

Skin Detection: A Bayesian Network Approach

Nicu Sebe¹, Ira Cohen², Thomas S. Huang³, Theo Gevers¹

¹ Faculty of Science, University of Amsterdam, The Netherlands

² HP Research Labs, USA

³ Beckman Institute, University of Illinois at Urbana-Champaign, USA

Abstract

The automated detection and tracking of humans in computer vision necessitates improved modeling of the human skin appearance. In this paper we propose a Bayesian network approach for skin detection. We test several classifiers and propose a methodology for incorporating unlabeled data. We apply the semi-supervised approach to skin detection and we show that learning the structure of Bayesian network classifiers enables learning good classifiers with a small labeled set and a large unlabeled set.

1. Introduction

Skin is arguably the most widely used primitive in human image processing research, with applications ranging from face detection [16] and person tracking [14] to pornography filtering [2, 6]. We are especially interested in skin detection as a cue for detecting people in real-world photographs. The main challenge is to make skin detection robust to the large variations in appearance that can occur. Skin appearance changes in color and shape is often affected by occlusion (clothing, hair, eye glasses, etc.). Moreover, changes in intensity, color, and location of light sources affect skin appearance. Other objects within the scene may cast shadows or reflect additional light and so forth. Finally, there are many other objects which are easily confused with skin: certain types of wood, copper, sand as well as clothes often have skin-like colors.

Research has been performed on the detection of skin pixels in color images and on the discrimination between skin and non-skin pixels by use of various statistical color models [9]. Saxe and Foulds [13] proposed an iterative skin identification method that uses histogram intersection in HSV color space. In contrast to the nonparametric methods mentioned above, Gaussian density functions [15] and a mixture of Gaussians [11] are often used to model skin color. The motivation for using a mixture of Gaussians is based on the observation that the color histogram for the skin of people with different ethnic background does not form a unimodal distribution, but rather a multimodal distribution. Recently, Jones and Rehg [10] conducted a large-scale experiment in which nearly 1 billion labeled skin tone pixels were collected (in normalized RGB color space). Comparing the performance of histogram and mixture models for skin detection, they found histogram models to be superior in accuracy and

computational cost.

In this paper we are interested in two aspects. First, the research efforts mentioned above tried to classify each individual pixel as being skin or non-skin. We want to take a different approach: can we learn the dependencies (the structure) between the pixels within a skin patch? Can we then use this structure for classification? To achieve this we use Bayesian Networks (Section 2). Bayesian Networks can represent joint distributions in an intuitive and efficient way; as such, Bayesian Networks are naturally suited for classification. Second, we are interested in using a framework that allows for the usage of labeled and unlabeled data. This is a very important aspect because one of the challenges facing researchers is the relatively small amount of available labeled data. Construction and labeling of a good ground-truth requires time and effort. However, collecting unlabeled data is not as difficult. It is therefore beneficial to use classifiers that are learnt with a combination of some labeled data and a large amount of unlabeled data. Bayesian networks are very well suited for this task: they can be learned with labeled and unlabeled data using maximum likelihood estimation. A discussion on how to incorporate unlabeled data in learning Bayesian Networks and the corresponding effect on the classification results is presented in Section 3. The experimental analysis on skin detection is presented in Section 4.

2. Bayesian Networks (BN) Classifiers

The goal is to label an incoming vector of observables \mathbf{X} . Each instantiation of \mathbf{X} is a *sample*. We assume that there is a *class variable* C ; the values of C are the *classes* (labels). Note that in our skin detection application, \mathbf{X} stands for pixels in an image patch (3x3 in our case) used as features for the classifier and we use two classes in C : skin and non-skin.

We want to build *classifiers* that receive a sample \mathbf{x} and output either one of the values of C . We assume 0-1 loss, and consequently our objective is to minimize the probability of classification error. If we knew exactly the joint distribution $p(C, \mathbf{X})$, the optimal rule would be to choose the class value with the maximum a-posteriori probability, $p(C|\mathbf{x})$. This classification rule attains the minimum possible classification error, called the *Bayes error*.

We consider probabilistic classifiers that represent the a-posteriori probability of the class given the features, $p(C, \mathbf{X})$, using Bayesian networks. A Bayesian network is composed of a directed acyclic graph in which every node is associ-

ated with a variable X_i and with a conditional distribution $p(X_i|\Pi_i)$, where Π_i denotes the parents of X_i in the graph. The directed acyclic graph is the *structure*, and the distributions $p(X_i|\Pi_i)$ represent the *parameters* of the network. We say that the assumed structure for a network, S' , is *correct* when it is possible to find a distribution, $p(C, \mathbf{X}|S')$, that matches the distribution that generates data; otherwise, the structure is *incorrect*. We use maximum likelihood estimation to learn the parameters of the network.

Given a Bayesian network classifier with parameter set Θ , the optimal classification rule under the maximum likelihood (ML) framework to classify an observed feature vector of n dimensions, $\mathbf{X} \in R^n$, to one of $|C|$ class labels, $c \in \{1, \dots, |C|\}$, is given as:

$$\hat{c} = \arg \max_c P(\mathbf{X}|c; \Theta). \quad (1)$$

There are two design decisions when building Bayesian network classifiers. The first is to choose the structure of the network, which will determine the dependencies among the variables in the graph. The second is to determine the distribution of the features. The features can be discrete, in which case the distributions are probability mass functions. The features can also be continuous, in which case one typically has to choose a distribution, with the most common being the Gaussian distribution. Both these design decisions determine the parameter set Θ which defines the distribution needed to compute the decision function in Eq. (1).

Two examples of popular Bayesian network classifiers are the Naive Bayes (NB) classifier, in which the features are assumed independent given the class, and the Tree-Augmented Naive Bayes classifier (TAN).

The NB classifier makes the assumption that all features are conditionally independent given the class label. Although this assumption is typically violated in practice, NB have been used successfully in many classification applications. One of the reasons for the NB success is attributed to the small number of parameters needed to be learnt.

If the features in \mathbf{X} are assumed to be independent of each other conditioned upon the class label c (the Naive Bayes framework), Eq. (1) reduces to:

$$\hat{c} = \arg \max_c \prod_{i=1}^n P(x_i|c; \Theta). \quad (2)$$

Now the problem is how to model $P(x_i|c; \Theta)$, which is the probability of feature x_i given the class label. In practice, the common assumption is that we have a Gaussian distribution and the ML can be used to obtain the estimate of the parameters (mean and variance).

Friedman, et al. [7] proposed the use of the Tree-Augmented Naive Bayes model as a classifier, to enhance the performance over the simple Naive-Bayes classifier. In the structure of the TAN classifier, the class variable is the parent of all the features and each feature has at most one other feature as a parent, such that the resultant graph of the features forms a tree. For learning the TAN classifier we do

not fix the structure of the Bayesian network, but we try to find the TAN structure that maximizes the likelihood function given the training data out of all possible TAN structures. In general, searching for the best structure has no efficient solution, however, searching for the best TAN structure does have one. The method is using the modified Chow-Liu algorithm for constructing tree augmented Bayesian networks [7]. The algorithm finds the tree structure among the features that maximizes the likelihood of the data by computation of the pairwise class conditional mutual information among the features and building a maximum weighted spanning tree using the pairwise mutual information as the weights of the arcs in the tree. The problem of finding a maximum weighted spanning is defined as finding the set of arcs connecting the features such that the resultant graph is a tree and the sum of the weights of the arcs is maximized. There have been several algorithms proposed for building a maximum weighted spanning tree [5] and in our implementation we use the Kruskal algorithm.

3. Semi-supervised Learning of BN

Is there value to unlabeled data in supervised learning of classifiers? This fundamental question has been increasingly discussed in recent years, with a general optimistic view that unlabeled data hold great value. Due to an increasing number of applications and algorithms that successfully use unlabeled data [1, 8] and magnified by theoretical issues over the value of unlabeled data in certain cases [3], semi-supervised learning is seen optimistically as a learning paradigm that can relieve the practitioner from the need to collect many expensive labeled training data.

Consider the following scenario. A sample (c, \mathbf{x}) is generated from $p(C, \mathbf{X})$. The value c is then either revealed, and the sample is a *labeled* one; or the value c is hidden, and the sample is an *unlabeled* one. The probability that any sample is labeled, denoted by λ , is fixed, known, and independent of the samples. Thus the same underlying distribution $p(C, \mathbf{X})$ models both labeled and unlabeled data. Given a set of N_l labeled samples and N_u unlabeled samples, we use maximum likelihood for estimating $\hat{\theta}$. The assumed distribution $p(C, \mathbf{X}|\theta)$ can be decomposed either as $p(C|\mathbf{X}, \theta) p(\mathbf{X}|\theta)$ or as $p(\mathbf{X}|C, \theta) p(C|\theta)$. A parametric model where both $p(\mathbf{X}|C, \theta)$ and $p(C|\theta)$ depend explicitly on θ is referred to as a *generative model*. The log-likelihood function of this model for a dataset with labeled and unlabeled data is:

$$L(\theta) = L_l(\theta) + L_u(\theta) + \log \left(\lambda^{N_l} (1 - \lambda)^{N_u} \right) \quad (3)$$

$$L_l(\theta) = \sum_{i=1}^{N_l} \log \left[\prod_C (p(C = c'|\theta) p(\mathbf{x}_i|c', \theta))^{I_{\{C=c'\}}(c_i)} \right]$$

$$L_u(\theta) = \sum_{j=(N_l+1)}^{N_l+N_u} \log \left[\sum_C p(C = c'|\theta) p(\mathbf{x}_j|c', \theta) \right] = \sum_{j=(N_l+1)}^{N_l+N_u} \log [p(\mathbf{x}_j|\theta)]$$

where $I_A(Z)$ is the indicator function (1 if $Z \in A$; 0 otherwise) and $p(C = c')$ are the mixing coefficients. $L_l(\theta)$ and

$L_u(\theta)$ are the likelihoods of the labeled and unlabeled data, respectively.

When unlabeled data are available, estimating the parameters of the Naive Bayes classifier can be done using the EM algorithm. As for learning the TAN classifier, we learn the structure and parameters using the EM-TAN algorithm, derived from [12].

Despite the optimistic view mentioned above, several disparate empirical evidences in the literature suggest that there are situations in which the addition of unlabeled data to a pool of labeled data causes degradation of the classifier's performance [1], in contrast to improvement of performance when adding more labeled data. In [4] we present an extensive analysis demonstrating that, counter to statistical intuition, when the assumed model of the classifier does not match the true data generating distribution, classification performance could degrade as more and more unlabeled data are added to the training set. Motivated by this, we consider a classification driven stochastic structure search (SSS) algorithm for learning the structure of Bayesian network classifiers that minimizes the probability of classification error.

First we define a measure over the space of structures which we want to maximize:

Definition 1 *The inverse error measure for structure S' is*

$$inv_e(S') = \frac{1}{\sum_S \frac{1}{p_S(\hat{c}(\mathbf{X}) \neq C)}}, \quad (4)$$

where the summation is over the space of possible structures and $p_S(\hat{c}(\mathbf{X}) \neq C)$ is the probability of error of the best classifier learned with structure S .

We use Metropolis-Hastings sampling to generate samples from the inverse error measure, without having to ever compute it for all possible structures. For constructing the Metropolis-Hastings sampling, we define a neighborhood of a structure as the set of directed acyclic graphs to which we can transit in the next step. Transition is done using a predefined set of possible changes to the structure; at each transition a change consists of a single edge addition, removal, or reversal. We define the acceptance probability of a candidate structure, S_{new} , to replace a previous structure, S_t as follows:

$$\min\left(1, \left(\frac{inv_e(S_{new})}{inv_e(S_t)}\right)^{\frac{1}{T}} \frac{q(S_t|S_{new})}{q(S_{new}|S_t)}\right) = \min\left(1, \left(\frac{p_{error}^t}{p_{error}^{new}}\right)^{\frac{1}{T}} \frac{N_t}{N_{new}}\right) \quad (5)$$

where $q(S'|S)$ is the transition probability from S to S' , T is a temperature factor, and N_t and N_{new} are the sizes of the neighborhoods of S_t and S_{new} respectively; this choice corresponds to equal probability of transition to each member in the neighborhood of a structure. This further creates a Markov chain which is aperiodic and irreducible, thus satisfying the Markov chain Monte Carlo (MCMC) conditions. Roughly speaking, T close to 1 would allow acceptance of more structures with higher probability of error than previous structures. T close to 0 mostly allows acceptance of structures that improve probability of error. A fixed T amounts to

changing the distribution being sampled by the MCMC, while a decreasing T is a simulated annealing run, aimed at finding the maximum of the inverse error distribution. The rate of decrease of the temperature determines the rate of convergence. Asymptotically in the number of data, a logarithmic decrease of T will guarantee convergence to a global maximum with probability that tends to one.

The advantages of the SSS algorithm are that it usually converges to better classifiers compared to other methods, and asymptotically can be shown to converge to the classifier with minimum error. Its biggest disadvantage is in the added complexity: for every structure being tested the parameters are estimated, followed by error estimation.

4. Skin Detection Experiments

In our experiments we use image patches of 9 pixels (a 3x3 patch) as the features in the Bayesian Network. We consider the rg chromaticity space, which is the most popular color space for skin color modeling [16].



Figure 1. An example of skin detection.

We use the database of Jones and Rehg [10] consists of 3,475 images containing skin and 8,796 non-skin images. Each image was manually segmented such that the skin pixels are labeled. Examples of detected skin patches are presented in Figure 1. In the experiments we randomly selected 3x3 skin and non-skin patches (100,000 in total). We leave out 40,000 patches for testing and train the Bayesian Network classifiers on the remaining 60,000. To compare the results of the classifiers, we use the receiving operating characteristic (ROC) curves. The ROC curves show, under different classification thresholds, ranging from 0 to 1, the probability of detecting a skin patch in a skin image, $P_D = P(\hat{C} = skin|C = skin)$, against the probability of falsely detecting a skin patch in a non-skin image, $P_{FD} = P(\hat{C} = non - skin|C \neq non - skin)$.

We first learn using all the training data being labeled (that is 60,000 labeled patches). Figure 2 (left) shows the resultant ROC curve for this case. The classifier learned with the SSS algorithm outperforms both TAN and NB classifiers, and all perform quite well, achieving high detection rates with a low rate of false alarm. Next we remove the labels of some of the training data and train the classifiers. Figure 2 (right) shows the case where the labels of 90% of the training data

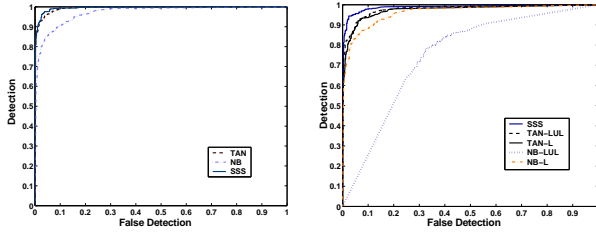


Figure 2. ROC curves showing detection rates of skin compared to false detection with all data labeled (left) and 90% unlabeled data (right): SSS, NB learned with labeled data only (NB-L) and with labeled and unlabeled data (NB-LUL), and TAN learned with labeled data only (TAN-L) and with labeled and unlabeled data (TAN-LUL).

(leaving only 600 labeled patches) were removed. We see that the NB classifier using both labeled and unlabeled data (NB-LUL) performs very poorly. The TAN based only on the 600 labeled images (TAN-L) and the TAN based on the labeled and unlabeled images (TAN-LUL) are close in performance, thus there was no significant degradation of performance when adding the unlabeled data.

In Table 1 we summarize the results obtained for different algorithms and in the presence of increasing number of unlabeled data. We fixed the false alarm to 1%, 5%, and 10% and we computed the detection rates. Note that the detection rates for NB are lower than the ones obtained for the other detectors. Overall, the results obtained with SSS are the best. We see that even in the most difficult cases, there was sufficient amount of unlabeled data to achieve almost the same performance as with a large sized labeled dataset.

Table 1. Detection rates (%) for different false positives

Detector		False detections		
		1%	5%	10%
NB	60,000 labeled	64.74	85.26	90.64
	600 labeled	61.34	81.27	84.85
	600 labeled + 54,000 unlabeled	60.05	80.77	83.98
TAN	60,000 labeled	86.85	96.22	99.00
	600 labeled	84.50	88.84	93.63
	600 labeled + 54,000 unlabeled	84.66	88.82	93.01
SSS	60,000 labeled	88.25	97.61	99.40
	600 labeled	85.23	92.51	96.15
	600 labeled + 54,000 unlabeled	87.66	95.82	98.32

5 Summary and Discussion

In this work we presented a Bayesian Network approach for skin detection. We considered several instances of Bayesian Networks and we suggested a methodology to perform skin detection using both labeled and unlabeled data.

In a nutshell, when faced with the option of learning with labeled and unlabeled data for skin detection using Bayesian networks, our discussion suggests using the following path. Start with Naive Bayes and TAN classifiers, learn only with the available labeled data, and test whether the model is correct by learning with the unlabeled data. If the result is not

satisfactory, then SSS can be used to attempt to further improve performance. If none of the methods using the unlabeled data improve performance over the supervised TAN (or Naive Bayes) the practitioner is faced with two options: discard the unlabeled data, or label some of the unlabeled data using the active learning methodology. Of course, active learning can be used as long as there are resources to label some samples.

Structure learning of Bayesian networks is not a topic motivated solely by the use of unlabeled data. Skin detection could be solved using classifiers other than Bayesian networks. However, this work should be viewed as a combination of 3 components; (1) the theory showing the limitations of unlabeled data is used to motivate (2) the design of algorithms to search for better performing structures of Bayesian networks and finally, (3) the successful application to skin detection by learning with labeled and unlabeled data.

References

- [1] S. Baluja. Probabilistic modeling for face orientation discrimination: Learning from labeled and unlabeled data. In *NIPS*, pages 854–860, 1998.
- [2] A. Bosson, G. Cawley, Y. Chan, and R. Harvey. Non-retrieval: blocking pornographic images. In *CIVR*, pages 50–60, 2002.
- [3] V. Castelli. *The relative value of labeled and unlabeled samples in pattern recognition*. PhD thesis, Stanford, 1994.
- [4] I. Cohen, F. Cozman, N. Sebe, M. Cirello, and T. Huang. Semi-supervised learning of classifiers: Theory, algorithms, and applications to human-computer interaction. *IEEE Trans. on PAMI*, to appear, 2004.
- [5] T. Cormen, C. Leiserson, and R. Rivest. *Introduction to algorithms*. MIT Press, Cambridge, MA, 1990.
- [6] M. Fleck, D. Forsyth, and C. Bregler. Finding naked people. In *ECCV*, volume 2, pages 593–602, 1996.
- [7] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29(2):131–163, 1997.
- [8] R. Ghani. Combining labeled and unlabeled data for multi-class text categorization. In *ICML*, pages 187–194, 2002.
- [9] B. Jedynek, H. Zheng, and M. Daoudi. Statistical models for skin detection. In *CVPR Workshop on Statistical Analysis in Computer Vision*, 2003.
- [10] M. Jones and J. Rehg. Statistical color models with application to skin detection. *IJCV*, 46(1):81–96, 2002.
- [11] S. McKenna, S. Gong, and Y. Raja. Modelling facial colour and identity with gaussian mixtures. *Pattern Recognition*, 31:1883–1892, 1998.
- [12] M. Meila. *Learning with mixture of trees*. PhD thesis, MIT, 1999.
- [13] D. Saxe and R. Foulds. Toward robust skin identification in video images. In *Automatic Face and Gesture Recognition*, pages 379–384, 1996.
- [14] K. Schwerdt and J. Crowley. Robust face tracking using color. In *Automatic Face and Gesture Recognition*, pages 90–95, 2000.
- [15] M.-H. Yang and N. Ahuja. Detecting human faces in color images. In *ICIP*, pages 127–130, 1998.
- [16] M.-H. Yang, D. Kriegman, and N. Ahuja. Detecting faces in images: A survey. *PAMI*, 24(1):34–58, 2002.