow p(x_s, y_s)/Z_\lambda$$

$$p(x_t, y_t) \Leftarrow p(x_t, y_t)/Z_\lambda$$

$$p(x_t - x_s, y_s, y_t) \Leftarrow p(x_t - x_s, y_s, y_t)/Z_\lambda$$

(2) Calculate all $\Delta\lambda$

$$\begin{aligned}
\Delta\lambda(x_s, y_s) &= \frac{1}{3} \ln \frac{q(x_s, y_s)}{p(x_s, y_s)} \\
\Delta\lambda(x_t, y_t) &= \frac{1}{3} \ln \frac{q(x_t, y_t)}{p(x_t, y_t)} \\
\Delta\lambda(x_t - x_s, y_s, y_t) &= \frac{1}{3} \ln \frac{q(x_t - x_s, y_s, y_t)}{p(x_t - x_s, y_s, y_t)}
\end{aligned}$$

(3) Update λ according to: $\lambda \Leftarrow \lambda + \Delta\lambda$

3. Goto 2. if λ have not converged.

The number of exponential operations involved in the upper algorithm is about n, 3 times less than the former algorithm. In practice, we observed an acceleration of more than 3 times in experiments, this is probably due to the avoidance of the subscript searching process in the calculation of $\Delta\lambda(x_t - x_s, y_s, y_t)$.

The TFOM with MaxEnt-estimated two-site marignals shall be referred to as TFOM2 later on.

3.4 Discussion

In this chapter, we first estimate the parameters of the HMM by establishing equations of local characteristics of the MRF. This method works for the HMM and for the FOM. In view of the fact that the distributions of MRFs on tree graphs can be analytically expressed as products of marginals, we tackle the parameter estimation of FOM through tree approximations. Supposing the underlying graphs are trees, the parameters can be expressed in terms of empirical marginal probabilities that can be estimated from the training set. However, direct estimation of the FOM parameters (the joint probabilities of two sites) from the training set will cause over-fitting since the orginal parameter space is huge even for small images. Two methods are explored to get around this difficulty. The first method is to decrease the dimensionality by assuming the independence of one-site marginal conditioned on the skin label and the gradient of neighboring colors conditioned on neighboring labels. The other method is to estimate the high-dimensional parameters from low-dimensional, i.e., empirically estimable, parameters from the training set with the maximum entropy principle. The later method seems more natural and more promising since it again lies inside the framework of maximum entropy principle, which assumes no artificial bias or prejudice on the parameters. Now that we have all the parameters for the MRF models. We are going to apply these models to skin probability estimation in the following chapters.

Résumé (Chapitre 4)

Lors de chapitres précédents nous avons estimé les paramètres des différents modèles obtenus par MaxEnt. Nous allons maintenant utiliser ces différents modèles pour l'inférence. C'est à dire calculer la probabilité qu'un pixel soit un pixel de peau à partir d'une image couleur.

Nous évaluons dans ce chapitre une technique classique d'inférence dans le contexte de l'imagerie appelée Markov Chain Monte Carlo (MCMC). En effet, comme l'espace d'état est très grand, de dimension 2^n où n est la dimension de l'image, il est impossible de simuler directement un échantillon issu d'un modèle Markovien. L'algorithme d'échantillonnage de Gibbs permet de transformer la complexité spatiale en complexité temporelle. Pour une image donnée, nous générons une séquence d'images de labels. Les probabilités pour "peau" et "non peau" sont alors estimées en effectuant une moyenne empirique.

Les résultats expérimentaux confirment la supériorité du modèle Markov Caché (HMM) sur le modèle de référence (baseline) et du modèle "arbre au premier ordre" (TFOM) sur le modèle HMM.

Chapter 4

Gibbs sampling

For a new input color image x, skin detection requires to compute for each pixel s the quantity $p(y_s|x)$. Direct summing over all other sites is computationally intractable due to the huge image space and the large number of terms in the partition function. One can sample from the distribution $p(y|x)$ instead, to get a sequence of label images $y^{(0)}, y^{(1)}, y^{(2)}, \ldots, y^{(n)}$ for the given color image x. By averaging over the samples, we can get all the marginals which are of our interests. However direct sampling from the joint distribution is impossbile due once more to the huge image space and the intractable partition function. Notice that all our models are MRF for certain neighborhood systems, i.e., they are Gibbs field. Now sampling from these fields can be achieved through Markov Chain Monte Carlo (MCMC) methods. The main idea of MCMC methods is to turn the intractable spatial complexity into the feasible temporal complexity through Markov chain. In what follows we will first introduce the Gibbs sampler in brief. For a detailed account of the Gibbs sampler, please refer to the classical book [Win03].

4.1 Gibbs fields

We can define an *energy function* H for a random field X. Then the distribution π of the form,

$$\pi(x) = \frac{\exp(-H(x))}{\sum_z \exp(-H(z))}, \forall x \in \mathbf{X} \tag{4.1}$$

is called a *Gibbs field* induced by H. We notate the denominator in (4.1) as Z, i.e.,

$$Z = \sum_z \exp(-H(z)) \tag{4.2}$$

It is often called *partition function* or *normalization function* since it ensures $\sum_{x\in\mathbf{X}} \pi(x) = 1$. The energy function H captures the interaction between sites. In fact, any field can be writen in this way since we can always define $H(x) = \ln(Z\pi(x)), \forall x \in \mathbf{X}$, for any specified constant Z. Gibbs fields have a special role that identify them among all other distributions as stated in the following theorem.

Theorem 1 (Gibbs variational principle) *Let H be an energy function defined on the configuration space $\mathbf{X}$, Z the partition function defined by (4.2), μ any distribution on $\mathbf{X}$, π the Gibbs distribution as in (4.1), then the following inequality holds:*

$$E(H;\mu) - \mathcal{H}(X;\mu) \geq -\ln Z \tag{4.3}$$

with equality if and only if $\mu = \pi$.

The proof is referred to [Win03]. This principle states that among all distributions with the mean energy $E(H;\pi)$, the Gibbs field π has the maximum entropy $\mathcal{H}(X;\pi) = \mathcal{H}_{\max}$ and thus is the most disordered/uniform state. The left side of the inequality in (4.3) is called the *free energy*.

It is often convenient to decompose the energy function into some terms defined on subsets of S. For this purpose, we introduce the concept of *potentical.* A potential $\mathcal{U}$ is a family of functions $\{\mathcal{U}_A; A \subseteq S\}$ on X such that

1. $\mathcal{U}_\emptyset = 0$,
2. $\mathcal{U}_A(x) = \mathcal{U}_A(y)$ if $x_s = y_s, \forall s \in A$.

In other words, $\mathcal{U}_A(X)$ only depends on the configuration on the subset A and it vanishes when A is an empty set. The energy function H of a Gibbs field can always be writen in the potential form: $H_{\mathcal{U}}(X) = \sum_{A\subseteq S} \mathcal{U}_A(X)$. $\mathcal{U}$ is called a *neighbor potential* with respect to a neighborhood system $(S,\mathcal{V})$ if $\mathcal{U}_A = 0$, $\forall A \subseteq S, A \notin \mathfrak{C}$, with $\mathfrak{C}$ being the set of cliques in $(S,\mathcal{V})$. The corresponding Gibbs field π is then called a *neighbor Gibbs field* and takes the following form:

$$\pi(x) = \frac{\exp(-\sum_{A\in\mathfrak{C}} \mathcal{U}_A(x))}{\sum_{z\in\mathbf{X}} \exp(-\sum_{A\in\mathfrak{C}} \mathcal{U}_A(z))} \tag{4.4}$$

In other words, the energy function only expands on cliques for a neighbor Gibbs field. $\mathcal{U}$ is a *pair potential* when $\mathcal{U}_A = 0$, $\forall A \subseteq S, |A| > 2$. We can easily check the pairwise MRF in (2.23) has a pair potential. The neighbor potentials of the 4-neighborhood system on pixel lattices are also pair potentials.

There exists a very important equivalence between the MRF and neighbor Gibbs fields as stated in the following theorem.

Theorem 2 (Equivalence theorem) *Let a neighborhood system $\mathcal{V}$ be given for S, then a random field π is a Markov random field for $\mathcal{V}$ if and only if it is a neighbor Gibbs field for $\mathcal{V}$.*

The proof is referred to [Win03]. This theorem is also called *Hammersley-Clifford* theorem.

4.2 Gibbs sampling

The limit theorem of Markov chain (see Appendix D) tells us that if we have a primitive Markov kernel p whose invariant distribution is μ, then no matter which state it starts with, it should approach μ in the long run. For Gibbs sampling we would replace μ with the target Gibbs fields π. Then the aim is to find a primitive Markov kernel p, such that π is invariant to p. Let us consider the following Markov kernels for each $I \subseteq S$:

$$\pi_I(y|x) = \begin{cases} \pi(y_I|x_{S\setminus I}) & \text{if } y_{S\setminus I} = x_{S\setminus I} \\ 0 & \text{if not} \end{cases} \tag{4.5}$$

These Markov kernels are again called *local characteristics* of π. For a MRF with the neighborhood system $\mathcal{V}$ or, say, a neighbor Gibbs field for $\mathcal{V}$, the local characteristics boil down to the following:

$$\pi_I(y|x) = \begin{cases} \pi(y_I|x_{\mathcal{V}_I}) & \text{if } y_{S\setminus I} = x_{S\setminus I} \\ 0 & \text{if not} \end{cases} \tag{4.6}$$

These take advange of the properties of MRF. The invariation of the Gibbs field π with respect to its local characteristics is assured by the following theorem [Win03]:

Theorem 3 *The Gibbs field π is reversible and invariant for its local characteristics π_I, $I \subseteq S$.*

We can now build the primitive transition kernel p whose invariant distribution is the Gibbs field π. The idea is to use the local characteristics of π. For a finite set S of sites, the enumeration $(s_1, \ldots, s_{|S|})$ is called a *visiting scheme*. We define the transition kernel for our Markov chain as

$$p(y|x) = \pi_{\{s_1\}} \ldots \pi_{\{s_{|S|}\}}(y|x), \forall x, y \in \mathbf{X} \tag{4.7}$$

The Markov chain with the transition kernel (4.7) induces the following algorithm: An initial configuration x is chosen, or picked at random following the initial distribution ν. Then site s_1 is chosen and the configuration is updated according to the single-site local characteristic $\pi_{\{s_1\}}(y_{s_1}x_{S\setminus\{s_1\}}|x_S)$. This procedure is repeated for each site in S according to the visiting scheme $(s_1, \ldots, s_{|S|})$. When all the sites are visited, we say a *sweep* is done, which means a transition according to the kernel p is finished. Since the Gibbs field π is invariant to each single-site local characteristic, it is invariant to the transition kernel p. And since each single-site updating in a sweep is always done with a positive probability, the transition probability p has to be strictly positive. Following Theorem 6 (see Appendix D), we have the following conclusion:

Theorem 4 *For a Gibbs field π with the transition kernel (4.7), we have*

$$\nu p^n \longrightarrow \pi, \text{as } n \longrightarrow \infty \tag{4.8}$$

for any initial distribution ν.

Following Theorem 7 (see Appendix D), we have

Theorem 5 *Let π be a Gibbs field with the transition kernel p defined by (4.7), ν is any initial distribution, $(X^{(0)}, X^{(1)}, \dots, X^{(n)})$ is the generated Markov chain, then we have*

$$E_{\nu p^n}\left[\left(\frac{1}{n+1}\sum_{i=0}^{n} f(X^{(i)}) - E_\pi(f)\right)^2\right] \longrightarrow 0, \quad \text{as} \quad n \longrightarrow \infty$$

for any function f.

If we choose the following functions, $\forall s \in S$, $\forall x_s \in \mathbf{X}_s$,

$$\delta_{x_s}(X_s) = \begin{cases} 1 & \text{if } X_s = x_s \\ 0 & \text{if not} \end{cases}$$

according to Theorem 5, we have,

$$\begin{aligned} & E_{\nu p^n}\left[\left(\frac{1}{n+1}\sum_{i=0}^{n} \delta_{x_s}(X_s^{(i)}) - E_\pi[\delta_{x_s}(X_s)]\right)^2\right] \\ = \; & E_{\nu p^n}\left[\left(\frac{1}{n+1}\sum_{i=0}^{n} \delta_{x_s}(X_s^{(i)}) - \pi(X_s = x_s)\right)^2\right] \\ \longrightarrow \; & 0, \text{as } n \longrightarrow \infty, \forall s \in S \end{aligned}$$

where $X_s^{(i)}$, $0 \leq i \leq n$ are the random variables at site s after i transitions. Therefore we can estimate the marginals of the Gibbs field π through simply averaging the δ functions on the sequence.

In the case of HMM defined in (2.26), we generate, using the Gibbs sampling algorithm, a sequence of label images

$$y^{(0)}, y^{(1)}, \dots, y^{(n_0)}, \dots, y^{(n)}$$

with stationary distribution (2.26). Then, we estimate the quantity $p(y_s|x)$ by the empirical mean

$$\frac{1}{n-n_0}\sum_{j=n_0+1}^{n} y_s^{(j)}$$

The Gibbs sampling algorithm used in our experiments is presented in detail in Algorithm 1. Note that u and y are matrices defined on the pixel lattice S and

$$p(Y_s=1|y_{(s)},x)=\frac{p(Y_s=1,y_{(s)}|x)}{\sum_{y_s}p(y|x)}=\phi(U(x;y)) \tag{4.9}$$

with

$$U(x;y)=\sum_{t\in\mathcal{V}_s}(a_1y_t-a_0(1-y_t))+\ln\frac{q(x_s|Y_s=1)}{q(x_s|Y_s=0)} \tag{4.10}$$

where ϕ is the logistic function and $\mathcal{V}_s$ are the neighbors of s. The algorithm is consistent in the sense that as $n\to\infty, \forall s\in S, u_s\to p(Y_s=1|x)$, according to Theorem 5.

Algorithm 1 Gibbs sampling algorithm

```
$u \Leftarrow 0$
randomly initialize the binary image $y^{(0)}$
**for** $j = 0$ to $n-1$ **do**
  $y \Leftarrow y^{(j)}$
  **for all** $s \in S$ **do**
    sample $Y_s^{(j+1)}$ according to $p(Y_s=1|y_{(s)},x)$
  **end for**
  **if** $j+1 > n_0$ **then**
    $u \Leftarrow u + y^{(j+1)}$
  **end if**
**end for**
$u \Leftarrow u/(n-n_0)$
```

In the case of TFOM, as for the HMM, the same algorithm leads to the following conditional distribution, $\forall s\in S$,

$$p(Y_s=1|y_{(s)},x)=\phi(U(x;y)) \tag{4.11}$$

with

$$U(x;y)=\sum_{t\in\mathcal{V}_s}\ln\frac{q(x_s,x_t,Y_s=1,y_t)}{q(x_s,x_t,Y_s=0,y_t)}-(n(s)-1)\ln\frac{q(x_s,Y_s=1)}{q(x_s,Y_s=0)} \tag{4.12}$$

4.3 Experiments

All experiments are made using the following protocol. The labeled Compaq database contains about $13,562$ photographs, in which there are $4,649$ photographs with skin and $8,913$ photographs without skin. Each photograph with skin is accompanied with a binary mask image indicating skin and non-skin regions. These masks were obtained by manual labeling, see Appendix A for a detailed account. This database is split into two almost equal parts randomly. The first part, containing nearly 674 million pixels is used as training data while the other one, the test set, is left aside for ROC curve computation. We use 32 bins per RGB channel in the following experiments. In what follows in the current section, the TFOM refers to TFOM1, the reason is that the associated project Poesia [SDT$^+$02], which aims to develop an Internet content filtering platform, requires real-time processing of different media including images. We developed TFOM1 first with the Gibbs sampling method. We found this scheme somehow too slow and not practical for usage under real environments. Later we turned to a faster algorithm —belief propagation—which we will introduce in the next chapter. The TFOM2 was developed after we adopted the belief propagation scheme.

Our working parameters for the Gibbs sampling algorithm are $n_0 = 0$ and $n = 99$. Some examples are presented in Figure 4.1, where first column displays test input images. The other columns display gray level output of different models. The gray-level is proportional to the evaluated quantity $p(Y_s = 1|x)$. HMM compares favorably with the Baseline model. The skin zones detected with the baseline model are generally blended with background false alarms in complex images. The HMM outputs are cleaner with real skin zones emphasized. In the first image, the Baseline model does not detect part of skin on the neck of the right person. In the second image, the Baseline model detects some skin pixels on the monitor and on the diskette package, which is false. The Baseline model gives very low probabilities to the hands of the operator sitting before the monitor, which could be easily ignored when the output is binarized with respect to some threshold. In the third image, a lot of background objects are marked with high skin probabilities, such as the wall, the poster, hair, etc. On the contratry, such problems are much alleviated by HMM. As for TFOM, one can visually appreciate the improvement in localization of the skin zones compared to the HMM. The detected skin regions are more precise. It is easier to recognize the shapes of the faces and hands than the HMM. In the first image, TFOM gives more precise contour of the glasses and the left hand of the right person. In the second image, the shape of the faces of the operator is obviously improved with respect to HMM. In the third image, we can see more details of hands, faces with TFOM than with HMM.

However, there is obvious misclassification of non-skin pixels as skin pixels on the dog of the

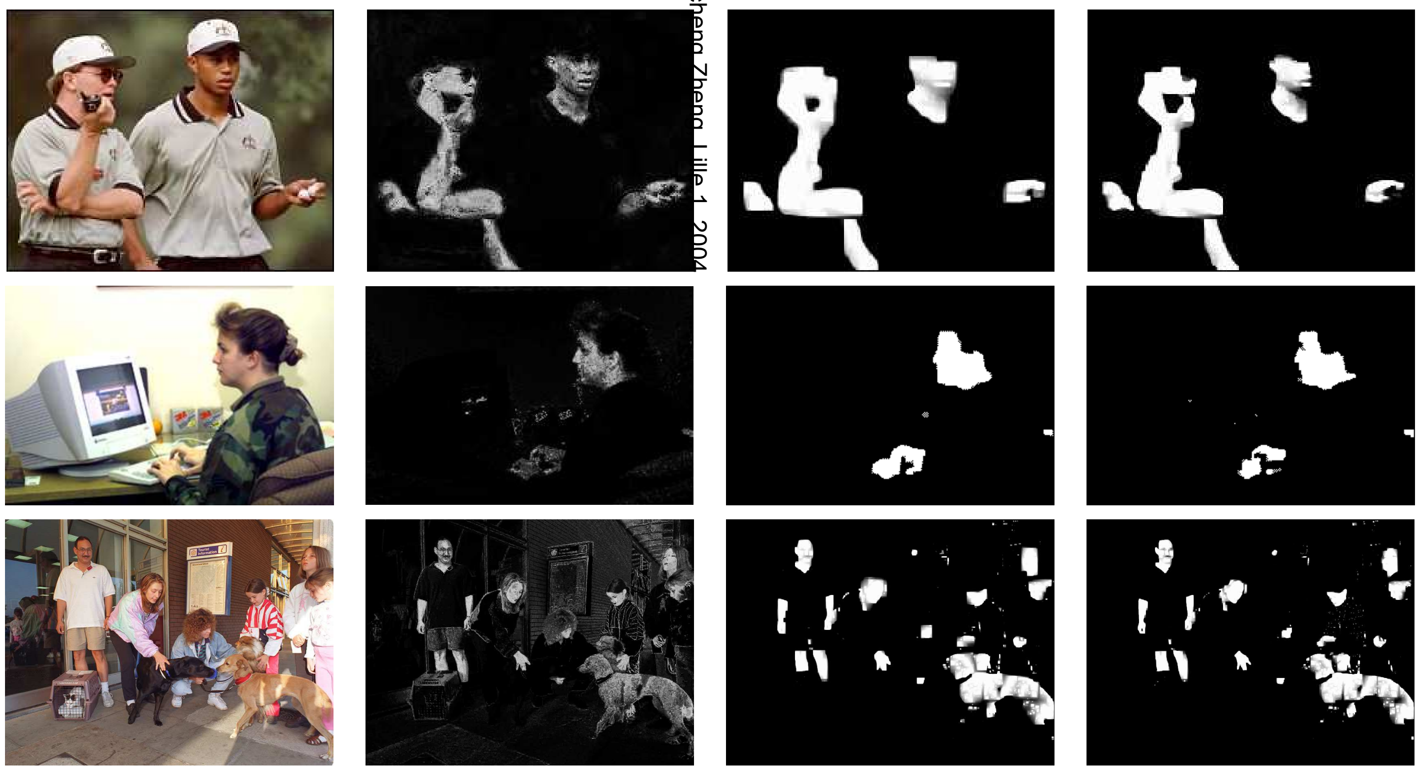


Figure 4.1: **First column:** original color images. From top to bottom, the sizes of the images are respectively 225 × 180 pixels, 300 × 200 pixels and 800 × 599 pixels. **Second column:** Baseline model. **Third column:** hidden Markov model. **Fourth column:** first order model. In the computed images, the grey level is proportional to the skin probability evaluated with the specified model.

third image for all three models. This is generally due to the similarity of its color to skin color. High-level features such as the shape of the skin regions might help discriminate it from real skin. Detailed examination of the pictures reveals that the discussed models are still far from reaching human performances. For example, the left arm of the right-most person in the first image of Figure 4.1 is visible in the Baseline model and not in the subsequent ones. Remark that the gray values indicating the probabilities for skin are very low. A zoom is provided in Figure 4.2. It is understandable that the regularizing models, HMM as well as the First Order Model, operating at the level of pixels, have produced a posterior probability that put very low likelihood for skin in this region. Indeed, the local evidence for skin is low and the neighboring values are also indicating low evidence. A high level model of limbs might be able to overcome these difficulties.

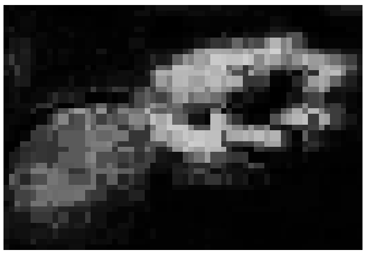


Figure 4.2: Zoom of first image, second column in Figure 4.1. Result of the Baseline model

Bulk results in the ROC curves are shown in Figure 4.3, which are calculated from 100 images (around 10 million pixels), randomly extracted from the test set. The Baseline model (with crosses) permits to detect more than 80% of the skin pixels with less than 10% of false positive rate. The ROC curve of HMM indicates an increase close to 2% in detection rate for the same false positive rate as the Baseline model, or a drop of about 1% in false positive for the same detection rate as the Baseline model. For example, setting 10% of flase positive rate, the Baseline model permites to detect 81% of skin pixels in average, while the HMM permits to detect 83% in average. We show now that this is significative. The test set is made of 100 images disjoint from the training set. This amounts to about 10^7 pixels, out of which about 6% are labeled as skin. These 6×10^5 pixels cannot be considered as independent since the color values of the images are correlated at small distance. Hence, we choose one out of ten of these pixels leading to a sample size of 6×10^4. For the Baseline model, the output label $y_\cdot^{(i)}, \forall i = 1, 2, \ldots, 6 \times 10^4$ has a binary distribution (also known as Bernoulli distribution) $b(0.81)$, with probability 0.81 for the label being 1 and probability $1 - 0.81$ for the label being 0. Its variance is $0.81(1 - 0.81)$, or equivalently, its standard deviation is $\sqrt{0.81(1 - 0.81)}$. Now consider the random variable $y_\cdot = \frac{1}{6\times10^4} \sum_{i=1}^{6\times10^4} y_\cdot^{(i)}$. It is a random variable with the same mean value 0.81, but with the variance $\frac{0.81(1-0.81)}{6\times10^4}$, or with the standard deviation $\sqrt{0.81(1-0.81)}\times(\sqrt{6\times10^4})^{-1}\leq 2\times10^{-3}$. The hypothesis that the proportion of 83% was due to random fluctuations is then rejected with a p-value close to 0. The ROC curve of TFOM shows an improvement of performance of around 1% over that of HMM. At 10% of false positive rate, the TFOM permits to detect around 84% of skin pixels. With a sample size of 6×10^4 pixels, the standard deviation around the HMM value is then $\sqrt{0.83(1-0.83)}\times(\sqrt{6\times10^4})^{-1}\leq 2\times10^{-3}$. The hypothesis that the proportion of 84% was due to random fluctuations is then rejected with a p-value close to 0.

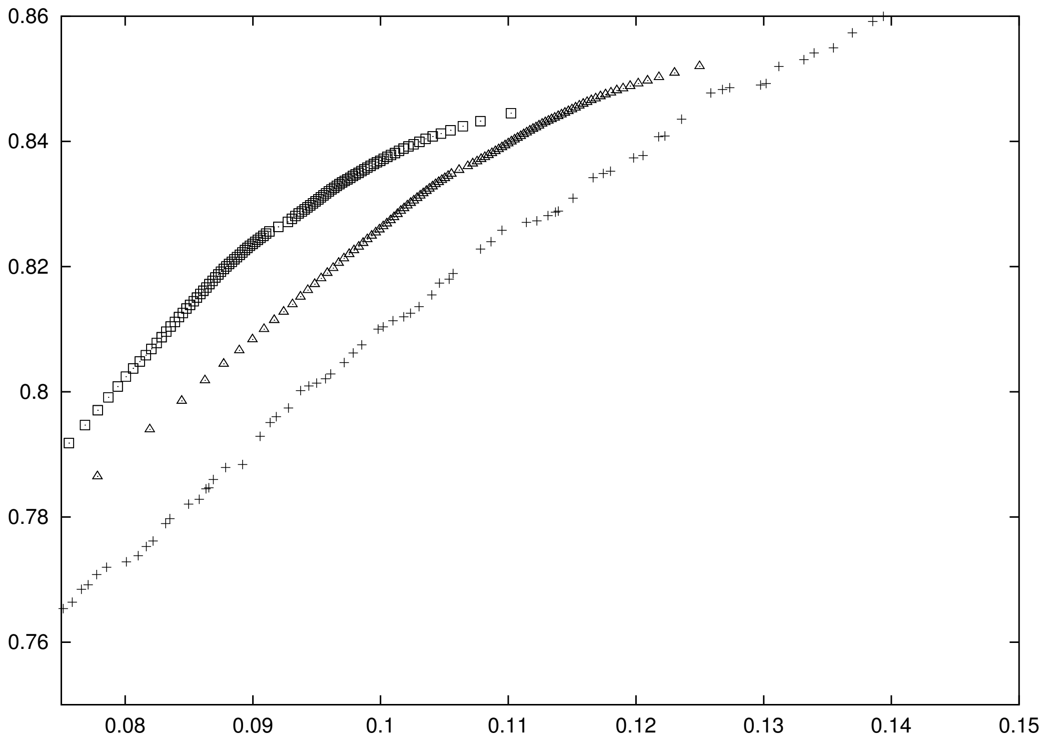


Figure 4.3: Receiver Operating Characteristics (ROC) curve for each model. x-axis is the false positive rate, y-axis is the detection rate. Baseline model is shown with crosses, HMM model with triangles, while the first order model is shown with squares.

The running time of both HMM and TFOM is proportional to the number of loops and the size of the input images. Letting $\mathcal{T}$ being the elapsed time for an input image with pixel lattice S, then $\mathcal{T}\approx n\times|S|\times\mathcal{T}_\phi$, where $\mathcal{T}_\phi$ is the time consumed to evaluate the sigmoïd function defined in (4.9) or in (4.11). For a PC with a Pentium 4 processor at 1.7 Ghz and 256 MB memory, the Baseline model takes about 0.088 second per image, HMM about 13.03 seconds per image, and TFOM1 about 23.25 seconds per image.

For many applications involving skin detection as an intermediate stage, processing time is of major importance. Much of our work is associated with Poesia [SDT$^+$02] project. This project aims to filter Internet content including images, text, etc., under real environment. So it has a requirement for real-time processing of all kinds of media, including images. The running time required by Gibbs sampling is clearly at a distance from the real-time requirement. In the following sections we replace the stochastic sampling algorithm by a deterministic scheme —belief propagation [YFW02]—in order to meet the required time constraints.

4.4 Discussion

In this chapter, we applied the models we built in the precedent chapters to estimating skin probabilities at pixel locations of given input images. The method we explored in this chapter is Gibbs sampling. Due to the high dimensionality of general images, it is impossible to sample directly from the image space according to the MRFs. We turned to the Markov chain Monte Carlo method. We showed that the MRFs are equivalent to Gibbs fields, which can be sampled with the local characteristics of the MRFs. Thus we transformed the intractable spacial complexity to the tractable temporal complexity with Gibbs sampling. For a given input image, we sample from its label image space according to the local characteristics of the Markov models. We build a sequence of label images for the given image. The skin probabilities on pixel locations are then easily computed from the samples. Experimental results reveal the superiority of the FOM to the HMM, and in turn to the Baseline model. However, one drawback of Gibbs sampling is that it generally takes a long time to get enough samples. And the method depends seriously on the starting configuration of the label image. In real environment we have the requirement to classify pixels quickly. Next chapter will explore a fast algorithm—the belief propagation algorithm—for probability inference.

Résumé (Chapitre 5)

Nous supposons que les graphes sont localement approximés par des arbres $\mathcal{T}_k^s$. Nous souhaitons calculer les marginales $p(y_s|x_t, t \in \mathcal{T}_k^s)$. Un calcul direct de cette quantité est très gourmand au temps de calcul. Un algorithme élégant et efficace appelé en anglais belief propagation (BP) et en français propagation de croyance permet de calculer les marginales en un temps linéaire en fonction de la taille des images. Son applicaition à la détection de la peau en temps réel montre, à travers la courbe ROC, les performances de cet algorithme.

Nous explorons deux types d'approximation par des arbres du graphe des pixels. La première, classique, se nomme les arbres de Bethe. La seconde, que nous introduisons, consiste à choisir des arbres en étoile. Nous montrons que l'algorithme BP accélère de manière très importante la vitesse de l'algorithme d'inférence en comparaison avec les algorithmes du chapitre précédent pour une même qualité de résultat.

Chapter 5

Belief propagation on tree graphs

Supposing the graphs under the MRF are trees, our aim is to compute for each pixel s, the quantity $p(y_s|x_s, s \in \mathcal{T}_k)$, for p being the THMM or the TFOM we built previously and for k ranging from 1 to say 5. This computation can be done exactly. Moreover, it can be done efficiently using the belief propagation (BP) algorithm [YFW02]. This algorithm has been discovered in different scientific communities. It is called BP in A.I., Viterbi algorithm in the special case of line graphs and dynamic programming in combinatorial optimization. It is a method to infer marginals from the joint probability models based on their local characteristics in a way called *messages* on directed graphs.

5.1 Graphical models

Inference problems can often be guided by *graphical models*, in which we represent variables and their relationship with *nodes/vertices* and *edges*. Inference on graphical models are implemented in many disciplines like AI, computer vision, statistical physics and error-correcting coding theory [YFW02], thus it is not surprising to see many variants of graphical models. In the following we shall introduce these models with *Bayesian networks* and *pairwise Markov random field* stressed.

5.1.1 Bayesian networks

Bayesian networks are directed graphs where the states or distributions of some nodes are dependent only on some causal nodes, so-called *parents* of these nodes. Let us take a simple example from [Pea88]. Consider an experiment with two coins and a bell. We toss the coins independently, and the bell rings whenever the outcomes of these two coins are the same. We can depict this experiment by Fig. 5.1. As shown in this figure, the state of the bell b depends on those of the two coins $c1$ and $c2$. This relationship in a formula is $p(b|c1, c2)$. $c1$ and $c2$ are both *parents* of b. So b is their *child* accordingly. In the graph, the relationship is shown with arrows pointing from the parents to their child. Their joint probability is given by

$$p(c1, c2, b) = p(c1)p(c2)p(b|c1, c2)$$

A Bayesian network defines an independent structure, where the states of nodes are only dependent on their direct parents and independent of other nodes given their parents. Generally a Bayesian network is a directed acyclic graph of n random variables x_i which are related by arrows, whose joint probability takes the following form:

$$p(x_1, \dots, x_n) = \prod_{i=1}^{n} p(x_i|par(x_i)) \tag{5.1}$$

where $par(x_i)$ denotes the parent(s) of x_i. For nodes without any parent, $p(x_i|par(x_i))$ are replaced by the marginals $p(x_i)$.

Our goal in general is to evaluate some marginal probabilities. For example, we might want to know the probability that the bell rings $p(b = \text{“ringing”})$. In order to do this, we need to sum out all the other variables in general:

$$p(b = \text{“ringing”}) = \sum_{c1} \sum_{c2} p(c1, c2, b = \text{“ringing”})$$

If we can observe the state $c1$ but not $c2$, then we say $c1$ is *observable* and $c2$ *hidden*. For example we know $c1 = \text{“head”}$ by observation, then we need not sum over the state $c1$ in order to know the probability that the bell rings, we have instead

$$p(b = \text{“ringing”}|c1 = \text{“head”}) = \frac{\sum_{c2} p(c1 = \text{“head”}, c2, b = \text{“ringing”})}{p(c1 = \text{“head”})}$$

For simple networks like the “coins and bell” example, direct summation is still doable. But for some large Bayesian networks which frequently emerge in image processing, the numbers of hidden nodes are generally too huge to sum up the hidden states directly since the number of summations increase exponentially with the number of nodes. The virtue of belief propagation is that we can calculate the marginal probabilities, at least in an approximate way, in a time that only grows linearly with the number of nodes in the network, as we shall see in the following sections. The calculated (approximate) marginal probabilities are called *beliefs*.

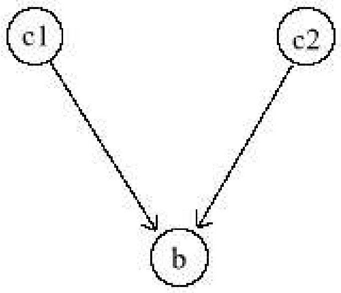


Figure 5.1: The Bayesian network for the "coins and bell" example. $c1$, $c2$ and b denote the states of the two coins and bell respectively.

5.1.2 Pairwise Markov random fields

Pairwise MRFs provide attractive models for computer vision problems such as image segmentation or image restoration [GG84]. They are undirected graphs in which there exist only pairwise relationship between nodes. There is no causal node in such graphs like those in the Bayesian networks where there are causal nodes called "parents" of other nodes. Our MRFs fit well with such graphical models. The graph for the HMM is shown in Fig. 5.2 for a lattice of 4×4. The label image y is the hidden layer with nodes y_s represented by empty circles. The color image x is the observed layer with nodes x_s represented by filled-in circles. The links between neighboring nodes represent the potential functions of the corresponding nodes. Two sorts of potentials are concerned here: the local evidence $\phi_s(x_s, y_s)$ of y_s from x_s and the compatibility $\psi_{st}(y_s, y_t)$ between the states y_s and y_t of neighboring sites s and t, which are defined as the following according to our setting,

$$\left\{ \begin{array}{lcl} \phi_s(x_s, y_s) & = & q(x_s | y_s) \\ \psi_{st}(y_s, y_t) & = & \exp(a_0(1 - y_s)(1 - y_t) + a_1 y_s y_t) \end{array} \right. \tag{5.2}$$

$\phi_s(x_s, y_s)$ give the clue for the underlying image y based on our observation of the color image x. $\psi_{st}(y_s, y_t)$ represent our prior "structure" knowledge about y. This Markov random field is called "pairwise" since all compatibility functions only depend on pairs $s \sim t$ of nodes. Our goal is the same as in Bayesian networks: to compute the beliefs $b(y_s)$ on individual nodes so that we can infer something, the skin/non-skin label in our case, about the underlying scene y. Once again a direct computation of marginal probabilities will take exponential time with respect to the number of nodes to be summed up, which is generally impossible to conduct in the case of images. Thus we have to fall back on some fast algorithm such as belief propagation.

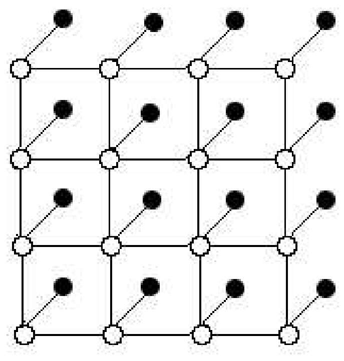


Figure 5.2: The graphical model of the pairwise Markov random field corresponding to the HMM in Section 2.5. The empty circles are the nodes of the label image y. They compose the hidden layer. The filled circles are the nodes of the color image x. They are the observable variables. The links between nodes represent the potential functions of the linked nodes

5.1.3 Other graphical models

Another problem for which the BP algorithm has given excellent results is the iterative decoding of error-correcting codes [YFW02]. BP is the decoding algorithm used to decode some of the best practical codes, including turbocodes [BGT93] and Gallager codes [Gal63]. Both of these codes can be formulated as parity-check codes and repersented by *Tanner graphs* [Tan81]. A Tanner graph is a pictorial representation of the parity-check constraints on the codewords (legal configurations of bits) of an error-correcting code. Inferring the initial transmitted codeword y based on the received word x leads to computing the marginal probability for each bit. For large coding system we resort to the BP algorithm. Besides Tanner graphs, there are *factor graphs* [KFL01] which are generalization of Tanner graphs.

As shown in [YFW02], every Bayesian network or pairwise MRF can be converted to a factor graph and vice versa. Thus all the graphical models that we have discussed are equivalent. Without losing generality, we shall focus our discussion about the BP algorithm on pairwise

MRF.

5.2 Belief propagation (BP) algorithm

"Inference" problems arise in statistical physics, computer vision, error-correcting coding theory, and AI. The BP algorithm is an efficient way to solve inference problems based on passing local messages [YFW02]. It consists in iterative updating of *messages* that are sent between nodes. The message $m_{ts}(x_s)$ is explained as how much the node t would consider the node s to be in the state x_s. The belief $b_s(x_s)$ at the node s is proportional to the local evidence $\phi_s(x_s)$ at s and the product of all messages getting to s, $m_{ts}(x_s)$, $t \in \mathcal{V}_s$,

$$b_s(x_s) \propto \phi_s(x_s) \prod_{t \in \mathcal{V}_s} m_{ts}(x_s) \tag{5.3}$$

The messages are determined self-consistently by the following updating rules,

$$m_{ts}(x_s) \longleftarrow \sum_{x_t} \left(\phi_t(x_t)\psi_{st}(x_s, x_t) \prod_{u \in \mathcal{V}_t \setminus \{s\}} m_{ut}(x_t) \right) \tag{5.4}$$

That is, in order to compute the message $m_{ts}(x_s)$, we summarize all the messages from the neighbors of t, except s, to t, then scale the product by the local evidence of t and the compatibility between s and t. Finally, we should sum up all the possibilities of x_t so that the final result is the average tendency that t thinks s should have to be in the state x_s. In a practical computation, we start with the nodes at the edges of the graphs. The initial messages are generally set to be unbiased. A message is calculated only when all necessary messages are available. Figure 5.3 shows the message passing routes for an example. It is easy to see that each message need only be computed once for a loop-free graph. Thus the whole computation takes a time proportional to the number of links in the graph, which is dramatically less than the time needed to compute the marginal probabilities naively.

The BP algorithm, as defined in terms of equations (5.3) and (5.4), does not depend on the topology of the graphs it runs on. Thus it can be applied on graphs with loops, too. We simply start from unbiased initial messages, and use the message updating rules (5.4) to iterate the messages until possibly they converge. We can then use (5.3) to read off the beliefs we are interested in. However we can never ignore the loops existing in the graphs since the algorithm is not guaranteed to converge for graphs with loops. As Pearl warned, "if we ignore the existence of loops and permit the nodes to continue communicating with each other as if the network were singly connected, messages may circulate indefinitely around these loops, and the process may not converge to a stable equilibrium" [Pea88]. One can indeed find examples of graphical models with loops, where, for certain parameter values, the BP algorithm fails to converge, or predicts beliefs that are inaccurate [YFW02]. On the other hand, the BP algorithm has been successful as a decoding algorithm for error-correcting codes defined on Tanner graphs that have loops, and also for some computer vision problems where the underlying MRF is full of loops [FPC00] [YFW02].

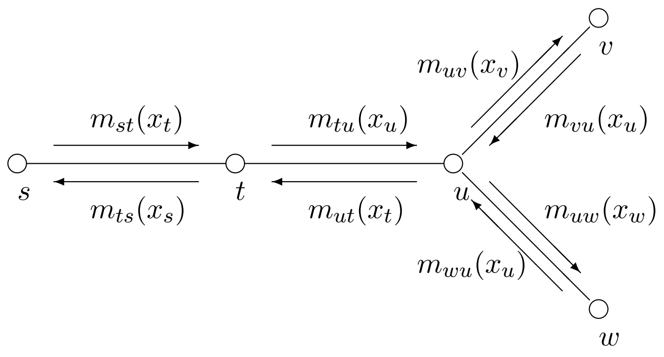


Figure 5.3: Messages on a simple graph

Remark that both the THMM defined in (2.26) and the TFOM defined in (3.11) can be represented by the following generic pairwise model expressed in terms of potentials:

$$p(y_{\mathcal{T}_k^s}|x_{\mathcal{T}_k^s}) \propto \prod_{u\sim v; u,v\in\mathcal{T}_k^s} \psi(x_u, x_v, y_u, y_v) \prod_{u\in\mathcal{T}_k^s} \phi(x_u, y_u) \tag{5.5}$$

For the THMM, the potentials are:

$$\begin{cases} \psi(x_u, x_v, y_u, y_v) &= a_0(1-y_u)(1-y_v) + a_1 y_u y_v \\ \phi(x_u, y_u) &= q(x_u|y_u) \end{cases} \tag{5.6}$$

In the case of TFOM, the potentials are then:

$$\begin{cases} \psi(x_u, x_v, y_u, y_v) &= \frac{q(x_u, x_v, y_u, y_v)}{q(x_u, y_u)q(x_v, y_v)} \\ \phi(x_u, y_u) &= q(x_u, y_u) \end{cases} \tag{5.7}$$

In order to get the marginal probability $p(y_s|x_{\mathcal{T}_k^s})$ from the model in (5.5), the BP algorithm consists in computing the messages $m_{ts}(y_s)$, $t \in \mathcal{V}_s$, then the marginal probability is computed according to

$$p(y_s|x_{\mathcal{T}_k^s}) \propto \phi(x_s, y_s) \prod_{t\in\mathcal{V}_s} m_{ts}(y_s) \tag{5.8}$$

This in turn leads to computing iteratively all the messages required by $m_{ts}(y_s)$, $t \in \mathcal{V}_s$:

$$m_{vu}(y_u) \leftarrow \sum_{y_v} \phi(x_v, y_v)\psi(x_u, x_v, y_u, y_v) \prod_{w \in \mathcal{V}_v, w \neq u} m_{wv}(y_v) \tag{5.9}$$

The initial messages from the leaves of the tree graphs are initialized with the value 1.

There is a close connection between the BP algorithm and the Bethe approximation of statistical physics. In particular, BP can only converge to a fixed point that is also a stationary point of the Bethe approximation to the free energy [YFW02]. We shall show the equivalence of BP algorithm and the minimization of Bethe free energy in the following sections. The demonstration basically follows the idea in [YFW02], with deduction and some notations adapted to the context of this thesis.

5.3 Bethe free energy

Recall the free energy defined in (4.3). We rewrite it in the following:

$$F = E(H; \mu) - \mathcal{H}(X; \mu) \tag{5.10}$$

where $E(H; \mu)$ is the average energy and $\mathcal{H}(X; \mu)$ the entropy. By Gibbs variational principle, for a given energy function H defined on the configuration space $\mathbf{X}$, the free energy gets minimized only when the distribution μ is exactly the Gibbs field π, which is defined as:

$$\pi(x) = \frac{1}{Z} \exp(-H(x)), \ \forall x \in \mathbf{X} \tag{5.11}$$

with $Z = \sum_{z \in \mathbf{X}} \exp(-H(z))$ being a normalization constant. Now for the pairwise MRF model defined in (2.23):

$$\pi(x) = \frac{1}{Z} \prod_{s \sim t} \psi_{st}(x_s, x_t) \prod_{s \in S} \phi_s(x_s) \tag{5.12}$$

suppose it is a Gibbs field, then its energy function is given by:

$$H(x) = -\sum_{s \sim t} \ln \psi_{st}(x_s, x_t) - \sum_{s \in S} \ln \phi_s(x_s) \tag{5.13}$$

Now we would like to find out a distribution μ with marginal probabilities $b_s(x_s)$ and $b_{st}(x_s, x_t)$, $s \sim t$, such that the free engergy is minimized. Then we will have $\mu = \pi$ with $b_s(x_s)$ and $b_{st}(x_s, x_t)$ being the marginal probabilities of the target distribution π.

We are now trying to express the free energy in terms of the marginal probabilities, or say, beliefs, $b_s(x_s)$ and $b_{st}(x_s, x_t)$. There are mainly two terms in the free energy: the average energy and the entropy. We begin with the average energy, which is:

$$\begin{aligned} E(H;\mu) &= \sum_x \mu(x)H(x) \\ &= -\sum_{s\sim t}\sum_x \mu(x)\ln\psi_{st}(x_s,x_t) - \sum_{s\in S}\sum_x \mu(x)\ln\phi_s(x_s) \\ &= -\sum_{s\sim t}\sum_{x_s,x_t} b_{st}(x_s,x_t)\ln\psi_{st}(x_s,x_t) - \sum_{s\in S}\sum_{x_s} b_s(x_s)\ln\phi_s(x_s) \end{aligned} \tag{5.14}$$

The entropy cannot be expressed in terms of marginal probabilities except for very special cases. Generally we have to settle for approximations. We know that under tree graphs the joint distribution of MRFs can be expressed by marginal probabilities, as has been shown in Section 3.2.1. So under tree approximations we can represent μ explicitly in terms of beliefs:

$$\mu(x) = \frac{\prod_{s\sim t} b_{st}(x_s,x_t)}{\prod_{s\in S} b_s(x_s)^{n_s-1}} \tag{5.15}$$

where n_s is the degree (number of neighbors) of the site s. For cases where the underlying graphs have loops, (5.15) is only an approximation. We then get the Bethe approximation to the entropy [YFW02],

$$\begin{aligned} \mathcal{H}(X;\mu) &= -\sum_x \mu(x)\left(\sum_{s\sim t}\ln b_{st}(x_s,x_t) - \sum_{s\in S}(n_s-1)\ln b_s(x_s)\right) \\ &= -\sum_{s\sim t}\sum_{x_s,x_t} b_{st}(x_s,x_t)\ln b_{st}(x_s,x_t) + \sum_{s\in S}(n_s-1)\sum_{x_s} b_s(x_s)\ln b_s(x_s) \end{aligned} \tag{5.16}$$

For tree (loop-free) graphs, the calculation of the entropy is exact and minimization of the Bethe free energy with respect to the marginal beliefs leads to the exact marginal probabilities. However for the graphs with loops, this is only an approximation.

5.4 Equivalence of BP to Bethe free energy minimization

For loop-free graphs, the Bethe free energy gets minimized if and only if the beliefs are exactly the marginal probabilities of the corresponding Gibbs field. For graphs with loops, however, the beliefs are only approximate. The belief propagation algorithm is related to the minimization of the Bethe free energy. In fact, the fixed points of the belief propagation algorithm are equivalent to the local stationary points of the Bethe free energy. Recall the Bethe free energy in the following form:

$$\begin{aligned} F_{\text{Bethe}} &= -\sum_{s\sim t}\sum_{x_s,x_t} b_{st}(x_s,x_t)\ln\psi_{st}(x_s,x_t) - \sum_{s\in S}\sum_{x_s} b_s(x_s)\ln\phi_s(x_s) \\ &\quad +\sum_{s\sim t}\sum_{x_s,x_t} b_{st}(x_s,x_t)\ln b_{st}(x_s,x_t) - \sum_{s\in S}(n_s-1)\sum_{x_s} b_s(x_s)\ln b_s(x_s) \end{aligned} \tag{5.17}$$

We shall minimize the Bethe free energy F_{Bethe} with respect to the beliefs under the following constraints: $\forall s \sim t, s, t \in S, \forall x_s \in \mathbf{X}_s, \forall x_t \in \mathbf{X}_t$,

$$\left\{ \begin{array}{lcl} \sum_{x_t} b_{st}(x_s, x_t) & = & b_s(x_s) \\ \sum_{x_s} b_{st}(x_s, x_t) & = & b_t(x_t) \end{array} \right. \tag{5.18}$$

and $\forall s \in S$,

$$\sum_{x_s} b_s(x_s) = 1 \tag{5.19}$$

in order to get the stationary points of F_{Bethe}. We introduce the Lagrange multipliers $\lambda_{ts}(x_s)$ and $\lambda_{st}(x_t)$, $\forall s \sim t, s, t \in S$, $\forall x_s \in \mathbf{X}_s$, $\forall x_t \in \mathbf{X}_t$ to enforce (5.18), γ_s, $\forall s \in S$ to enforce (5.19), and construct the Lagrangian L.

The equation $\frac{\partial L}{\partial b_s(x_s)} = 0$ gives:

$$(n_s - 1)(\ln b_s(x_s) + 1) = -\ln \phi_s(x_s) + \sum_{u \in \mathcal{V}_s} \lambda_{us}(x_s) + \gamma_s \tag{5.20}$$

$\frac{\partial L}{\partial b_{st}(x_s, x_t))} = 0$ gives:

$$\ln b_{st}(x_s, x_t) = \ln \psi_{st}(x_s, x_t) + \lambda_{st}(x_t) + \lambda_{ts}(x_s) - 1 \tag{5.21}$$

These are the stationary points of the Bethe free energy in parametric forms.

Now suppose we run the BP algorithm till it converges. That is, we get the fixed point for the messages $m_{st}(x_t)$, $m_{ts}(x_s)$, $\forall s \sim t, s, t \in S, \forall x_s \in \mathbf{X}_s, \forall x_t \in \mathbf{X}_t$. Let us define

$$\left\{ \begin{array}{lcl} \lambda_{st}(x_t) & = & \ln \phi_t(x_t) + \sum_{u \in \mathcal{V}_t \setminus \{s\}} \ln m_{ut}(x_t) \\ \lambda_{ts}(x_s) & = & \ln \phi_s(x_s) + \sum_{u \in \mathcal{V}_s \setminus \{t\}} \ln m_{us}(x_s) \end{array} \right. \tag{5.22}$$

Using BP fixed-point conditions (5.3) and (5.4), it is easy to check that the constraints (5.18) and (5.19), the stationary conditions (5.20) and (5.21) are all satisfied. So the fixed point of the BP algorithm is the stationary point of the Bethe free energy function. Similarly, given the beliefs and the parameters that satisfy the stationary equations (5.20) and (5.21), and the constraints (5.18) and (5.19), we can define messages according to (5.22), and it is easy to show that the messages and beliefs must satisfy the BP fixed-point equations (5.3) and (5.4). These show the equivalence between the BP fixed points and the Bethe free energy stationary points.

The BP algorithm for graphical models with loops is not garanteed to converge. But since the BP fixed points correspond to the minimum Bethe free energy, one can simply choose to minimize the free energy. Such free energy minimizations are slower than the BP algorithm, but they are at least guaranteed to converge [WT01] [YFW02]. On the other hand, empirical exploration indicates that when BP fails to converge, it is a clue that the results from minimizing the Bethe free energy will also be quite inaccurate [YFW02].

5.5 Experiments

All experiments are made using the labeled Compaq database introduced in Section 4.3. We use 32 bins per RGB channel in the following experiments.

5.5.1 Tree hidden Markov model

Bethe tree approximation

Figure 5.4 shows outputs of Bethe THMM on several test images. The outputs of Bethe tree approximations with depths one and four are presented as well as those of the Baseline model. All the skin mask outputs are gray-level images. For the Baseline model, the gray-level is proportional to $p(Y_s = 1|x_s)$. For THMM, the gray-level is proportional to $p(Y_s = 1|x_t, t \in \mathcal{T}_k^s)$, $k = 1, 4$. The Bethe THMM easily gets messages overflowed for $k > 4$. We remark that the beliefs of THMM are almost binary, especially when deeper trees are applied. As can be seen from the images, Bethe THMM gives higher probabilities to detected skin pixels than the Baseline model does. Most skin pixels are detected. Let us look at the third column of Fig. 5.4 first. The first two images compare favorably to the outputs of the Baseline model since real skin pixels are given very strong response. However for the third image, some false alarms are also strengthened, which is not what we want to see. The output of the second image compares also favorably to that of Fig. 4.1, where the left arm of the right person is not detected. The Bethe THMM with belief propagation here detects this part correctly. The detected skin regions are also more precise than those of HMM in Fig. 4.1. With higher depths, Bethe THMM begins to lose some details of detected skin regions, and some small false skin regions existing in depth-1 tree output are removed a bit, too.

Figure 5.5 shows ROC curves of Bethe THMM and Baseline model computed from the test set. The Baseline model permit to detect 80.5% of the skin pixels with 10% of false positive rate. Bethe THMM has a uniform improvement over the Baseline model. Depth-1 Bethe THMM permits to detect 82.3% of the skin pixels for the same false positive rate. In Chap. 4, when using a Gibbs sampler to estimate the probability for skin at pixel locations, we obtained comparable results with the ones reported here, but with a running time increased by a factor of more than ten, we shall report the time issues later in this section. Deeper Bethe THMM's make the ROC curves more concentrated with little increase on the pixel classification performance.

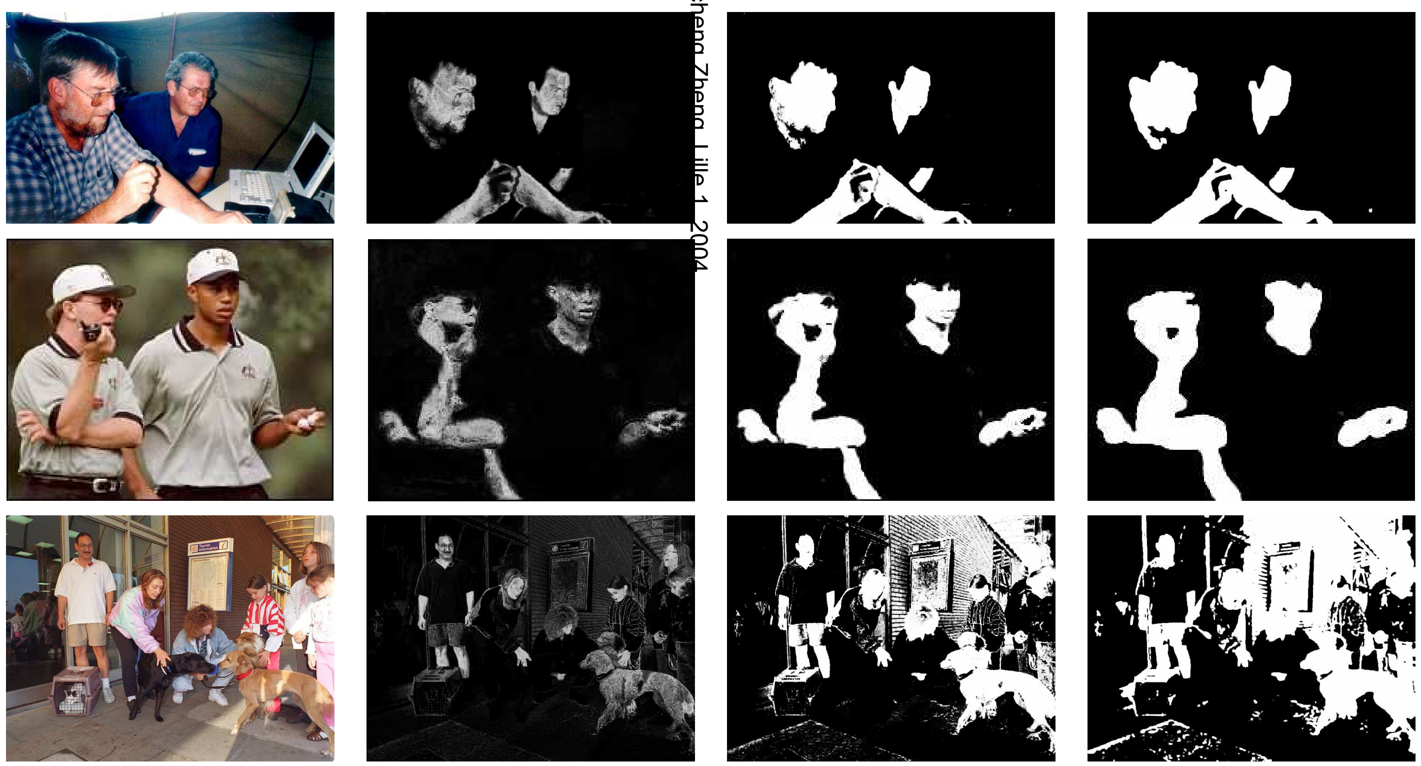


Figure 5.4: Bethe THMM outputs compared to the Baseline model outputs. **First column:** input color images. **Second column:** Baseline model. **Third column:** Bethe tree hidden Markov model, depth one. **Fourth column:** Bethe tree hidden Markov model, depth four. In the computed images, the gray level is proportional to the skin probability evaluated with the specified model. The image in the first row has 583 × 375 pixels, the one in the second row 225 × 180 pixels, the one in the third row 800 × 599 pixels

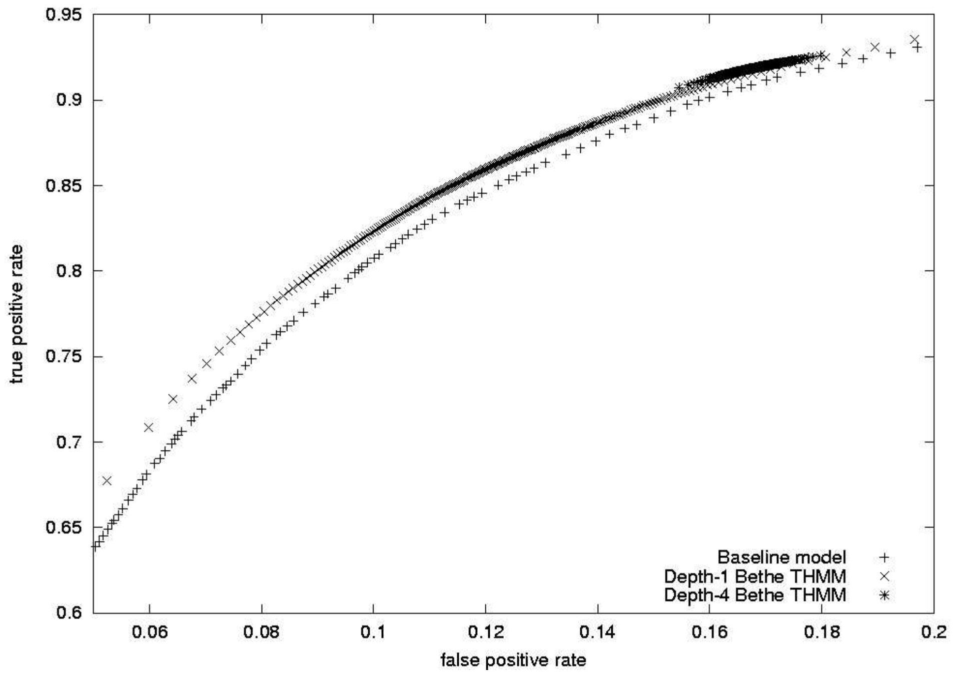


Figure 5.5: Bethe THMM ROC curves compared with that of the Baseline model

4-star tree approximation

Figure 5.6 shows outputs of 4-star THMM on several test images. The depth-1 4-star tree approximation is equivalent to the depth-1 Bethe tree approximation. One can simply refer to Fig. 5.4 for depth-1 4-star tree results. For the sake of comparison, the outputs of depth-4 and depth-8 4-star THMM as well as those of the depth-4 Bethe THMM are presented in Fig. 5.6. All the skin mask outputs are gray-level images as in Fig. 5.4. As Bethe THMM, the beliefs of 4-star THMM are also almost binary, especially when deeper trees are applied. The 4-star THMM can run on deeper trees than the Bethe THMM since the number of nodes is only linear to the depths in 4-star trees while exponential in Bethe trees. Comparing the third column to the second column, we can see that false alarms are a little less in 4-star THMM than Bethe THMM. For example, the pixels on the monitor in the first image are less labeled as skin by 4-star THMM. And in the second image, Bethe THMM labels some pixels on the T-shirt of the right person as skin. But these false alarms are removed by 4-star THMM. The skin regions output by depth-4 4-star THMM are also a little more precise than those output by Bethe THMM of the same depth. 4-star THMM of depth eight outputs less detailed skin regions than that of depth four for given examples. The comparison is clearer by ROC curves which give the quantitative evaluation of the models.

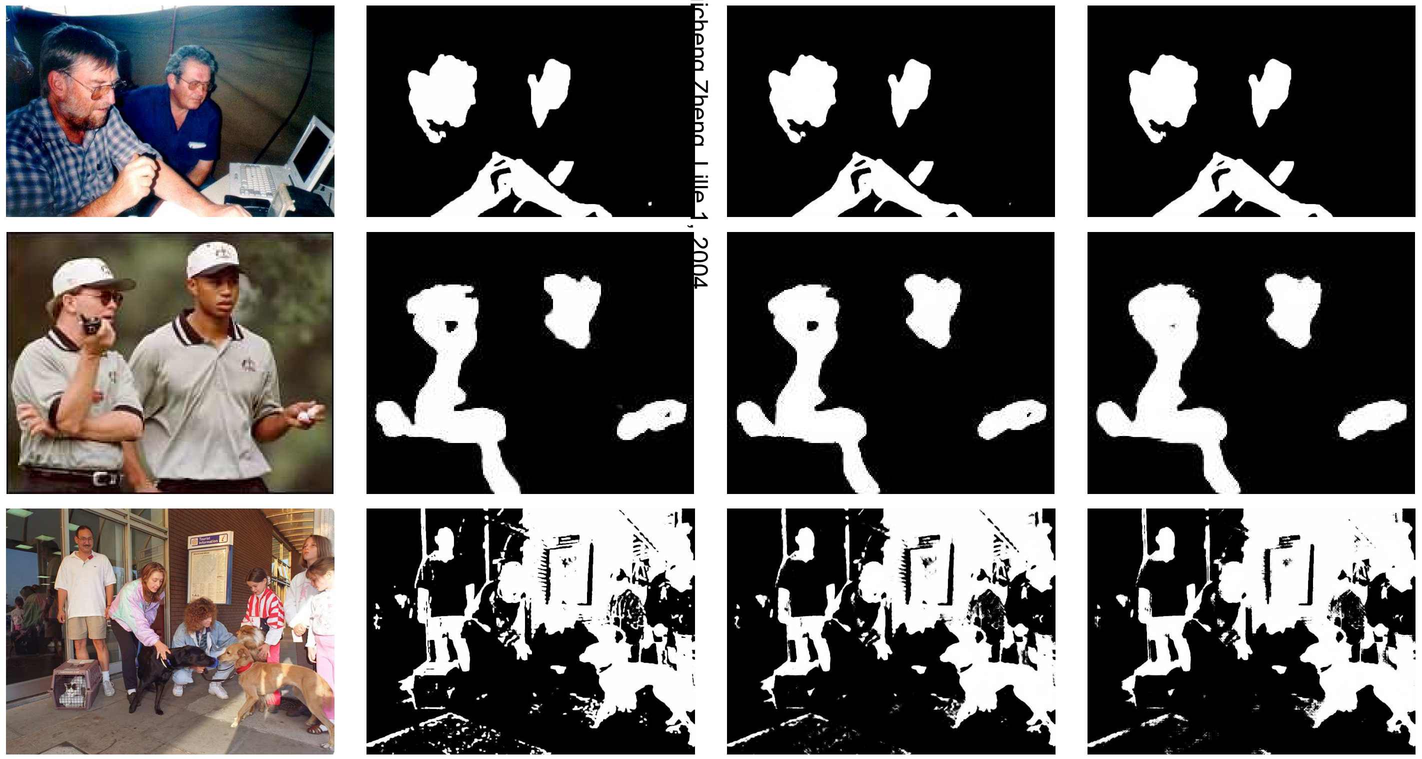


Figure 5.6: 4-star THMM outputs compared to Bethe THMM outputs. **First column:** input color images. **Second column:** Bethe tree hidden Markov model, depth four. **Third column:** 4-star tree hidden Markov model, depth four. **Fourth column:** 4-star tree hidden Markov model, depth eight. In the computed images, the gray level is proportional to the skin probability evaluated with the specified model. The image in the first row has 583×375 pixels, the one in the second row 225×180 pixels, the one in the third row 800×599 pixels

Figure 5.7 shows ROC curves of Bethe and 4-star THMM computed from the test set. The ROC curve of depth-4 Bethe THMM is more concentrated than that of depth-4 and depth-8 4-star THMM maybe due to more nodes included and highly overlaped in Bethe trees (remark that there are 161 nodes in a depth-4 Bethe tree while only 17/33 nodes in a depth-4/8 4-star tree). Apparently depth-8 4-star THMM has a better ROC curve than depth-4 Bethe THMM. At a first look, the superiority of much simpler 4-star tree approximations to Bethe tree approximations is somehow amazing. But taking into account the fact that more and more nodes get overlapped with the expansion of Bethe trees, this might be understandable.

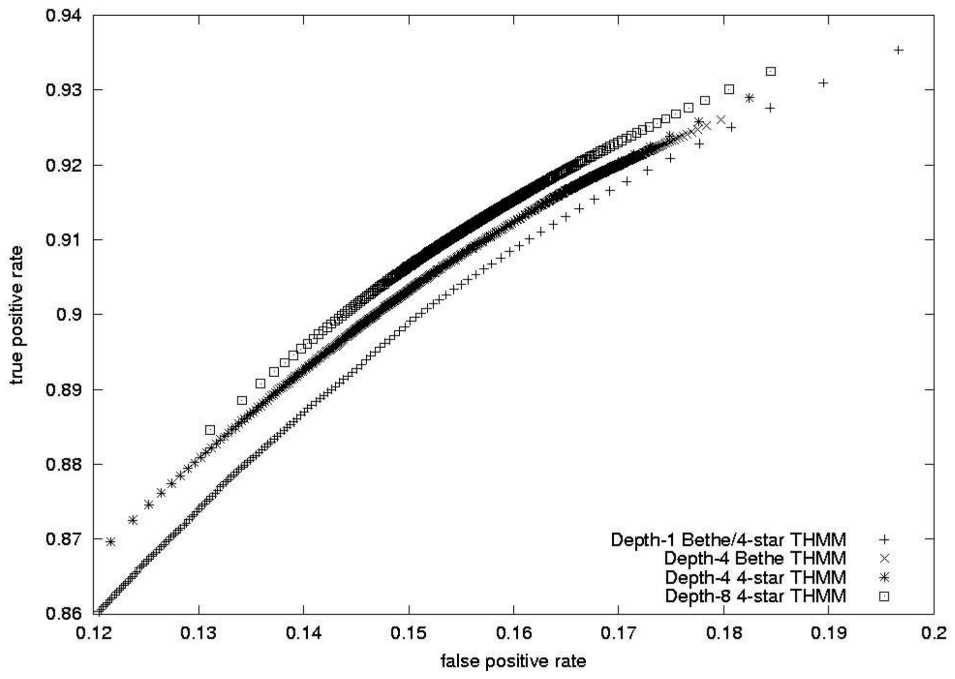


Figure 5.7: Bethe/4-star THMM ROC curves

All the experiments are performed on a PC with a Pentium 4 processor at 1.7 Ghz and 256 MB memory. The execution time for the depth-1 Bethe THMM is about 1 second per image. It is about 0.29 second per image for the depth-1 4-star THMM. The execution time of both models increases almost linearly with the depth of the THMM.

5.5.2 Tree first-order model

Results for depth-1 Bethe/4-star TFOM1 and TFOM2 are presented in Fig. 5.8. Note that depth-1 Bethe tree is equivalent to depth-1 4-star tree. For the sake of comparison, the outputs of depth-1 THMM are also put together. More iterations do not improve the performance of TFOM.

The reason could be that the tree graphs built for the classification are only approximations to the real pixel lattice, and even the pairwise MRFs are only approximations to the true distributions. We do not know the exact distribution $p(x, y)$ of the color images and their labels. All that we have done is to try to find models on tree graphs to approximate the underlying true distribution $p(x, y)$ based on the observed features. We have been trying to mimic some aspects of the ground-truth distribution by building all the MRF and tree graphs. When some models work, we can say that model seems to grab the true properties. When other models fail, it might be that the true distribution does not expose those properties. The TFOM outputs are not so binary as their THMM counterparts. At a first glance they look more like the outputs of the Baseline model. The shapes of the detected skin regions are more precise than THMM. If we compare the TFOM outputs to the Baseline model outputs in Fig. 5.4, we will find that the former model detects the glasses racks of the people in the first image, while the latter model failed to do that. The left arm of the right person in the second image is also assigned higher probablities by TFOM than by the Baseline model. When comparing TFOM2 (fourth column) to TFOM1 (third column), we will see that more skin pixels are assigned high skin probabilities by TFOM2 than by TFOM1. This is obvious in the first and second images.

We show the bulk results in Fig. 5.9. Only ROC curves of depth-1 Bethe/4-star TFOM are computed in this figure. We can see from the figure that TFOM2 performs uniformly better than TFOM1. For example, TFOM2 can detect 83.1% of the skin pixels when 10% of false positive rate is allowed. While TFOM1 can only detect 81.5% of skin pixels at the same false positive rate. The reason could be that the approximation (3.17) made in TFOM1 is too empirical and not as good as the more natural maximum entropy estimation made in TFOM2. The performance of TFOM2 also compares favorably to that of THMM which permits to detect 82.3% of the skin pixels at the false positive rate of 10% for depth-1 Bethe/4-star trees.

The elapsed time is about 0.66 second per image for depth-1 4-star TFOM1, and about 0.70 second per image for depth-1 4-star TFOM2.

5.6 Discussion

Instead of temporally expensive Gibbs sampling strategy, we used BP algorithm for probability inference at pixel locations. This algorithm is well expressed with graphical models. Thus we began with the introduction of general graphical models. Of particular interests is the pairwise MRF. We showed that the distribution that we were tring to find is the Gibbs field, which is the distribution with the least free energy under certain energy function. For pairwise MRF on

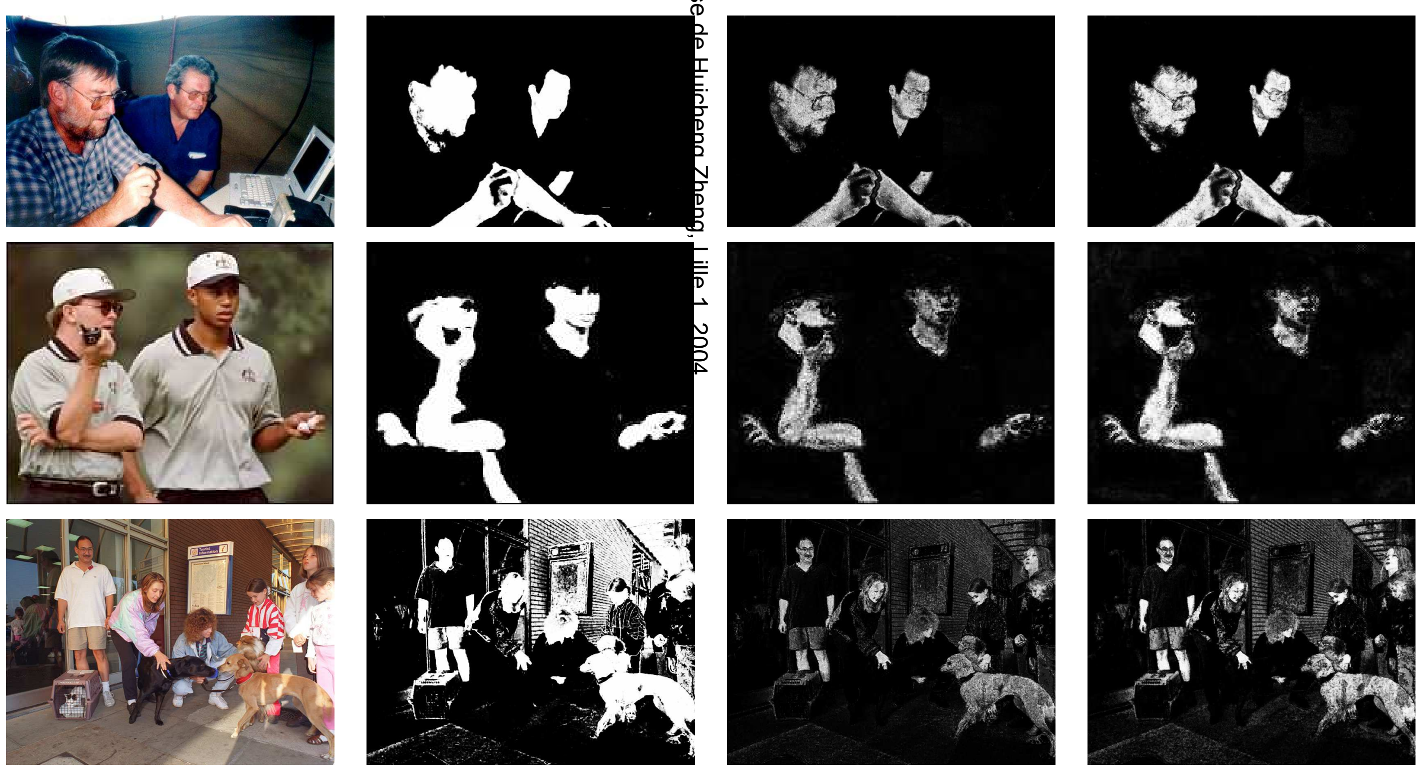


Figure 5.8: TFOM outputs compared to THMM outputs. **First column:** input color images. **Second column:** tree hidden Markov model. **Third column:** tree first-order model 1. **Fourth column:** tree first-order model 2. In the computed images, the gray level is proportional to the skin probability evaluated with the specified model. Bethe/4-star trees of depth one are used in all experiments. The image in the first row has 583 $\times$ 375 pixels, the one in the second row 225 $\times$ 180 pixels, the one in the third row 800 $\times$ 599 pixels

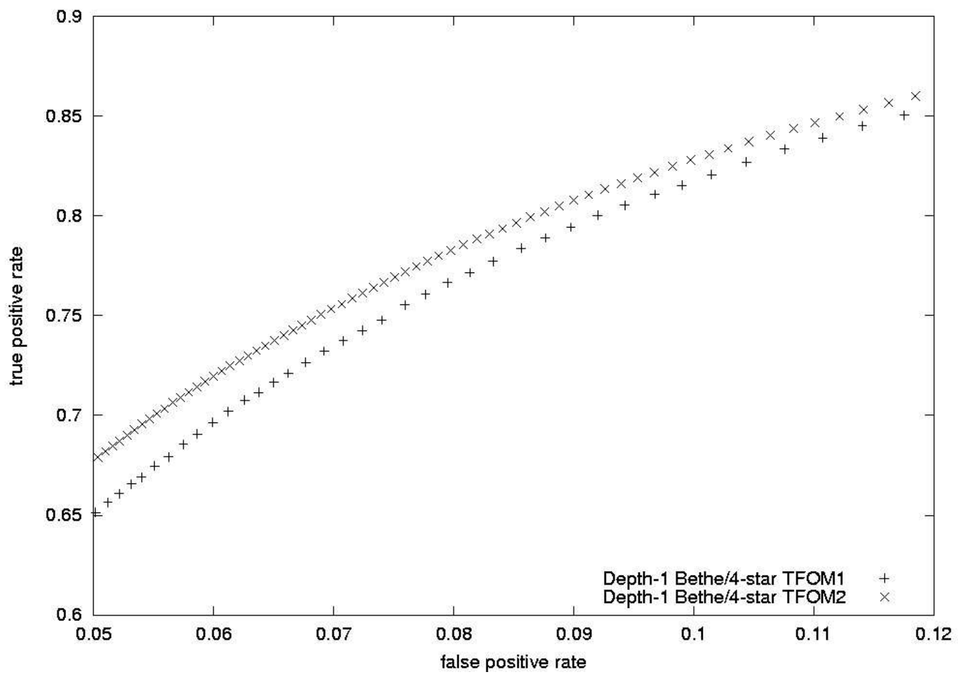


Figure 5.9: ROC curves of TFOM1 and TFOM2

tree graphs the free energy has special pairwise form in terms of marginal probabilities on single sites and pairs of sites. The free energy is called Bethe free energy in this case. For graphs with loops, however, the Bethe free energy is only an approximation of the true free energy. We showed that the fixed points of the BP algorithm are the stationary points of the Bethe free energy. Two sorts of tree approximations to pixel lattice, Bethe tree approximation and 4-star tree approximation, are explored. We showed that the BP algorithm dramatically improved the speed of skin probability inference at pixel locations, while it still exhibited comparable pixel classification performance to that of Gibbs sampling.

Résumé (Chapitre 6)

Un algorithme de détection de la peau peut être appliqué pour détecter ou reconnaître un humain dans une image. Dans ce chapitre, nous présentons une application de l'algorithme de détection de la peau pour détecter les images à caractère pornographique. Le résultat de l'algorithme de détection de la peau est une image constituée, en chaque pixel, de la probabilité pour que ce pixel soit un pixel de peau. Dans un premier temps, nous calculons des descripteurs de formes pour cette image. Dans un second temps, nous utilisons un perceptron à deux couches pour construire une décision — image à caractère pornographique ou non — à partir des valeurs des descripteurs de forme.

Nos expériences menée sur une large collection d'images provenant de l'internet confirment l'importance de la bonne détection de la peau pour détecter les images pornographiques.

Nous présentons dans ce chapitre le projet POESIA qui signifie — Public Open-source Environment for a Safer Internet Access — un logiciel public et libre pour un acces plus sûr à l'internet. Notre système de détection des images pornographiques est utilisé afin de filtrer les images. Les essais conduits par les utilisateurs ont montré que les performances peuvent être améliorées en combinant les résultats de plusieurs images apparaissant sur une même page.

Chapter 6

Blocking adult images

Images are an essential part of today's World Wide Web. The statistics of more than 4 million HTML webpages reveal that 70.1% of webpages contain images and that on average there are about 18.8 images per HTML webpage [SDT+02]. These images are mostly used to make attractive Web contents or to add graphical items to mostly textual content, such as navigational arrows.

However, images are also contributing to harmful (e.g. pornographic) or even illegal (e.g. paedophilic) Internet content. So effective filtering of images is of paramount importance in an Internet filtering solution.

To block adult content, some representative companies as NetNanny and SurfWatch, operate by maintaining lists of URL's and newsgroups and require constant manual updating. Abundant literature is available, but the Internet is very rapidly evolving, not only quantitatively. Each day, 3 million pages are appearing on the Web. Detection based on image content analysis has the advantage to process equally all the images without the need for updating, so will produce more effective filtering.

By taking advantage of the fact that there is a strong correlation between images with large patches of skin and adult images, the work on blocking adult images will benefit from skin detection. There is already a large amount of work on this track.

The WIPE [WLWF98b] system developed by Wang, Li, Wiederhold and Firschein uses a manually-specified color histogram model as a prefilter in an analysis pipeline. Input images whose average probability of skin is low are accepted as non-offensive. Images that contain considerable skin pass on to a final stage of analysis where they are classified using wavelet features. The algorithm uses a combination of Daubechies wavelets, normalized central moments, and color histograms to provide semantically-meaningful feature vector matching.

Forsyth's [FFB96] research group has designed and implemented an algorithm to screen images of naked people. Their algorithms involve a skin filter and human figure grouper. The skin color model used by Fleck, Forsyth and Bregler consists of a manually specified region in a log-opponent color space. Detected regions of skin pixels form the input to a geometric filter based on skeletal structure. As indicated in their paper, 52.2% true positive rate and 3.4% false positive rate have been obtained for a test set of 138 images with naked people and 1401 assorted benign images. However, it takes about 6 minutes on a workstation for the figure grouper in their algorithm to process a suspect image passed by the skin filter.

Jones and Rehg [JR02] propose techniques for skin color detection by estimating the distribution of skin and non-skin color in the color space using labeled training data. To detect adult images, some simple features are extracted. The discrimination performance based solely on skin is rather good for such simple features.

Bosson et al. [BCCH02] propose a pornography detection system which is integrated in a commercial system. This system is also based on skin detection. They compared the generalised linear model, the k-nearest neighbor classifier, the multi-layer perception (MLP) classifier and the support vector machine and found that the MLP gives the best classification performance.

Our approach is based on the TFOM skin detection. The output of skin detection is a grayscale *skin map* with the gray levels being proportional to the skin probabilities. We extract some simple features from the skin map which compose a feature vector. We train a MLP classifier on $5,084$ patterns from the training set. In the test phase, the MLP classifier takes a quick decision on the input pattern in one pass.

6.1 Feature extraction

There are propositions for high-level features based on grouping of skin regions [FFB96] that might distinguish adult images from those not, but here we have a requirement to process the images speedily. The project associated with our work, Poesia [SDT$^+$02], works under real environments. It requires real-time processing of all kinds of media, including images and text. So, along with [JR02] [WLWF98b], we are interested to try simpler features.

Since skin distribution is of the paramount importance for the detection of adult images [JR02] [BCCH02], all our current features are based on the skin map. For the sake of practicality, the features should be simple and easy to calculate.

We first binarize the skin map by simple thresholding. We then implement morphological open/close operations to remove noise and connect broken regions. Small skin regions are considered insignificant and discarded. Many of our features are based on the fit ellipses calculated on the skin map, since they could meet our requirement for simplicity and capture some important shape information. For a detailed account of the formulae used to calculate fit ellipses for gray-scale regions, see Appendix E. We observed from experiments that for approaches based on skin detection, portraits have a tendence to be detected as adult images since generally portraits expose plenty of skin as adult ones. The fit ellipses will hopefully at least help discriminate portraits from adult images. We will calculate two fit ellipses for each skin map—the Global Fit Ellipse (GFE) and the Local Fit Ellipse (LFE). The GFE is computed on the whole skin map, while the LFE only on the largest skin region in the skin map. The GFE and LFE capture most of the skin distributions in the whole image and in the largest skin region respectively. Figure 6.1 shows the GFE as well as the LFE for a skin map.

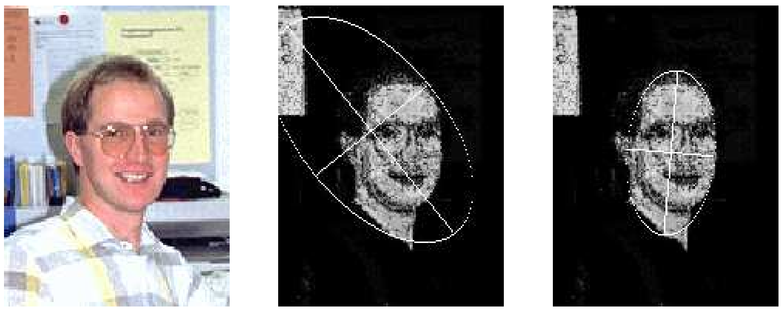


Figure 6.1: First: the original input image. Second: the Global Fit Ellipse (GFE) on the skin map. Third: the Local Fit Ellipse (LFE) on the skin map

With the skin map, we extract 9 features from the input image. The first 3 features are global:

- the average skin probability of the whole image
- the average skin probability inside the GFE
- number of skin regions in the image

The other 6 features are computed on the largest skin region of the input image.

- distance from the centroid of the largest skin region to the center of the image
- angle of the major axis of the LFE from the horizontal axis

- ratio of the minor axis to the major axis of the LFE
- ratio of the area of the LFE to that of the image
- average skin probability inside the LFE
- average skin probability outside the LFE

All these features compose a simple feature vector. No effort was done to find the correlation between features.

6.2 Pattern Recognition

The feature extraction steps described in the previous subsection produce a feature vector for each image. The task is then to find the decision rule on this feature vector that optimally separates adult images from those not.

Evidence from [BCCH02] shows that the MLP classifier offers a statistically significant performance over several other approaches such as the generalized linear model, the k-nearest neighbor classifier and the support vector machine.

The semilinear feedforward net as reported by Rumelhart, Hinton, and Williams has been found to be an effective system for learning discriminants for patterns from a body of examples [Pao89]. The multi-layer perceptron used in this work is introduced in detail in [Pao89]. For the sake of integrity, this network is introduced briefly here with the specific information of our application.

This network is composed of an input layer, indexed by i, a hidden layer, indexed by j, and an output layer, indexed by k. There are 20 nodes at the hidden layer, 1 node at the output layer since we need only one indication for the adult content. The number of hidden nodes is chosen empirically. Each node at the hidden layer and the output layer is associated with an activation function, which is a sigmoïdal function that operates on the corresponding net input of that node. The links between the input layer and the hidden layer are associated with weights denoted by w_{ji}, links between the hidden layer and the output layer with weights denoted by w_{kj}. The learning procedure starts off with a random set of weight values. In the feedforward procedure, each training pattern p is fed into the network to evaluate the output o_p. In the backpropagation procedure, the error at the output E_p necessitates changes $\Delta_p w_{kj}$ in the weights between the hidden layer and the output layer. This procedure goes on backward, the net calculates $\Delta_p w_{ji}$ for all the w_{ji} between the input layer and the hidden layer in the net for the pattern p based on $\Delta_p w_{kj}$. This procedure is repeated for all the patterns in the training set to yield the resulting Δw_{kj} and Δw_{ji} for all the weights for that one presentation. The corrections to the weights are then made. The training patterns are then fed into the network again and the outputs are again evaluated in a feedforward manner. Weight changes are again evaluated using the backpropagation procedure. We iterate the feedforward and backpropagation procedures for the training set of patterns. In a successful learning exercise, the system error will decrease with the number of iterations, and the procedure will converge to a stable set of weights, which will exhibit only small fluctuations in value as further learning is attempted [Pao89].

For an input pattern p, the output of this net o_p is a real number between 0 and 1. The nearer the number is to 1, the more possibly the input pattern corresponds to an adult image. We then set a threshold T, $0 < T < 1$, to get the binary decision. In the test phase, the net takes a quick decision on the input pattern in one pass.

6.3 Experiments

All experiments are made using the following protocol. The database contains 10,168 photographs, which are imported from the Compaq database [JR02] and the Poesia database [SDT$^+$02]. It is split into two equal parts randomly, with 1,297 adult photographs and 3,787 other photographs in each part. Then these two parts are used as the training set and the test set respectively. In Fig. 6.2 we show some examples of the results for adult images and non adult images from the test set. The outputs of the MLP are shown just below the corresponding images.

There are some cases where our detector does not work well. In Fig. 6.3 several such examples are presented. The first adult image is not detected since the skin appears almost white due to over-exposure. We see that most of the skin is not detected on the skin map. The second adult image contains two connected big frames. The LFE of this image will then be very big, and the average skin probability inside this LFE will be very small. The third image is benign, but it is detected adult since the toy dog takes a skin-like color and the average skin probabilities inside the GFE and the LFE are very high. The fourth image is a portrait but it is detected as an adult image since it exposes a lot of skin and even the hair and the clothes take skin-like colors. We believe skin detection based solely on color information cannot do much more, so maybe some other sorts of information is needed to improve the adult image detection performance. For example, some kind of face detector could be implemented to improve the results. Moreover, generally adult images in web pages tend to appear together, and are surrounded by text, which could be an important clue for the adult content detector.

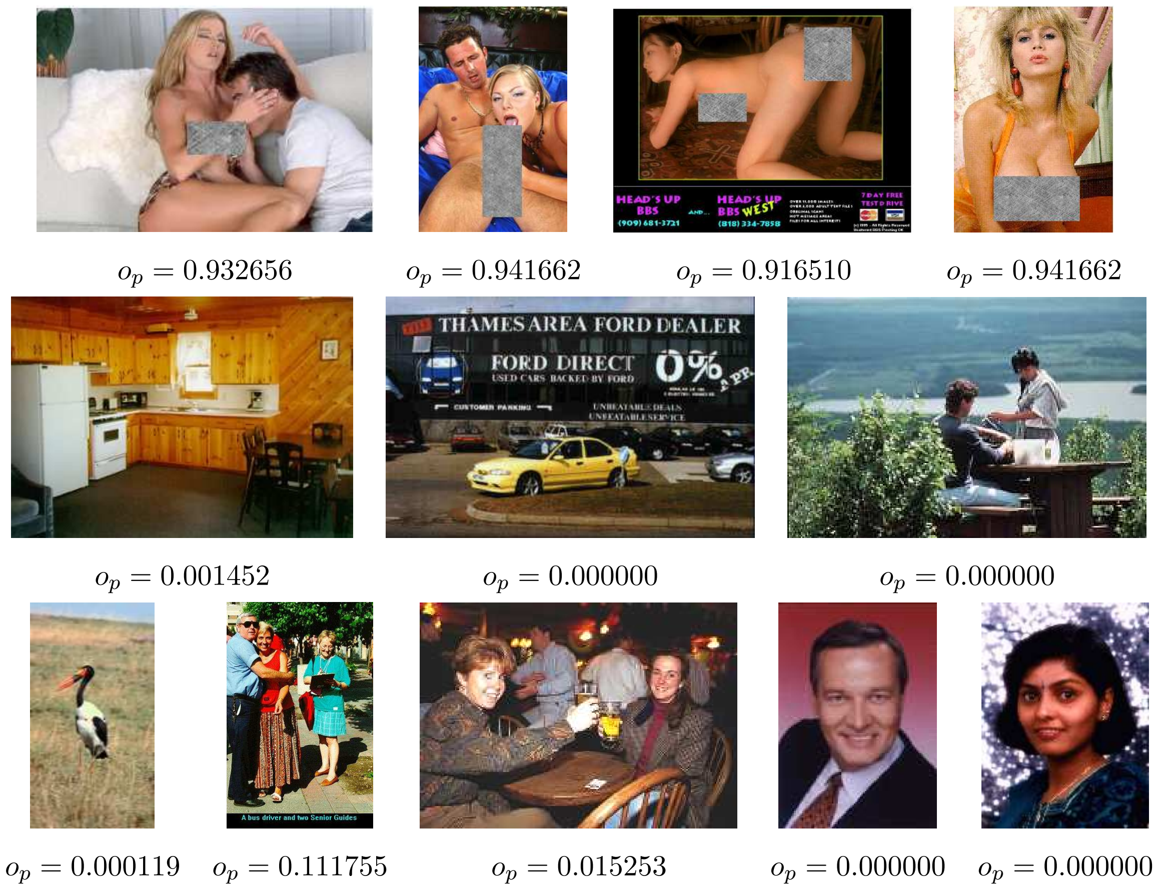
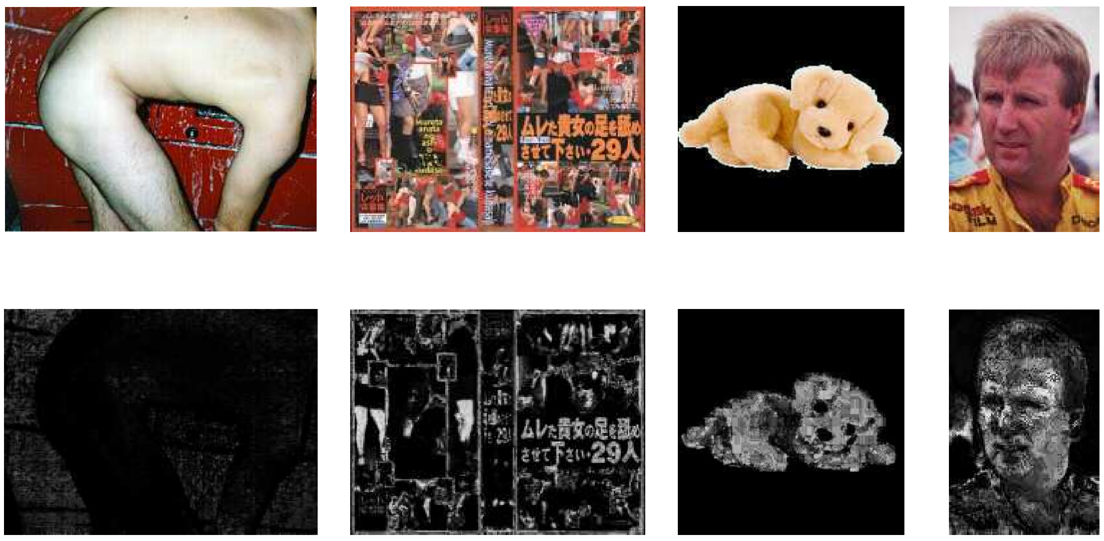


Figure 6.2: First row: Experimental results on adult images. Second and third rows: Experimental results on non adult images. Below the images are the associated outputs of the MLP



$o_p = 0.006828$ $o_p = 0.000005$ $o_p = 0.899044$ $o_p = 0.938251$

Figure 6.3: First row: original images. Second row: the corresponding skin maps. Below the skin maps are the corresponding outputs of the MLP. The first two columns are adult images with low responses, while the other two columns are benign images with high responses

By varying the threshold T, a ROC curve is achieved as shown in Fig. 6.4. The elapse time is about 1.51×10^{-5}second/pixel, i.e., about 1 second for a 256×256 image.

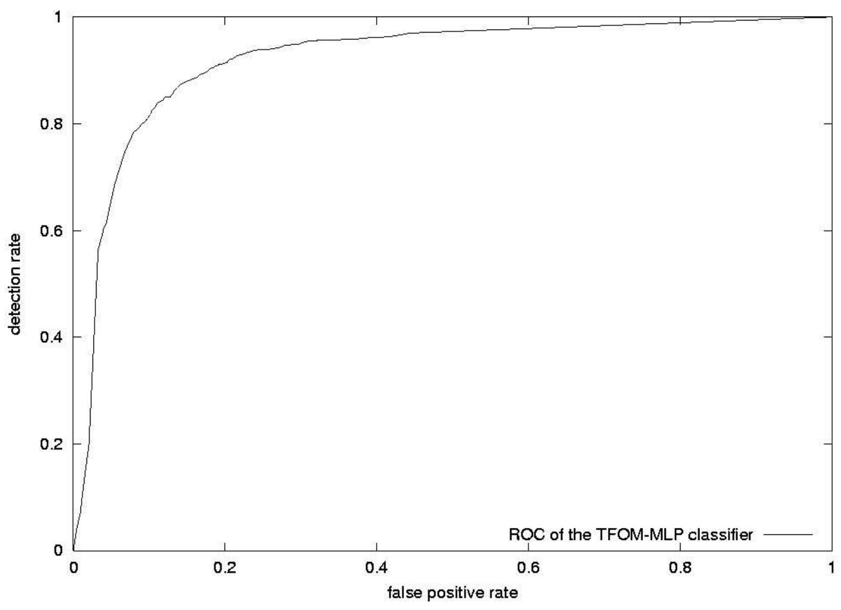


Figure 6.4: ROC curve of the TFOM-MLP adult image detector

6.4 Poesia system—content-based Internet filtering

The adult image detector developed in the former sections has been incorporated into the Poesia system [SDT+02]. The *Poesia* project is funded by the European Commission and takes place in the Information Society and Technology Safer Internet Action Plan [1]. The Safer Internet Action Plan (IAP) promotes safety on the Internet by tackling the controversial issue of illegal, harmful and racist content. Poesia means a Public Open-source Environment for a Safer Internet Access.

Poesia seeks to develop, test, evaluate and promote a fully open-source, and extensible, state of the art, filtering and catching software solution. It filters harmful content in several channels (Web, Email, News) combining innovative technologies to achieve more effective filtering than existing products. The filtered categories of contents include pornography, racism, violence and gross language. Filtering covers a range of modes, including image filtering, natural language text filtering, URL (Universal Resource Locator), PICS (Platform for Internet Content Selection) and JavaScript filtering. The filter was initially deployed in English, Italian and Spanish. To cover other European languages additional work is required and should be effectively possible

[1]http://www.saferinternet.org/filtering/poesia.asp

because of the open-source model and the portability of the technical solutions. Poesia aims to become the standard filtering solution deployed by educational institutions and for home use.

Figure 6.5 shows the architecture of the Poesia system. For Web content filtering, HTTP (HyperText Transfer Protocol) is processed by Shweby proxy server, which is an open source software with support of ICAP (Internet Content Adaptation Protocol). So the Poesia system has to interface through the ICAP protocol to Shweby. The Poesia system is organized around a central monitor, which deals with external information sources (web, email, ...) and pre-processes and distributes the information to be filtered to specialized filters (eg for NLP (Natural Language Processing), images) and decisors. For Email filtering, the Poesia system contains an external Exim mail filter which delegates the filtering to the Poesia monitor. The monitor communicates with Shweby (using ICAP), with Exim via a Poesia exim filter (using a specific poesia mail protocol), with other information sources (NNTP (Network News Transfer Protocol), IRC (Internet Relay Chat), ...) and with specialized filters (NLP in spanish, english, french; language detectors; image filters; Javascript and URL filters; ...) and decisors. Filters and decisors communicate directy only with the Poesia monitor.

The Poesia monitor starts, schedules and communicates with Poesia specific filters (e.g. image or NLP filters) and subprocesses (like the decision mechanism). It communicates with these filters with a specific protocol. Filters do not communicate directly with each other or with decision mechanisms. All such communication goes to the monitor. The monitor is able to continue processing from one filter to another. A given filter can produce a temporary result which is redirected to other filter(s).

The Poesia filtering system has been tested by the end user group. A set of pages and images were acquired using a Spider to randomly sample the WWW. The spider traverses links within identified sites to retrieve pages at varying depths. The final corpus consists of approximately $20,000$ pages ($10,000$ non-pornographic pages, $10,000$ pornographic pages). The end users tested the individual image filter as well as the whole Poesia system on the test corpora. As a conclusion, the incorporation of filters in Poesia generally improves the performance of the individual image filter. For the image test corpus, the true positive rate of pornographic web pages is raised from 0.952 to 0.972, while the false positive rate is decreased from 0.106 to 0.054. This is basically due to the fact that when testing the image filter individually, a single image was used whilst in the Poesia system the image filter produces its classification based on multiple images in a page, thus the increased information improved its accuracy.

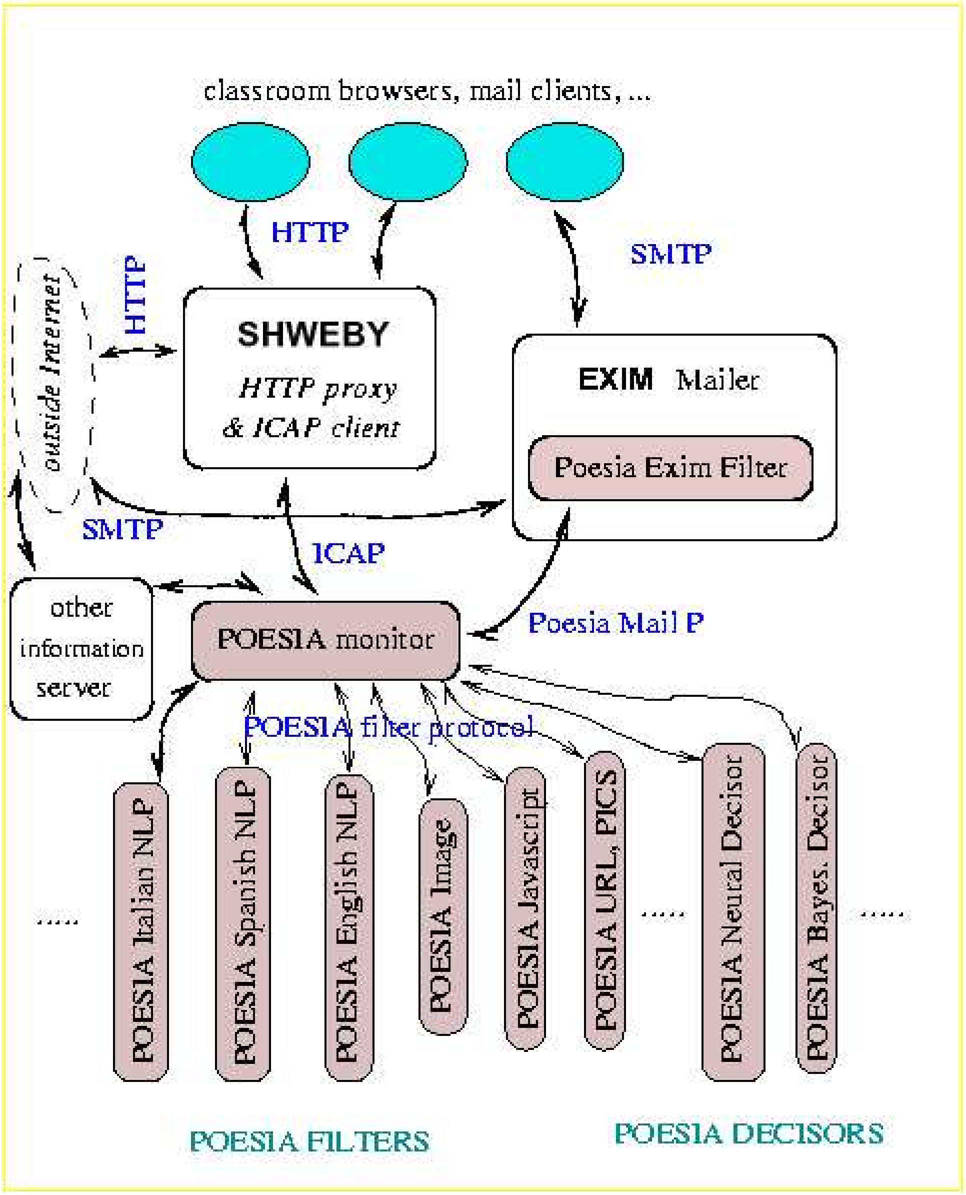


Figure 6.5: Poesia overall architecture [SDT+02]

6.5 Discussion

We applied our skin detector in this chapter to adult image detection. The general use of adult image detection is Internet content inspection. Thus it is generally required to transact the images as quickly as possible. So the implemented skin detection adopted the BP algorithm. We extract several simple features from the skin map, which is a gray-scale skin probability image. We trained a two-layer perceptron on these features. We did a lot of experiments on a large image database, which showed stimulating performance despite the simplicity of the features we extracted. In this chapter we also explained Poesia—a Public Open-source Environment for a Safer Internet Access. Poesia adopts the previously introduced adult image detector in the image filtering module. It also incorporates some other advanced state-of-the-art techniques, e.g. NLP, Javascript filtering, etc., to filtering the potentially harmful web pages or content from Internet. It provides the platform for us to show that the combination of several images on the same web page improved the overall pornographic web page detection considerably.

Résumé (Chapitre 7)

Dans ce chapitre, nous donnerons un résumé de notre travail. Les principales contributions sont les suivantes :

1. Nous appliquons la méthode du maximum d'entropie sur la moyenne (MaxEnt) pour la détection de la peau dans les images en couleur. C'est, à notre connaissance, la première étude de la sorte. Nous construisons ainsi le "modèle indépendant" ainsi que deux modèles Markoviens. Nos résultats expérimentaux démontrent les accroissements de performances qui résultent de l'utilisation de ces nouveaux modèles par rapport au modèle indépendant.

2. Nous démontrons l'unification des différents modèles à travers l'approche MaxEnt. Nous démontrons d'autre part que cette approche permet de construire des modèles qui ont des bonnes performances empiriques.

3. Pour l'estimation des paramètres, ainsi que pour l'inférence, nous approximons, localement, le graphe des pixels, contenant de nombreuses boucles, par un arbre. Ceci donne lieu à une variété de modèles, chacun indexé par un type d'arbre. Nous montrons que l'algorithme de "belief propagation" permet alors d'effectuer l'inférence de manière particulièrement efficace.

4. Nous utilisons notre détecteur de peau pour détecter les images pornographiques. Nous obtenons des résultats très encourageants.

5. Les meilleurs algorithmes issus de cette thèse sont incorporés dans le logiciel public et libre POESIA de filtrage de l'internet.

Même si nous avons fait de notre mieux pour découvrir des algorithmes performants pour la détection de la peau et pour le filtrage de l'internet, il reste de nombreuses pistes et questions ouvertes pour de futures recherches. Ces recherches sont utiles car la détection de la peau est une étape fondamentale pour de nombreuse applications.

Nous donnons dans ce chapitre les pistes de recherches suivantes :

1. Des contraintes différentes pour le maximum d'entropie sur la moyenne qui mèneront à de nouveaux modèles Markoviens. Nous illustrerons à l'aide de modèles graphiques.

2. Nous suggérons d'implémenter d'autres représentations des couleurs que le traditionel RGB (rouge, vert, bleu) afin de réduire la complexité des modèles.

3. Nous avons, dans cette thèse effectué des expériences avec les arbres de Bethes et les arbres étoilés. Nous proposons de considérer les arbres couvrants ainsi que les quadtrees.

4. Une alternative prometteuse à l'algorithme de Belief Propagation dans le cas de graphes contenant des boucles est le Generalized Belief Propagation.

5. Pour mieux détecter les images pornographiques, il pourrait être utile de détecter les visages et/ou les membres (bras, jambes). Nous présenterons aussi des applications possibles de notre détecteur de peau.

Chapter 7

Conclusions and perspectives

7.1 Conclusions

The goal of skin modeling is to build a decision rule which seperates skin pixels from non-skin ones. Skin detection plays an important role in various applications such as face detection, searching and filtering image content on the web, video segmentation and face/head tracking, ... However, skin detection is not an easy task, especially in the case of web images. First, skin colors may vary from person to person. Furthermore, since web images are captured under various conditions, they are subject to all kinds of noise and distortion.

In this thesis, we have shown that the nowadays popular MaxEnt method can lead to efficient models for skin detection, which is a supervised image segmentation problem. The Baseline model has been developed by Jones and Rehg by simply assuming pixel independence. However we showed that under certain constraints, we can derive this model through the maximum entropy framework. The Baseline model is in fact a MaxEnt model that puts constraints on one-site marginal probabilities. This gives us more insight into the well-known simple statistical skin model. Performance, measured by the ROC curve on the Compaq database is impressive for such a simple model. The observation of the image database tells us that the skin pixels are not totally independent. Instead, they tend to appear together and form some regions. A natural step would be to model the distribution with MRFs. Again we showed that the constructed MRF models, the HMM as well as the FOM, are both MaxEnt models under respective constraints. Thus, we witnessed that the independent model as well as the MRF models are all unified into the maximum entropy framework. The HMM includes constraints that force smoothness of the label image, while the FOM put constraints on colors and labels of neighboring pixels.

Except the Baseline model, which can be solved analytically, all the models that we built are in parametric forms. We tackled parameter estimation in Chapter 3. We explored a method to estimate the parameters of the HMM based on equations of local characteristics of the MRF. This method is effective for MRF models that do not have many unknown parameters. For the FOM, which has a lot of unknown parameters, we approximate the pixel lattice with tree graphs. With tree approximations, the FOM is expressed in terms of empirical marginal probabilities on neighboring pixels. Direct estimation of these empirical probabilities from the training set will cause over-fitting since the orginal parameter space is huge for ordinary images. Two methods are explored to reduce dimensionality. The first method is to decrease the dimensionality by assuming conditional independence of one-site color and two-site color gradient. The resulting model is TFOM1. The other method is to estimate the high-dimensional parameters from some low-dimensional histograms of the training set with MaxEnt. This model is TFOM2. In estimating parameters for TFOM2, we devised a variant of the generalized iterative scaling algorithm by rearranging calculation, which accelarates the parameter estimation process drastically.

In order to compute skin probabilities at pixel locations for given color images with the constructed MaxEnt models, we first implemented a stochastic sampling scheme, Gibbs sampling. Experiments revealed that HMM performs better than the Baseline model. The performance is once more improved when we replace the HMM with the TFOM. However, Gibbs sampling scheme is quite time consuming and not suitable for real-time applications, such as Internet filtering. Remark that all the MRF models built in our context are pairwise MRFs. By approximating the pixel lattice with tree graphs, we can compute the marginal probabilities efficiently with the BP algorithm. This algorithm is closely related to the minimization of the Bethe approximation of the free energy. We explored two sorts of tree approximations to the pixel lattice, the Bethe tree approximation and the 4-star tree approximation. The experimental results seem to favor the latter a bit. Experiments showed that BP scheme is much faster than Gibbs sampling scheme with comparable performance and it matches the real-time requirement.

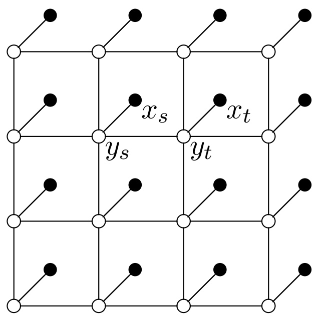
In Chapter 6, we exhibited an application of the TFOM-BP skin detection scheme to adult image filtering of Internet. We extracted several simple features from the skin map. We trained a two-layer perceptron on these features. The experiments on a large image database showed stimulating performance despite the simplicity of the features based only on skin maps, which provides an evidence that skin detection plays a very important role in adult image detection. In this chapter we also introduced Poesia, a project associated with much of our research. The experiments conducted by end users of the Poesia project show that combination of images in one web page improves the performance of adult image detection for Internet filtering.

7.2 Perspectives

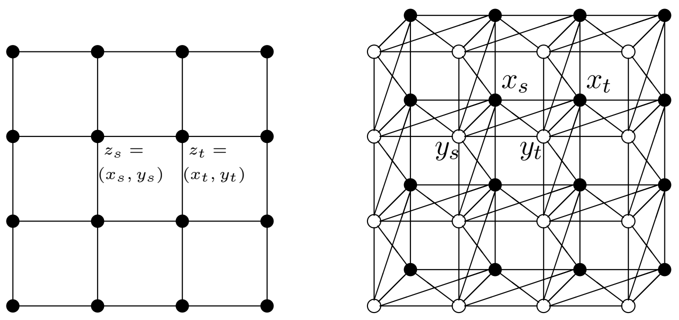
Skin detection is of paramount importance for a variety of applications related to computer vision and pattern recognition. In this section, we would like to give some suggestions for furture research on this subject and the related models and methods.

New constraints

The HMM has two sets of potentials $\psi_{st}(y_s, y_t)$ and $\phi_s(x_s, y_s)$. $\psi_{st}(y_s, y_t)$ force smoothness of the label image and $\phi_s(x_s, y_s)$ force compatability between the observed image x and the label image y. The graphical model for HMM is then Fig. 7.1(a). In FOM, we go further to include the full compatability between (x_s, x_t, y_s, y_t) for $s \sim t \in S \times S$. Letting $z = (x, y)$, the graphical model of FOM is then the left one of Fig. 7.1(b). The introduced new variable z hides the full interaction between x and y. The potentials $\psi_{st}(x_s, x_t, y_s, y_t)$ imply cliques on four nodes x_s, x_t, y_s and y_t. So the full graphical model expanded on x and y would be the right one of Fig. 7.1(b), which seems complex already. Parameter estimation for FOM is hard since $\psi_{st}(x_s, x_t, y_s, y_t)$ involve a huge parameter space. Most of the difficulty comes from the product subspace of x_s and x_t. We then have to fall back on some approximations such as color gradient conditional independence in TFOM1 or introducing new variables $d_{st} = x_t - x_s$ and establishing connections between corresponding nodes in TFOM2.



(a) HMM



(b) FOM

Figure 7.1: Graphical models of HMM and FOM

Our feeling is that the compatabilities between x_s and x_t are not as important as those between y_s and y_t, since generally image x is given, the structure is already implied. Out of this motivation, we can remove the links between x_s and x_t, which gives us the model in Fig.7.2.

Each maximal clique for the model in Fig. 7.2 contain three nodes x_s, y_s and y_t, or x_t, y_s and

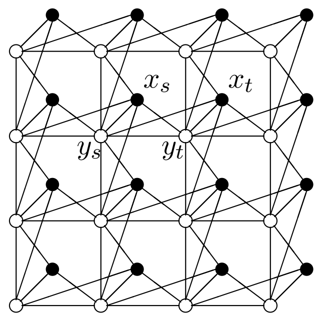


Figure 7.2: New graphical model

y_t, for $s \sim t \in S \times S$, which implies potentials on (x_s, y_s, y_t) and (x_t, y_s, y_t). In order to obtain such a model through MaxEnt, we can apply the following constraints:

$$\forall s \sim t \in S \times S, \forall x_s \in C, \forall x_t \in C, \forall y_s \in \{0,1\}, \forall y_t \in \{0,1\},$$

$$\begin{cases} p(x_s, y_s, y_t) & = & q(x_s, y_s, y_t) \\ p(x_t, y_s, y_t) & = & q(x_t, y_s, y_t) \end{cases} \tag{7.1}$$

Then the solution is:

$$p(x, y) \propto \exp \sum_{s \sim t} (\lambda(s, t, x_s, y_s, y_t) + \lambda(s, t, x_t, y_s, y_t)) \tag{7.2}$$

with $\lambda(s, t, x_s, y_s, y_t)$ and $\lambda(s, t, x_t, y_s, y_t)$ being the parameters to be estimated subject to the constraints (7.1). It is not hard to show that the model in (7.2) fulfills the following conditional independence:

$$p(x|y) = \prod_{s \in S} p(x_s | y_t, t \in \{s\} \cup \mathcal{V}_s) \tag{7.3}$$

Since $p(x, y) = p(x|y)p(y)$, we need only to estimate parameters for $p(y)$, which should be easier to do.

Other color spaces for the TFOM

The main difficulty that we face for TFOM is the high dimensionality of $q(x_s, x_t, y_s, y_t)$ in the RGB color space. Supposing 32 bins per color channel, there are $32^3 \times 32^3 \times 2 \times 2 = 4.29 \times 10^9$ bins in the histogram. But there are only 6.74×10^8 training pixels in the Compaq database. Thus direct empirical estimation of the parameters $q(x_s, x_t, y_s, y_t)$ will cause drastic over-fitting. That is why we reduce the dimension by assuming conditional independence of color gradients with color on one site, or by estimating the parameters with maximum entropy principle as introduced in Section 3.3.2. We could also try other means to reduce the dimension. One possibility is to replace RGB color space with normalized RGB space or HSV space. In this way we can drop one channel of the color space to have an approximated two-dimensional color space. Still supposing 32 bins per color channel, $q(x_s, x_t, y_s, y_t)$ will then require to fill a histogram of $32^2 \times 32^2 \times 2 \times 2 = 4.19 \times 10^6$, which is tractable now.

Spanning tree approximations to the pixel lattice

In this thesis we have experimented with Bethe tree approximation and 4-star tree approximation. As we have shown, Bethe tree approximation involves a considerable number of overlaped vertices when the trees grow. There are 41 real pixels out of 161 tree nodes in a Bethe tree of depth-4, that is, there are $161-41 = 120$ nodes which are only repeated pixels. On the other hand, 4-star trees are so simple that they ignore many pixels that could have important interaction with the root nodes. For example, the 4-star tree of depth-2 in Fig. 7.3 ignores the pixels a, b, c and d, which may have more influence on s than u. When this tree grows, such ignorance may make it far from the real pixel lattice.

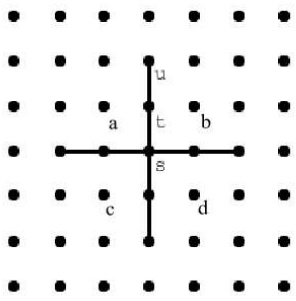


Figure 7.3: A 4-star tree of depth-2. It does not include the pixels a, b, c and d, which may be more important to s than the pixel u

Instead of Bethe trees or 4-star trees, we may consider tree approximations such that all the pixels in the lattice are contained without overlap, which are so-called *spanning trees*. For a pixel lattice of 3×3, we show some examples of spanning trees in Fig. 7.4. They are local tree approximations to the 3×3 pixel lattice centered at s.

Remark that each tree has exactly 8 edges. In fact, it is not hard to prove that a tree with n nodes must have $n-1$ edges based on the iterative nature of trees. On the contrary, a connected graph with n nodes and $n-1$ edges must be a tree. Therefore it is equivalent to find spanning

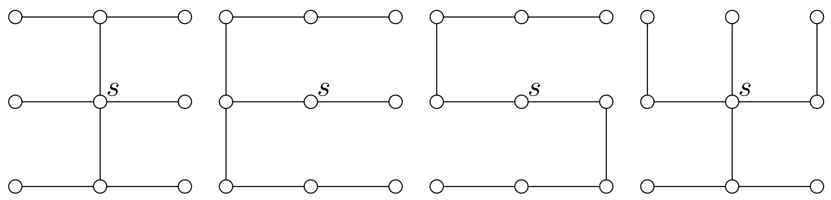


Figure 7.4: Spanning tree approximations to the 3×3 pixel lattice centered at s

trees on an $n \times n$ pixel lattice or to find the connected graphs with $n \times n - 1$ edges on the pixel lattice. Apparently there are so many possibilities even for a small pixel lattice. But we can restrain the solution to those spanning trees with certain properties. For example, we can limit the spanning trees to those invariant to rotations of $90°$ and $180°$. We may also assign weights to edges in the spanning trees, which increase when the distances from the edges to the root site get larger. We can then focus on those spanning trees with the minimum total weights. In fact, this opens the connection to the minimal spanning tree research. To eliminate the subjective factors in choosing spanning tree approximations, we can even average the outputs of different spanning trees to get the final result. In building such a mixture model of trees, the particular bias of individual trees will hopefully be removed.

Quadtree models

All the tree models considered by now are on the same plane. Laferté et al. [LPH00] proposed MRF on the quadtree for image classification. The construction of their tree models are illustrated in Fig. 7.5. The set of nodes of the tree is notated as S, and its *root* is referred to as site r. Any node s different from r has a unique *parent* node denoted s^- with "-" recalling the decrease of resolution. Conversely, the set of the four *children* of any non-leaf node s is denoted s^+. A *descendant* of s is a node t such that s belongs to the unique chain that joins t to the root r. The set of descendants of s and s is denoted as $\mathrm{d}(s)$. The nodes belonging to the same "generation" n from the root form the n-th resolutional level S^n of the tree. The coarsest level reduces to the root node: $S^0 = \{r\}$. The finest level is S^N for some positive integer N. The authors applied the quadtree models for image classification problem on both synthetic and natural images. Their conclusion is that such hierarchical tree-based models dramatically reduce the computational load, and improve the estimation quality. Our feeling is that such multiresolutional tree models will give us the possibility to include the influence of farther pixels into the tree models efficiently. For example, for the TFOM, we can attach the variable set (x_s, y_s) at node s of the quadtree model.

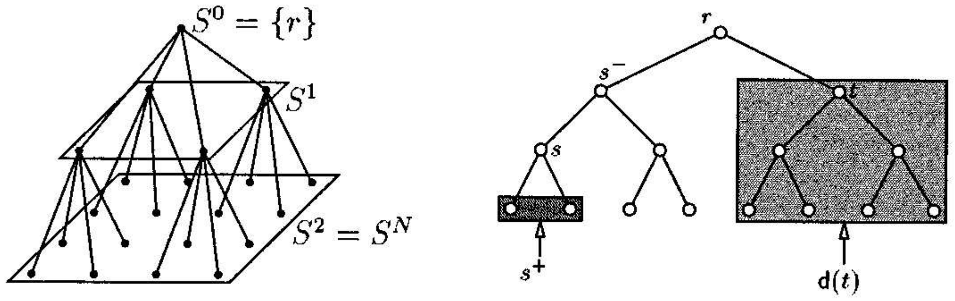


Figure 7.5: The construction of the quadtree and notations. The resolution increases from S^0 to S^N. Here is an example of three resolutions, $N = 2$. The right graph shows a dyadic tree for ease of illustration [LPH00]

Generalized belief propagation

In the BP algorithm that we implemented in this thesis, all messages are from single nodes to single nodes. It is natural to expect that messages from groups of nodes to other groups of nodes could be more informative, and thus lead to better inference. That is the basic intuitive idea behind generalized belief propagation (GBP) [YFW02]. Yedidia et al. claimed that their experiments showed GBP algorithms can significantly out-perform ordinary BP on two-dimensional pairwise MRFs, and for decoding error-correcting codes, especially when the graphical models under consideration have short loops [YFW02].

Applications of skin detection

The application to adult image detection for Internet filtering, which we show in the thesis, is based on very simple features extracted from only the skin map. In fact we could also design higher level features. One possibility is to design a face detector, then most of the false alarms caused by portraits could be eliminated since generally adult images expose considerable skin regions other than faces. We could also follow the idea of Fleck [FFB96] to design some kind of geometrical analysor of limbs to catch higher-level features of people in photographs. But we should also take care that such schemes may take considerable running time.

We can also implement skin detection to face detection, face/head tracking or video segmentation in the future work.

Publications

Book chapter

H. Zheng, M. Daoudi, C. Tombelle and C. Djeraba. Adult image filtering for internet safety. *Multimedia Security Handbook.* B. Fuhrt and D. Kirouski Eds, CRC Press, 2004.

International Journals

H. Zheng, M. Daoudi and B. Jedynak. Blocking adult images based on statistical skin detection. *Electronic Letters on Computer Vision and Image Analysis*, Vol. 4, No. 2, pages 1–14, 2004.

B. Jedynak, H.Zheng and M. Daoudi. Improving skin detection with higher order models. *under review, Image and Vision Computing*, Elsevier.

International conferences and workshops

H. Zheng, H. Liu and M. Daoudi. Blocking objectionable images: adult images and harmful sysbols. *Proceedings of IEEE International Conference on Multimedia and Expo*, Taipei, Taiwan, June $27^{th} - 30^{th}$, 2004.

H. Zheng, M. Daoudi and B. Jedynak. Adult image filtering for web safety. In *Proceedings of* 2^{nd} *International Symposium on Image/Video Communications over fixed and mobile networks*, pages 77–80, Brest, France, July $7^{th} - 9^{th}$, 2004.

B. Jedynak, H. Zheng and M. Daoudi. Statistical models for skin detection. *IEEE Workshop on Statistical Analysis in Computer Vision*, in conjunction with CVPR'2003, Madison, Wisconsin, June $16^{th} - 22^{nd}$, 2003.

B. Jedynak, H. Zheng and M. Daoudi. Maximum entropy models for skin detection. *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, Lisbon, Portugal, July 7th–9th, 2003. Published in *Lecture Notes in Computer Science*, Vol. 2683, pages 180–193, Springer-Verlag, Heidelberg, 2003.

National conferences and workshops

H. Zheng, M. Daoudi and B. Jedynak. From maximum entropy to belief propagation: applied to skin detection. *British Machine Vision Conference*, London, United Kingdom, September 7th – 9th, 2004.

H. Zheng, M. Daoudi and B. Jedynak. Adult image detection using statistical model. *Compression et Représentation des Signaux Audiovisuels*, Lille, France, May 25th – 26th, 2004.

B. Jedynak, H. Zheng, M. Daoudi and D. Barret. Maximum entropy models for skin detection. *Proceedings of Indian Conference on Computer Vision, Graphics and Image Processing*, December 2002.

Technical reports

B. Jedynak, H. Zheng, M. Daoudi and D. Barret. Maximum entropy models for skin detection. *Technical report, PUB.IRMA*, Lille, Vol. 57, Number XIII, 2002.

B. Starynkevitch, M. Daoudi, C. Tombelle, H. Zheng et al. POESIA software architecture definition document. *Technical report*, Deliverable 3.1, DEcember 2002. http://www.poesia-filter.org/pdf/Deliverable_3_1.pdf.

Bibliography

[Ash65] R. Ash. *Information Theory.* Interscience Publ., New York, 1965.

[ATD01] A. Albiol, L. Torres, and E.J. Delp. Optimum color spaces for skin detection. In *Proc. of the International Conference on Image Processing*, volume 1, pages 122–124, Tessaloniki, Greece, 2001.

[BCCH02] A. Bosson, G.C. Cawley, Y. Chan, and R. Harvey. Non-retrieval: blocking pornographic images. proceedings of the intl. conf. on image and video retrieval,london, uk, 2002. *Lecture Notes in Computer Science*, 2383:50–60, 2002.

[BCL01] D. Brown, I. Craw, and J. Lewthwaite. A som based approach to skin detection with application in real time systems. In *Proc. of the British Machine Vision Conference*, volume 2, pages 491–500, 2001.

[Bes86] Julian Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society, B*, 48(3):259–302, 1986.

[BGT93] C. Berrou, A. Glavieux, and P. Thitimajshima. Near shannon limit error-correcting coding and decoding: turbo-codes. In *Proc. of IEEE International Conference on Communications*, pages 1064–1070, Geneva, Switzerland, 1993.

[Bir98] S. Birchfield. Elliptical head tracking using intensity gradients and color histograms. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 232–237, Santa Barbara, CA, 1998.

[BM00] J. Brand and J.S. Mason. A comparative assessment of three approaches to pixel-level human skin-detection. In *Proceedings. 15th International Conference on Pattern Recognition*, volume 1, pages 1056–1059, Barcelona, Spain, September 2000.

[BPP96] A. Berger, S. Della Pietra, and V. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, 1996.

[CB00] D. Chai and A. Bouzerdoum. A bayesian approach to skin color classification in ycbcr color space. In *Proc. IEEE Region Ten Conference(TENCON'2000)*, volume 2, pages 421–424, 2000.

[CFP02] G. Celeux, F. Forbes, and N. Peyrard. Em procedures using mean field-like approximations for markov model-based image segmentation. *Pattern Recognition*, 36(1):131–144, January 2002.

[CJ83] G. Cross and A.K. Jain. Markov random field texture models. *IEEE Trans. on PAMI*, 5(1):25–39, 1983.

[CJSW01] H.D. Cheng, X.H. Jiang, Y. Sun, and J. Wang. Color image segmentation: advances and prospects. *Pattern Recognition*, 34(12):2259–2281, December 2001.

[CT91] T. M. Cover and J. A. Thomas. *Elements of Information Theory.* Wiley-Interscience, August 1991.

[CWY95] Q. Chen, H. Wu, and M. Yachida. Face detection by fuzzy pattern matching. In *Proc. IEEE International Conference on Computer Vision*, pages 591–596, 1995.

[DF00] Fabio Divino and Arnoldo Frigessi. Penalized pseudolikelihood inference in spatial interaction models with covariates. *Scandinavian Journal of Statistics*, 27(3):445–458, 2000.

[DR72] J.N. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. *Annals of Mathematical Statistics*, 43:1470–1480, 1972.

[FFB96] M.M. Fleck, D.A. Forsyth, and C. Bregler. Finding naked people. In *Proc. European Conf. on Computer Vision*, pages 593–602. B. Buxton, R. Cipolla, Springer-Verlag, Berlin, Germany, 1996.

[FPC00] W.T. Freeman, E.C. Pasztor, and O. T. Carmichael. Learning low-level vision. *International Journal of Computer Vision*, 40(1):25–47, 2000.

[Gal63] R.G. Gallager. *Low-density parity check codes.* MIT Press, 1963.

[GG84] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Trans. on PAMI*, 6:721–741, 1984.

[GM02] G. Gomez and E. Morales. Automatic feature construction and a simple rule induction algorithm for skin detection. In *Proc. of the ICML Workshop on Machine Learning in Computer Vision*, pages 31–38, 2002.

[Gom02] G. Gomez. On selecting colour components for skin detection. In *Proceedings. 16th International Conference on Pattern Recognition*, volume 2, pages 961–964, 2002.

[GS85] S. Guiasu and A. Shenitzer. The principle of maximum entropy. *The Mathematical Intelligencer*, 7(1):43–48, 1985.

[HAMJ02] R-L. Hsu, M. Abdel-Mottaleb, and A.K. Jain. Face detection in color images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):696–706, May 2002.

[HS92] R.M. Haralick and L.G. Shapiro. *Computer and Robot Vision*, volume 1. Addison-Wesley, 1992.

[Jay] E. Jaynes. Probablity theory: The logic of science. http://omega.albany.edu:8008/JaynesBook. chapter 11.

[Jay57a] E. T. Jaynes. Information theory and statistical mechanics i. *Phiys. Rev.*, 106:620–630, 1957.

[Jay57b] E. T. Jaynes. Information theory and statistical mechanics ii. *Phiys. Rev.*, 108:171–190, 1957.

[JPCSV99] L. Jordao, M. Perrone, J.P. Costeira, and J. Santos-Victor. Active face and feature tracking. In *Proceedings of International Conference on Image Analysis and Processing*, pages 572–576, Venice, Italy, September 1999.

[JR98] Michael J. Jones and James M. Rehg. Statistical color models with application to skin detection. Technical Report CRL 98/11, Compaq, 1998.

[JR02] M.J. Jones and J. M. Rehg. Statistical color models with application to skin detection. *International Journal of Computer Vision*, 46(1):81–96, January 2002.

[KFL01] F.R. Kschischang, B.J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, Feb. 2001.

[KPS03] J. Kovač, P. Peer, and F. Solina. Human skin colour clustering for face detection. In B. Zajc, editor, *International Conference on Computer as a Tool, EUROCON*, Ljubljana, Slovenia, September 2003.

[LPH00] J-M. Laferté, P. Pérez, and F. Heitz. Discrete markov image modeling and inference on the quadtree. *IEEE Trans. on Image Processing*, 9(3):390–404, March 2000.

[LY02] J.Y. Lee and S.I. Yoo. An elliptical boundary model for skin color detection. In *Proc. International Conference on Imaging Science, Systems and Technology*, Las Vegas, USA, June 2002.

[Mit] MitOpenCourseWare. Lecture 15: Gaussian channel. http://ocw.mit.edu/OcwWeb/Electrical-Engineering-and-Computer-Science/6-441Transmission-of-InformationSpring2003/LectureNotes/index.htm.

[ML79] A. Martin-Lof. The equivalence of ensembles and gibbs'phase rule for classical lattice-systems. *Journal of Statistical Physics*, 20:557–569, 1979.

[OBP96] N. Oliver, F. Berard, and A. Pentland. Lafter: Lips and face tracker. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition*, pages 123–129, Puerto Rico, 1996.

[Pao89] Y.-H. Pao. *Adaptive Pattern Recognition and Neural Networks*, pages 121–129. Reading. Addison-Wesley, Massachusetts, 1989.

[Pea88] J. Pearl. *Probabilistic Reasoning in intelligent systems: networks of plausible inference.* Morgan Kaufmann, 1988.

[Poy97] C.A. Poynton. Frequently asked questions about color. http://www.poynton.com/PDFs/ColorFAQ.pdf, 1997.

[Sch03] Rob Schapire. Foundations of machine learning. http://www.cs.princeton.edu/courses/archive/spring03/cs511/scribe_notes/0410.pdf, April 2003.

[SDT$^+$02] B. Starynkevitch, M. Daoudi, C. Tombelle, H. Zheng, and et al. Poesia software architecture definition document. Technical Report Deliverable 3.1, POESIA consortium, December 2002. http://www.poesia-filter.org/pdf/Deliverable_3_1.pdf.

[Sel85] R. W. Sellens. A prediction of the droplet size and velocity distribution in a spray from first principles. Ma.sc. thesis, University of Waterloo, 1985.

[Sha48] C. E. Shannon. A methematical theory of communication. *The Bell System Technical Journal*, 27:379–423,623–656, July, October 1948. Reprinted in C. E. Shannon and Weaver, *The mathematical theory of communication*, University of Illinois Press, Urbana, 1949.

[SMHL00] M. Soriano, B. Martinkauppi, S. Huovinen, and M. Laaksonen. Skin detection in video under changing illumination conditions. In *Proc. Computer Vision and Pattern Recognition*, volume 1, pages 839–842, 2000.

[Smi78] A.R. Smith. Color gamut transform pairs. In *Proc. of the 5th Annual Conf. on Computer Graphics and Interactive Techniques*, pages 12–19, August 1978.

[SSA04] L. Sigal, S. Sclaroff, and V. Athitsos. Skin color-based video segmentation under time-varying illumination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(7):862–877, July 2004.

[ST98] E. Saber and A. Tekalp. Frontal-view face detection and facial feature extraction using color, shape and symmetry based cost functions. *Pattern Recognition Letters*, 19(8):669–680, 1998.

[Tan] Inder Jeet Taneja. *Generalized Information Measures and Their Applications.* To be published. http://www.mtm.ufsc.br/∼taneja/book/book.html.

[Tan81] R.M. Tanner. A recursive approach to low complexity codes. *IEEE Trans. Information Theory*, 27:533–547, Sept. 1981.

[TDA98] Jean-Christophe Terrillon, Martin David, and Shigeru Akamatsu. Automatic detection of human faces in natural scene images by use of a skin color model and of invariant moments. In *IEEE Third International Conference on Automatic Face and gesture Recognition*, pages 112–117, 1998.

[TSFA00] Jean-Christophe Terrillon, M. N. Shirazi, H. Fukamachi, and S. Akamatsu. Comparative performance of different skin chrominance models and chrominance spaces for the automatic detection of human faces in color images. In *Fourth International Conference On Automatic Face and gesture Recognition*, pages 54–61, 2000.

[VSA03] V. Vezhnevets, V. Sazonov, and A. Andreeva. A survey on pixel-based skin color detection techniques. In *Proc. 13th International Conference on the Computer Graphics and Vision*, pages 85–92, Moscow, Russia, September 2003.

[WD95] Chihsin Wu and Peter C. Doerschuk. Tree approximations to markov random fields. *IEEE Trans. on PAMI*, 17(4):391–402, April 1995.

[Wika] Wikipedia. Hsv color space. http://en.wikipedia.org/wiki/HSV_color_space.

[Wikb] Wikipedia. Information entropy. http://en.wikipedia.org/wiki/Information_entropy.

[Win95] G. Winkler. *Image Analysis, Random Fields and Dynamic Monte Carlo Methods.* Springer-Verlag, 1995.

[Win03] G. Winkler. *Image Analysis, Random Fields and Markov Chain Monte Carlo Methods.* Springer-Verlag, 2nd edition, 2003.

[WLWF98a] James Z. Wang, Jia Li, Gio Wiederhold, and Oscar Firschein. Classifying objectionable websites based on image content. *Notes in Computer Science, Special issue on iteractive distributed multimedia systems and telecommunication services*, 21/15:113–124, 1998.

[WLWF98b] James Ze Wang, Jia Li, Gio Wiederhold, and Oscar Firschein. System for screening objectionable images. *Images, Computer Communications Journal*, 21(15):1355–1360, 1998.

[WT01] M. Welling and Y.W. Teh. Belief optimization for binary networks: A stable alternative to loopy belief propagation. In *Proc. of the 17th Conference in Uncertainty in Artificial Intelligence*, pages 554–561, Seattle, Washington, USA, Aug. 2001.

[WZL00] Y.N. Wu, S.C. Zhu, and X.W. Liu. Equivalence of julesz ensemble and frame models. *International Journal of Computer Vision*, 38(3):247–265, July 2000.

[YA98] M. Yang and N. Ahuja. Detecting human faces in color images. In *Proceedings of the International Conference on Image Processing*, pages 127–130, Chicago, IL, 1998.

[YFW02] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Understanding belief propagation and it's generalisations. Technical Report TR-2001-22, Mitsubishi Research Laboratories, January 2002.

[You98] Laurent Younes. Estimation and annealing for gibbsian fields. *Annales de l'Institut Henry Poincaré, Section B, Calcul des Probabilités et Statistique*, 24:269–294, 1998.

[Zha92] J. Zhang. The mean field theory in em procedure for markov random fields. *IEEE Trans. on Signal Processing*, 40(10):2570–2583, October 1992.

[ZSQ99] B.D. Zarit, B.J. Super, and F.K.H. Quek. Comparison of five color models in skin pixel classification. In *Proceedings. International Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems*, pages 58–63, Corfu, Greece, September 1999.

[ZWM98] S.C. Zhu, Yingnian Wu, and David Mumford. Filters, random fields and maximum entropy (frame): towards a unified theory for texture modeling. *International Journal of Computer Vision*, 27(2):107–126, 1998.

Appendix A

The Compaq image database

The compaq image database used in our work is acquired from the authors in [JR02]. The authors obtained a large set of image files from a parallel crawl of the world wide web from multiple starting points. This image set was pruned by manually removing all files that were not photos. The authors sampled randomly from this large set and got a smaller dataset.

The dataset contains $13,562$ photos. Each photo in this dataset was processed in the following manner: The photo was examined to determine if it contained skin. If no skin was present, it was placed in the non-skin group. If it contained skin, regions of skin pixels were manually labeled using the software tool shown in Fig. A.1. This tool allows a user to interactively segment regions of skin by controlling a connected-components algorithm. Clicking on a pixel establishes it as a seed for region growing. The threshold slider controls the Euclidean distance in RGB space around the seed that defines the skin region. By clicking on different points in the photo and adjusting the slider, regions of skin with fairly complex shapes can be segmented quickly. In labeling skin the authors excluded the eyes, hair, and mouth opening. The result is a binary mask which is stored along with each photo that identifies its skin pixels.

However, we should mention that it is difficult to get a perfect segmentation of the skin in any given image. Some photos contain skin patches of such a small size (e.g. crowd scenes) that segmentation was problematic. Even in photos with large regions of skin it is often hard to precisely define their boundaries (e.g. on the forehead where skin is obscured by hair) [JR02].

Figure A.2 shows some samples from the Compaq image database. There are images with skin and their labeled mask images as well as images without skin pixel.

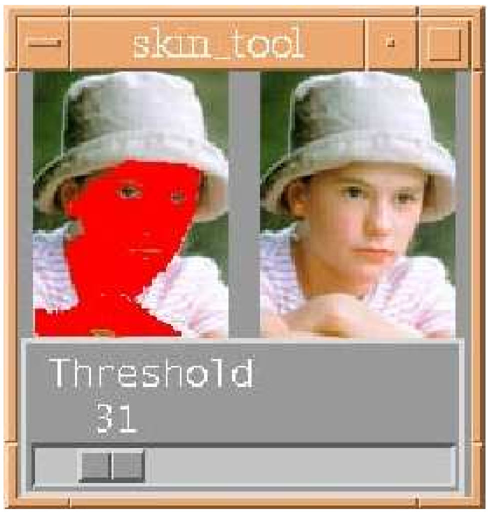
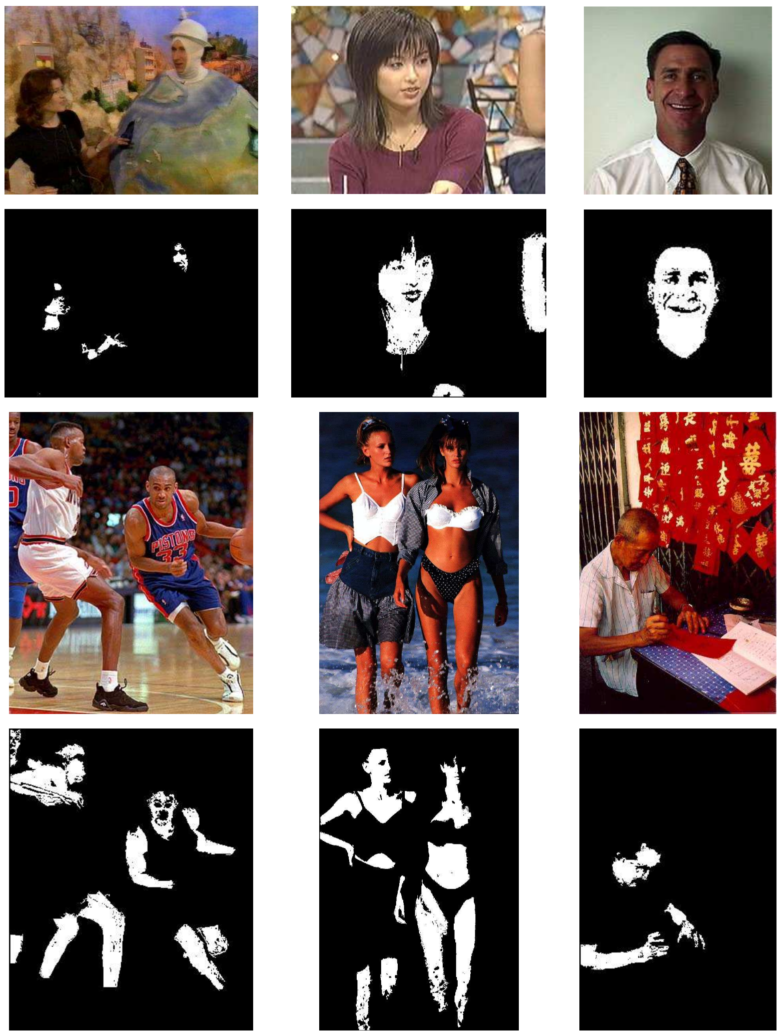


Figure A.1: Snap shot of the tool for segmenting the skin regions of an image. The left image shows the completed manual segmentation with the skin pixels highlighted in red. The right image shows the original image [JR02]



(a) Images with skin and their mask images

(b) Images without skin

Figure A.2: Samples from the Compaq image database

Appendix B

Proofs of the maximum entropy solution

Proof of the Baseline model

Here we shall derive a MaxEnt solution for the joint distribution $p(x, y)$ under the constraints $\mathcal{C}_0$. See (2.16).

Remark that the constraints in (2.16) are expectations with respect to p. Indeed,

$$p(x_s, y_s) = E_p[\delta_{x_s}(X_s)\delta_{y_s}(Y_s)] \tag{B.1}$$

with

$$\delta_a(b) = \begin{cases} 1 & \text{if } a = b \\ 0 & \text{if } a \neq b \end{cases}$$

Then, following Section 2.1, the MaxEnt solution under $\mathcal{C}_0$ can be obtained using Lagrange multipliers. One gets:

$$p(x, y) = \exp(\lambda_0 + \sum_{s \in S} \lambda(s, x_s, y_s)) \tag{B.2}$$

where the parameters λ should be set up such that the constraints are satisfied. From Section 2.1 we know that if such a solution exists, it must be the unique maximum entropy solution under these constraints. Now if

$$\forall x_s \in C, \forall y_s \in \{0, 1\}, q(x_s, y_s) > 0 \tag{B.3}$$

then one can choose

$$\lambda_0 = 0 \text{ and } \lambda(s, x_s, y_s) = \ln q(x_s, y_s) \tag{B.4}$$

which leads to the unique solution of the MaxEnt problem:

$$p(x,y) = \prod_{s \in S} q(x_s, y_s) \tag{B.5}$$

It is not hard to check that the constraints $\mathcal{C}_0$ are satisfied.

Condition in (B.3) is saying that there is no empty bin in the empirical joint histogram $q(x_s, y_s)$. This will be our case. MaxEnt solutions still exist when (B.3) is not verified.

Proof of the hidden Markov model

Here we shall obtain a MaxEnt solution for the joint distribution $p(x,y)$ under $\mathcal{C}_0 \cap \mathcal{D}$, see (2.16) and (2.24).

As for $\mathcal{C}_0$, the constraints in $\mathcal{D}$ are expectations. Indeed,

$$\forall y_s \in \{0,1\}, \forall y_t \in \{0,1\}, p(y_s, y_t) = E_p[\delta_{y_s}(Y_s)\delta_{y_t}(Y_t)] \tag{B.6}$$

Using once more Lagrange multipliers, one obtains that the MaxEnt solution, if it exists, is

$$\begin{aligned} p(x,y) &= \exp H(x,y,\lambda_0,\lambda_1,\lambda_2,\lambda_3) \text{ with} \\ H(x,y,\lambda_0,\lambda_1,\lambda_2,\lambda_3) &= \lambda_0 + \textstyle\sum_{s\in S} \lambda_1(s, x_s, y_s) + \\ &\quad \textstyle\sum_{s\sim t \in S\times S} \lambda_2(s,t)(1-y_s)(1-y_t) + \\ &\quad \textstyle\sum_{s\sim t \in S\times S} \lambda_3(s,t) y_s y_t \end{aligned} \tag{B.7}$$

where $s \sim t$ is a couple of 4-neighbors pixels and λ_0, λ_1, λ_2, λ_3 define parameters that should be set up such that the constraints are satisfied. Starting from (B.7),remark that

$$p(x_s, y_s) = \sum_{x_t; t\in S, t\neq s} \sum_{y_t; t \in S, t\neq s} p(x,y) = \exp\left(\lambda_0 + \lambda_1(s, x_s, y_s)\right) g(s, y_s) \tag{B.8}$$

with $g(s, y_s)$ a function that doesn't depend on x_s. Now,

$$p(y_s) = \sum_{x_s} p(x_s, y_s) = \exp(\lambda_0) g(s, y_s) \sum_{x_s} \exp\left(\lambda_1(s, x_s, y_s)\right) \tag{B.9}$$

hence

$$p(x_s|y_s) = \frac{p(x_s, y_s)}{p(y_s)} = \frac{\exp\left(\lambda_1(s, x_s, y_s)\right)}{\sum_{x_s} \exp\left(\lambda_1(s, x_s, y_s)\right)} \tag{B.10}$$

Since $p(x,y)$ lies in $\mathcal{C}_0$, it verifies: $p(x_s|y_s) = q(x_s|y_s)$. Assuming positivity (B.3), we can choose

$$\lambda_1(s, x_s, y_s) = \ln q(x_s|y_s) \tag{B.11}$$

Now, constraints in $\mathcal{D}$, see (2.24), do not depend on the location $s \sim t$. Hence, one can reduce to translation invariant models as in (2.25).

Proof of the first-order model

Constraints in $\mathcal{C}_1$, see (2.29) are also expectations. Indeed,

$$p(x_s, x_t, y_s, y_t) = E_p[\delta_{(x_s)}(X_s)\delta_{(x_t)}(X_t)\delta_{(y_s)}(Y_s)\delta_{(y_t)}(Y_t)] \tag{B.12}$$

Using Lagrange multipliers, one obtains (2.30).

Appendix C

Iterative scaling algorithm

Suppose a random variable X can take on n different discrete values $(x_1, x_2, \cdots, x_n)$ with the unknown distribution p. Now we observed a sequence of samples $x^{(1)}, \ldots, x^{(N)}$, $N > 1$ from p. We measure a line of $m, m < n$, features $f_k(X), k = 1, \cdots, m$, for X. The expectation of these features $E[f_k(X)]$ should be set equal to those measured from the training set $\hat{E}[f_k(X)] = \frac{1}{N}\sum_{j=1}^{N} f_k(x^{(j)})$, i.e., $E[f_k(X)] = \hat{E}[f_k(X)]$. We learned from Section 2.1 that the maximum entropy solution to p while these constraints are satisfied is the following parametric form

$$p_\lambda(x_i) = \frac{1}{Z_\lambda} \exp(\sum_{k=1}^{m} \lambda_k f_k(x_i)), \quad \forall 1 \leq i \leq n \tag{C.1}$$

where the partition function

$$Z_\lambda = \sum_i \exp(\sum_{k=1}^{m} \lambda_k f_k(x_i)) \tag{C.2}$$

Theoritically the parameters $\lambda_k, k = 1, \ldots, m$ should be solved by the following equations:

$$E[f_k(X)] = \sum_{i=1}^{n} p_\lambda(x_i) f_k(x_i) = \hat{E}[f_k(X)], \quad 1 \leq k \leq m \tag{C.3}$$

However the above equation system involves a huge number of equations even for small images. We have to get around it through other means. Stick to the parametric form p_λ defined by (C.1). We learned from [BPP96] that the MaxEnt model p^* is equivalent to a distribution p_{λ^*} where λ^* maximize the likelihood of the empirical distribution predicted by the model p_λ. That is,

$$\lambda^* = \arg\max_\lambda \mathcal{L}(\lambda)$$

in which

$$\mathcal{L}(\lambda) = \prod_{j=1}^{N} p_\lambda(x^{(j)})$$

By monotonity of the log function, we shall maximize the log likelihood function $L(\lambda)$ with respect to λ, where

$$\begin{aligned} L(\lambda) &= \frac{1}{N}\sum_{j=1}^{N}\ln p_\lambda(x^{(j)}) = \frac{1}{N}\sum_{j=1}^{N}\left(\sum_{k=1}^{m}\lambda_k f_k(x^{(j)}) - \ln Z_\lambda\right) \\ &= \sum_{k=1}^{m}\lambda_k \hat{E}[f_k] - \ln Z_\lambda \end{aligned} \tag{C.4}$$

Now the maximum of the log likelihood function could be done numerically with the iterative scaling algorithm. In this algorithm, a sequence of parameter set $\lambda^{(0)}, \ldots, \lambda^{(t)}, \ldots$ is generated. The likelihood is augemented a little after each updating of the parameters. It goes to the maximal value in the long run. In the following we shall give a brief derivation of this algorithm. Supposing

$$\sum_{k=1}^{m} f_k(x_i) = c, \quad 1 \le i \le n,$$

where $c > 0$ is a constant. [Sch03] gives the deduction for the case of $c = 1$. Here we shall extend it to the general case of any positive c. We write $\Delta\lambda_k^{(t)} = \lambda_k^{(t+1)} - \lambda_k^{(t)}$, $1 \le k \le m$. The augmentation of the log likelihood function from $\lambda^{(t)}$ to $\lambda^{(t+1)}$ is calculated as follows.

$$\begin{aligned} \Delta^{(t)}L &= L(\lambda^{(t+1)}) - L(\lambda^{(t)}) \\ &= \sum_{k=1}^{m}\Delta\lambda_k^{(t)}\hat{E}[f_k] - \ln(\frac{Z_{\lambda^{(t+1)}}}{Z_{\lambda^{(t)}}}) \end{aligned} \tag{C.5}$$

Now

$$\begin{aligned} \frac{Z_{\lambda^{(t+1)}}}{Z_{\lambda^{(t)}}} &= \sum_{i=1}^{n}\frac{\exp(\sum_{k=1}^{m}\lambda_k^{(t+1)}f_k(x_i))}{Z_{\lambda^{(t)}}} \\ &= \sum_{i=1}^{n}\frac{\exp(\sum_{k=1}^{m}\lambda_k^{(t)}f_k(x_i))\exp(\sum_{k=1}^{m}\Delta\lambda_k^{(t)}f_k(x_i))}{Z_{\lambda^{(t)}}} \\ &= \sum_{i=1}^{n}p_{\lambda^{(t)}}(x_i)\exp(\sum_{k=1}^{m}\Delta\lambda_k^{(t)}f_k(x_i)) \end{aligned} \tag{C.6}$$

Since the exponential function is convex and $\sum_{k=1}^{m}\frac{f_k(x_i)}{c} = 1$, by Jensen's inequality, we have

$$\exp(\sum_{k=1}^{m}\Delta\lambda_k^{(t)}f_k(x_i)) = \exp(\sum_{k=1}^{m}c\Delta\lambda_k^{(t)}\frac{f_k(x_i)}{c}) \le \sum_{k=1}^{m}\frac{f_k(x_i)}{c}\exp(c\Delta\lambda_k^{(t)})$$

Plug the above result into (C.6) then (C.5), we have

$$\Delta^{(t)}L \ge \sum_{k=1}^{m}\Delta\lambda_k^{(t)}\hat{E}[f_k] - \ln\left(\sum_{k=1}^{m}\exp(c\Delta\lambda_k^{(t)})E_{\lambda^{(t)}}[f_k]\right) + \ln c = B_\lambda^{(t)} \tag{C.7}$$

This inequality defines the lower bound of the increasement of the log likelihood. We shall choose the $\Delta\lambda^{(t)}$ such that $B_\lambda^{(t)}$ is maximized, so that the sequence $\lambda^{(0)}, \dots, \lambda^{(t)}, \dots$ increases the likelihood in the fastest possible direction. By taking partial derivatives of $B_\lambda^{(t)}$ with respect to $\Delta\lambda_k^{(t)}$, $1 \le k \le m$, and setting them to 0, we get:

$$\frac{\partial B_\lambda^{(t)}}{\Delta\lambda_k^{(t)}} = \hat{E}[f_k] - \frac{c\exp(c\Delta\lambda_k^{(t)})E_{\lambda^{(t)}}[f_k]}{\sum_{l=1}^m \exp(c\Delta\lambda_l^{(t)})E_{\lambda^{(t)}}[f_l]} = 0 \tag{C.8}$$

Remark that whenever we find a solution $\Delta\lambda_k^{(t)}$, $1 \le k \le m$, then $\Delta\lambda_k^{(t)} + a$ with a being an arbitrary constant will also be the solution to equations (C.8). We can choose the set of $\Delta\lambda_k^{(t)}$ such that $\sum_{l=1}^m \exp(c\Delta\lambda_l^{(t)})E_{\lambda^{(t)}}[f_l] = c$. With this choice, we have the solution:

$$\Delta\lambda_k^{(t)} = \frac{1}{c}\ln\frac{\hat{E}[f_k]}{E_{\lambda^{(t)}}[f_k]} \tag{C.9}$$

The lower bound $B_\lambda^{(t)}$ of the increasement of log likelihood is then:

$$B_\lambda^{(t)} \; = \; \sum_{k=1}^m \Delta\lambda_k^{(t)}\hat{E}[f_k] = \sum_{k=1}^m \frac{\hat{E}[f_k]}{c}\ln\frac{\hat{E}[f_k]}{E_{\lambda^{(t)}}[f_k]} = D\left(\frac{\hat{E}[f]}{c}\|\frac{E_{\lambda^{(t)}}[f]}{c}\right) \ge 0$$

where $D\left(\frac{\hat{E}[f]}{c}\|\frac{E_{\lambda^{(t)}}[f]}{c}\right)$ is the KL distance between the two distributions $\frac{\hat{E}[f]}{c}$ and $\frac{E_{\lambda^{(t)}}[f]}{c}$. $D\left(\frac{\hat{E}[f]}{c}\|\frac{E_{\lambda^{(t)}}[f]}{c}\right)$ is always positive unless $\hat{E}[f] = E_{\lambda^{(t)}}[f]$, where the constraints (C.3) are fulfilled. So the updating of the parameters according to (C.9) will increase the log likelihood function (C.4) until the equations (C.3) hold, and the likelihood function gets maximized.

The algorithm starts from an arbitrary intial set of parameters $\lambda^{(0)}$. Then these parameters are updated iteratively according to (C.9). We write the updating rule explictly as follows:

$$\lambda_k^{(t+1)} = \lambda_k^{(t)} + \frac{1}{c}\ln\frac{\hat{E}[f_k]}{E_{\lambda^{(t)}}[f_k]}, \quad 1 \le k \le m \tag{C.10}$$

for $t = 0, 1, \dots$

The sequence of distributions will then be updated following

$$p_{\lambda^{(t+1)}}(x) = \frac{1}{Z}p_{\lambda^{(t)}}(x)\prod_{k=1}^m\left(\frac{\hat{E}[f_k]}{E_{\lambda^{(t)}}[f_k]}\right)^{f_k(x)/c}$$

Let us try to understand this intuitively. Assume that $E_{\lambda^{(t)}}[f_l] < \hat{E}[f_l]$, $\forall l \in I$, where $I \subset \{1, \dots, m\}$ is the subset of indices of features. The corresponding ratios inside the product above will be greater than 1, therefore after the iteration, the distribution will concentrate more on the points with higher feature values f_l, and the expected values of the features f_l are brought closer to the empirical average [Sch03].

Appendix D

Markov chains

This introduction follows [Win03] with notations adjusted to the convention of this thesis and some efforts to make it more condensed. Let $\mathbf{X}$ be a finite set. The family of conditional distribution $(p(X|X = x), x \in \mathbf{X})$ is called a *transition probability* or a *Markov kernel*. $\mathbf{X}$ is called the *state space*. One can imagine a *Markov kernel* to be a matrix of $|\mathbf{X}| \times |\mathbf{X}|$ in which each row is the distribution $p(X|X = x)$ corresponding to a $x \in X$.

A *Markov chain* on the finite state space $\mathbf{X}$ is a stochastic process that starts from some initial distribution ν on $\mathbf{X}$ and transits according to a sequence of Markov kernels $p_1, p_2, \ldots$ on $\mathbf{X}$. If $p_i = p$, $\forall i \geq 1$, then it is a *homogeneous* Markov chain. A Markov chain fulfills the following property:

$$\pi(x_n|x_0, x_1, \ldots, x_{n-1}) = \pi(x_n|x_{n-1}), \forall n \geq 1, \forall x_0, x_1, \ldots, x_{n-1} \in \mathbf{X} \tag{D.1}$$

For a Markov chain on the finite state space $\mathbf{X}$ with kernels p_i, $i \geq 1$, given the initial state x_0, the probability of its ending up with the state x_n after n transitions is

$$p_1 p_2 \ldots p_n(x_n|x_0) = \sum_{x_1 \in \mathbf{X}} \cdots \sum_{x_{n-1} \in \mathbf{X}} p_1(x_1|x_0) \ldots p_n(x_n|x_{n-1}) \tag{D.2}$$

If the Markov chain starts with the intial distribution ν then its probability of being in the state x_n after n transitions is:

$$\nu p_1 p_2 \ldots p_n(x_n) = \sum_{x_0 \in \mathbf{X}} \sum_{x_1 \in \mathbf{X}} \cdots \sum_{x_{n-1} \in \mathbf{X}} \nu(x_0) p_1(x_1|x_0) \ldots p_n(x_n|x_{n-1}) \tag{D.3}$$

Note that $\nu p_1 p_2 \ldots p_n$ corresponds to matrix multiplication if we consider ν as a row vector. All the rules of matrix multiplication carry over to the composition of kernels. In particular, we have the associative law: $(p_1 \ldots p_i)(p_{i+1} \ldots p_n) = (p_1 \ldots p_j)(p_{j+1} \ldots p_n)$, $1 \leq i, j \leq n-1$. For a homogeneous Markov chain, we have $p_i = p$, $\forall i \geq 1$, then we can simply write p^n instead of $p_1 p_2 \ldots p_n$. If there exists a positive integer τ such that $p^\tau(x, y) > 0$, $\forall x, y \in \mathbf{X}$, then p is a *primitive* kernel. In what follows, we limit the discussion to the homogeneous Markov chain.

If a probability distribution μ satisfies $\mu p = \mu$, then we have $\mu p^n = \mu$ for each $n \geq 0$. Such a distribution is said to be *invariant* or *stationary* for p. The primitive kernel is linked to the invariant distribution through the following theorem.

Theorem 6 (Limit theorem) *Suppose p is a primitive Markov kernel on a finite space. Then it must have a unique strictly positive invariant distribution μ. Moreover, uniformly for any intial distribution ν,*

$$\nu p^n \longrightarrow \mu \quad \text{as} \quad n \longrightarrow \infty$$

The proof is referred to [Win03].

For any Markov kernel p and any distribution μ, we notate $\mu p(x_1, x_2) = \mu(x_1) p(x_2|x_1)$. It is the probability of the sequence (x_1, x_2) given the initial distribution μ and the Markov kernel p. We say p and μ satisfy the *detailed balance equation* if

$$\mu p(x, y) = \mu p(y, x), \forall x, y \in \mathbf{X} \tag{D.4}$$

Then the Markov chain with the initial distribution μ and transition kernel p is called *reversible* in time, and p is called reversible with respect to μ. An important conclusion follows that if p is reversible with respect to μ, then μ is invariant with respect to p, the proof of which is referred to [Win03].

In what follows we are going to introduce the *law of large numbers* for Markov chain. Before that, we shall explain some terminology. For two distributions μ and ν, the *norm of total variation* of the difference $\mu - \nu$ is defined as

$$\|\mu - \nu\| = \sum_{x \in \mathbf{X}} |\mu(x) - \nu(x)|$$

The *contraction coefficient* of a Markov kernel p is defined by

$$c(p) = \frac{1}{2} \max_{x, y \in \mathbf{X}} \|p(\cdot|x) - p(\cdot|y)\|$$

Note that $c(p) \leq 1$ always holds since $p(\cdot|x)$ and $p(\cdot|y)$ are probability distributions. For a strictly positive Markov kernel p we have $c(p) < 1$. The law of large numbers for a Markov chain is stated as follows:

Theorem 7 (Law of large numbers) *Let p be a Markov kernel on the finite space $\mathbf{X}$ with an invariant distribution μ, $c(p) < 1$. Then for any initial distribution ν and any function f on $\mathbf{X}$, we have*

$$E_{\nu p^n}\left[\left(\frac{1}{n+1}\sum_{i=0}^{n} f(X_i) - E_\mu(f)\right)^2\right] \longrightarrow 0, \quad \text{as} \quad n \longrightarrow \infty$$

Appendix E

Fit ellipses for gray-scale regions

Here we shall detail the formulae used in Chapter 6 to calculate the fit ellipses for gray-scale regions in images. Let S be the set of pixels of a normalized gray-scale image y. For a given pixel $s \in S$, y_s is the gray-scale value at s, which is a real number in the interval $[0, 1]$. r_s is the row coordinate and c_s is the column coordinate of pixel s. Let R be a subset of S, that is, $R \subseteq S$.

Centroid of R: The centroid $(\overline{r}, \overline{c})$ of R in the image y is defined as:

$$\begin{cases} \overline{r} &= \frac{1}{A}\sum_{s\in R} r_s y_s \\ \overline{c} &= \frac{1}{A}\sum_{s\in R} c_s y_s \end{cases} \tag{E.1}$$

where $A = \sum_{s\in R} y_s$.

Second-order Moments of R:

Second-order row moment:

$$\mu_{rr} = \frac{1}{A}\sum_{s\in R}(r_s - \overline{r})^2 y_s = \frac{1}{A}\sum_{s\in R} r_s^2 y_s - \overline{r}^2 \tag{E.2}$$

Second-order column moment:

$$\mu_{cc} = \frac{1}{A}\sum_{s\in R}(c_s - \overline{c})^2 y_s = \frac{1}{A}\sum_{s\in R} c_s^2 y_s - \overline{c}^2 \tag{E.3}$$

Second-order mixed moment:

$$\mu_{rc} = \frac{1}{A}\sum_{s\in R}(r_s - \overline{r})(c_s - \overline{c}) y_s = \frac{1}{A}\sum_{s\in R} r_s c_s y_s - \overline{rc} \tag{E.4}$$

Fit ellipse of R: Let l_{minor} be the length of the minor axis of the fit ellipse of R, and l_{major} be the length of the major axis of the fit ellipse of R. Let $\theta \in [-\frac{\pi}{2}, \frac{\pi}{2}]$ be the angle of the major axis of the fit ellipse of R counterclockwise from the column axis. There are four cases for the calculation of l_{minor}, l_{major} and θ [HS92]:

1. $\mu_{rc} = 0$ and $\mu_{rr} > \mu_{cc}$

$$\begin{cases} l_{major} &= 4\sqrt{\mu_{rr}} \\ l_{minor} &= 4\sqrt{\mu_{cc}} \\ \theta &= -90^\circ \end{cases} \tag{E.5}$$

2. $\mu_{rc} = 0$ and $\mu_{rr} \leq \mu_{cc}$

$$\begin{cases} l_{major} &= 4\sqrt{\mu_{cc}} \\ l_{minor} &= 4\sqrt{\mu_{rr}} \\ \theta &= 0^\circ \end{cases} \tag{E.6}$$

3. $\mu_{rc} \neq 0$ and $\mu_{rr} \leq \mu_{cc}$

$$\begin{cases} l_{major} &= \sqrt{8\left((\mu_{rr} + \mu_{cc}) + \sqrt{(\mu_{rr} - \mu_{cc})^2 + 4{\mu_{rc}}^2}\right)} \\ l_{minor} &= \sqrt{8\left((\mu_{rr} + \mu_{cc}) - \sqrt{(\mu_{rr} - \mu_{cc})^2 + 4{\mu_{rc}}^2}\right)} \\ \theta &= \arctan \frac{-2\mu_{rc}}{(\mu_{cc}-\mu_{rr})+\sqrt{(\mu_{cc}-\mu_{rr})^2+4{\mu_{rc}}^2}} \end{cases} \tag{E.7}$$

4. $\mu_{rc} \neq 0$ and $\mu_{rr} > \mu_{cc}$

$$\begin{cases} l_{major} &= \sqrt{8\left((\mu_{rr} + \mu_{cc}) + \sqrt{(\mu_{rr} - \mu_{cc})^2 + 4{\mu_{rc}}^2}\right)} \\ l_{minor} &= \sqrt{8\left((\mu_{rr} + \mu_{cc}) - \sqrt{(\mu_{rr} - \mu_{cc})^2 + 4{\mu_{rc}}^2}\right)} \\ \theta &= \arctan \frac{(\mu_{rr}-\mu_{cc})+\sqrt{(\mu_{rr}-\mu_{cc})^2+4{\mu_{rc}}^2}}{-2\mu_{rc}} \end{cases} \tag{E.8}$$

Let A_e be the area of the fit ellipse, then we have [HS92]:

$$A_e = 4\pi\sqrt{\mu_{rr}\mu_{cc} - {\mu_{rc}}^2},$$