DISCRIMINATIVE GAUSSIAN MIXTURES FOR INTERACTIVE IMAGE SEGMENTATION

Jue Wang

Department of Electrical Engineering, University of Washington
Seattle, WA 98195, USA

ABSTRACT

Recently graph-cut optimization has been extensively explored for interactive image segmentation. In this paper we propose *Discriminative Gaussian Mixtures* (DGMs) to boost the performance of graph-cut-based segmentation. Given the user specified pixels, our algorithm analyzes their distributions in color, texture and spatial spaces and produces optimized Gaussian mixtures to set the data cost in the image graph, under the criteria of maximizing the discriminant power. We also show how to assemble novel training data to train DGMs for the link cost in the graph. Experimental results demonstrate that DGMs can noticeably improve the performance of graph-cut segmentation on texture-rich images.

Index Terms— Image segmentation, Interactive systems

1. INTRODUCTION

As shown in Figure 1, in the interactive segmentation scenario, the user first roughly specifies a few foreground and background pixels using scribbles, then invoke automatic procedures to generate the final fine mask for the foreground.

Using graph-cut-based optimization for interactive image segmentation was first introduced in [1], where segmentation was achieved by minimizing a "Gibb" energy

$$E(\underline{L}, \underline{\theta}, \mathbf{z}) = U(\underline{L}, \underline{\theta}, \mathbf{z}) + V(\underline{L}, \mathbf{z}), \qquad (1)$$

where $\mathbf{z} = (z_1, ..., z_n)$ are pixels in the image, $\underline{L}$ are binary labels(foreground or background) for pixels and $\underline{\theta}$ are parameters of models describing the distributions of the foreground and background in the chosen feature space. $U(\underline{L}, \underline{\theta}, \mathbf{z})$ is the *data cost* which evaluates the fit of labels $\underline{L}$ to pixels $\mathbf{z}$, given the statistical models $\underline{\theta}$. $V(\underline{L}, \mathbf{z})$ is the *link cost* which penalizes discontinuities of labels in locally smooth regions. The total energy can be minimized by solving a graph-cut problem with very efficient algorithms.

Recently many approaches have been proposed to improve the original graph-cut-based approach for better performance, such as the GMMRF method [2], the GrabCut system [3], the LazySnapping system [4] and the multilevel banded approach [5]. Surprisingly, these approaches mainly focus on parameter estimation and energy minimization in Equation 1, and leave the problem of feature and model selection to their simplest solutions. For instance, [3] used Gaussian Mixture Models(GMMs) in RGB space with 5 components for both foreground and background; [4] used K-means with 64 components in RGB space; and [1] and [5] used single 1D Gaussian in the histogram of grayscale pixel values. Also, these approaches trained their models separately on foreground and background samples and paid no attention to the discriminant power of these models.

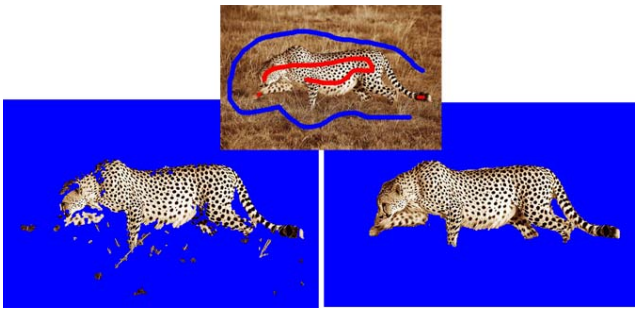

Fig. 1. Top: original image with user input. Left: Segmentation by graph-cut using traditional GMMs. Right: Segmentation by graph-cut using DGMs proposed in this paper.

On the other hand, applying clustering algorithms, especially GMMs in multiple feature spaces has been extensively studied for traditional non-interactive image segmentation. Permuter et al. [6] studied GMMs in color and texture spaces for image retrieval. Chen et al. [7] used K-means in color and texture spaces for unsupervised image segmentation. In [8] GMMs were trained with spatial constraints under a modified expectation-maximization(EM) framework for segmentation. Although these approaches were successful, optimizing GMMs for interactive segmentation has not been studied.

This paper aims at optimizing GMMs for graph-cut based interactive image segmentation. Specifically, we maximize the discriminative power of GMMs so that the optimization result of graph-cut can lead to more satisfying segmentation. The Discriminant Guassian Mixtures we propose here employ a modified EM framework to achieve automatic feature selection, mixture number selection and parameter estimation, under the criteria of maximizing the discriminant power of the models.

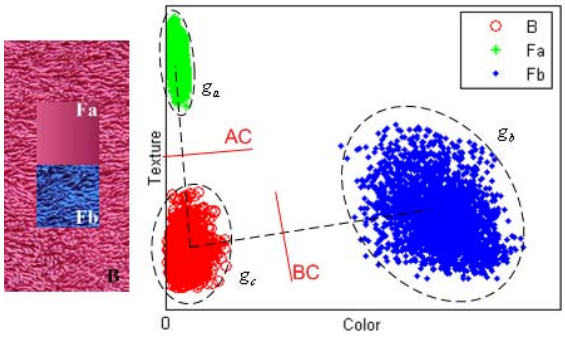

Fig. 2. Left: A synthetic example. Right: projecting feature vectors of pixels into a 2D space.

2. DISCRIMINANT GAUSSIAN MIXTURES

2.1. Formulations

Given the fact that any single feature may not be sufficient to segment the foreground in difficult images such as the one shown in Figure 1, for a pixel z we extract features from color, texture and spatial spaces, denoted as x_z^c, x_z^t and x_z^s, respectively. x_z^c is the color of z in Luv color space, and x_z^s is the (x, y) coordinate of z. The texture descriptor x_z^t is an 8 dimensional vector computed from 38 filter banks by using the the rotationally invariant, multi-scale, Maximum Response MR8 filter bank [9], which has been proven to be efficient for texture classification tests [10]. The final feature vector $x_z = [x_z^c, x_z^s, x_z^t]$ has a dimension of 13.

User specified foreground and background pixels are used to train a positive and a negative GMM. Our GMM is defined as

$$f(x;\theta) = \sum_{k=1}^{K} p_k g(x; m_k, \sigma_k, w_k) \tag{2}$$

where $g()$ is a modified multi-dimensional Gaussian function by introducing a new parameter vector w_k. Specifically, $w_k = [w_k^c, w_k^s, w_z^t]$ is a three dimensional vector consisting of three weights for each of the feature subspace. By assuming the three feature subspaces are orthogonal so that the covariance matrices $\sigma_k = diag\{\sigma_k^c, \sigma_k^s, \sigma_k^t\}$ and the mean vectors $m_k = [m_k^c, m_k^s, m_k^t]$, the modified Gaussian can be written as

$$g(x; m_k, \sigma_k, w_k) = C \cdot \exp\left\{-\frac{1}{2}\left[w_k^c \cdot M_d(x^c, m_k^c, \sigma_k^c) + w_k^s \cdot M_d(x^s, m_k^s, \sigma_k^s) + w_k^t \cdot M_d(x^t, m_k^t, \sigma_k^t)\right]\right\} \tag{3}$$

where $M_d(x, m, \sigma)$ is the Manhattan distance from a vector x to a Gaussian $G(m, \sigma)$, and C is a constant for normalization.

The underlying idea for introducing w_k into the gaussian mixture is to allow the model to automatically choose the best combination of features which can maximize its discriminant power. A synthetic example is shown in Figure 2 for better demonstration of the idea. In this example the foreground object contains two distinct regions F_a and F_b while the background B is a solid texture region. Clearly F_a and F_b will form two modes (g_a and g_b) for the foreground GMM, and the background will have a single mode g_c in the feature space, as shown in Figure 2b. Since F_b and B have almost the same texture, considering texture feature in g_b will do no good to but only harm the classification performance. Thus we should set $w_b^t \approx 0$ and $w_b^c \approx 1$ (spatial features are ignored in this example). Similarly, we should set $w_a^t \approx 1$ and $w_a^c \approx 0$ so that pixels in region F_a can be better separated from the background by using only texture information. This dynamic weight setting enables the whole mixture model to give better classification performance.

Mathematically, the weights for different feature spaces can be calculated by applying a Linear Discriminant Analysis (LDA) between a pair of foreground and background modes. As shown in Figure 2b, applying LDA between g_a and g_c will result in a linear classifier AC which guarantees maximal separability between g_a and g_c. The weights, w_a^c and w_a^t, are the parameters of this linear classifier.

2.2. Parameter Estimation

Introducing weights w_k into Eqn. 3 brings discriminant power into Gaussian mixtures, however it complicates the parameter estimation procedure. In the traditional EM framework, foreground and background GMMs are trained separately to only minimize the fitting error of the models to the corresponding samples. In our formulation, we estimate the parameters for positive and negative Gaussian mixtures jointly to minimize the fitting errors as well as maximize the separability. To achieve this we propose a modified EM algorithm as described as follows.

- Initialization: We apply K-Means to foreground and background samples separately to form initial modes for two Gaussian mixtures and calculate the means and covariance matrices of Gaussians. For each Gaussian the weights w^c, w^s and w^t are set equally to be 1/3.
- Repeat the following EM steps until convergence:
  - E-step: Calculate the membership probabilities of each sample based on the modified Gaussian in Equation (3).
  - M-step 1: Calculate the mean and covariance matrix of each Gaussian based on membership probabilities of samples.
  - M-step 2: For each Gaussian g_k^f in the foreground mixture, find the Gaussian g_l^b in the background mixture which has the closest distance to it. Apply LDA between g_k^f and g_l^b to calculate the weights w_k^c, w_k^s and w_k^t for g_k^f.

– M-step 3: Update the weights for each background Gaussian using the same method.

- End.

2.3. Selecting Number of Modes

Setting the number of modes properly is another key issue to the success of our segmentation algorithm. In general, too few modes will result in large fitting errors, while too many modes will lead to over-fitting. To automatically adjust the number of modes to achieve the best possible segmentation results, we define a foreground model performance descriptor as

$$I_f = \frac{\sum_{n=1}^{N_b} f^F(x_n^b;\theta^f)}{\sum_{n=1}^{N_f} f^F(x_n^f;\theta^f)} \quad (4)$$

where x_n^f,$n = 1,...,N_f$ are foreground pixel samples, and x_n^b,$n = 1,...,N_b$ are background ones. f^F is the foreground DGM we obtained from training. Essentially, I_f is defined as the ratio of the fitting error of background samples to the fitting error of foreground samples using the same foreground model. Apparently, a small I_f value indicates a better discriminant power of the foreground model.

We initially set K_f, the number of foreground modes to be 1, and gradually increase it. In the meanwhile, we calculate the corresponding I_f values. We stop at iteration t when $abs\|I_f(t+1) - I_f(t)\| < \epsilon$, where ϵ is a predefined threshold, and choose current K_f value as the final number of foreground modes. Similarly, we define a background model performance descriptor and use it to choose the proper number for the background modes.

Figure 3 shows the K_f vs. I_f curve and K_b vs. I_b curve for the image shown in Figure 1. Both curves have a waterfall shape. For this example the proper numbers of foreground and background modes are both chosen to be 6.

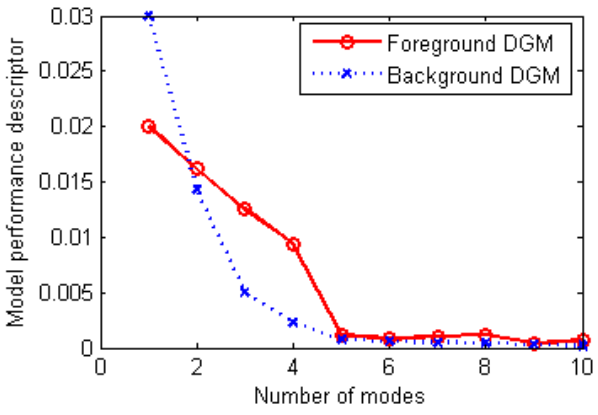

Fig. 3. K_f vs. I_f and K_b vs. I_b curve.

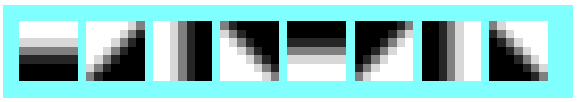

Fig. 4. Edge patch templates.

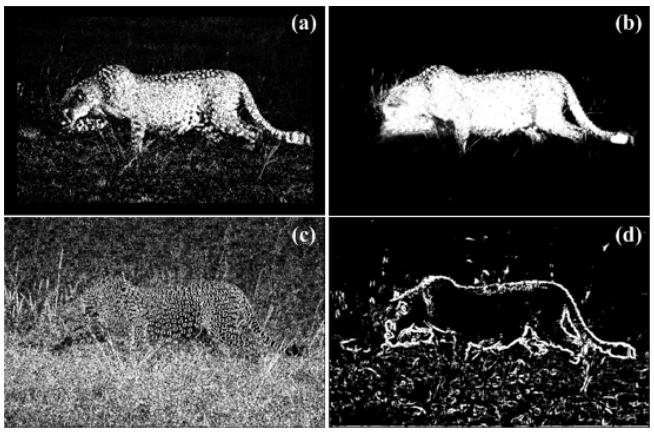

Fig. 5. (a-b): foreground probabilities computed from traditional GMMs and DGMs. (c-d): foreground boundary probabilities computed from a static function and DGM.

3. DGM FOR GRAPH-CUT

We use the Discriminant Gaussian Mixture we described above to set both the data cost and the link cost in the image graph, and use graph-cut to compute a final segmentation.

3.1. Setting Data Cost

Using DGMs for setting data cost is straightforward. Given the user specified foreground and background pixels, we train a foreground and background mixture f^F and f^B, and use them to set a data cost for a new pixel z as

$$U(L_z,z) = \begin{cases} \frac{f^F(x_z,\theta^f)}{f^F(x_z,\theta^f)+f^B(x_z,\theta^b)} & : \quad L_z = 0 \\ \frac{f^B(x_z,\theta^b)}{f^F(x_z,\theta^f)+f^B(x_z,\theta^b)} & : \quad L_z = 1 \end{cases} \quad (5)$$

For user marked pixels the data cost is set to be infinite as in previous approaches.

3.2. Setting Link Cost

The link cost encourages graph-cut to cut through the foreground edges. In previous approaches [1, 3, 4] this term is set to be a static exponential function under the assumption that large gradients in the image correspond to the foreground edges. This is true when the image is smooth, however for highly textured images strong gradients are all over the image thus cannot be used to identify the foreground boundary.

We use the proposed DGMs to train a foreground edge model to better identify the foreground boundary. Suppose we

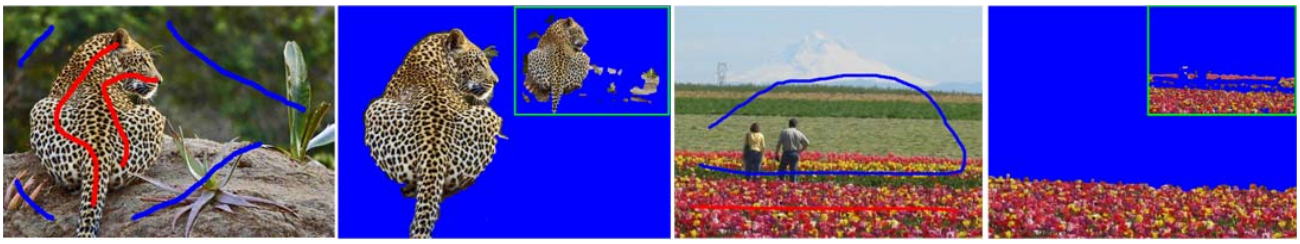

Fig. 6. Two more segmentation results. Images with green borders are segmented by traditional graph-cut based segmentation approach.

obtain some foreground edge pixels as positive samples, and some pixels inside the foreground and background as negative samples, we can train an On-the-Edge DGM f^e and an Off-the-Edge DGM f^{ne}, using the same method as we train f^F and f^B. Thus for a pair of neighboring pixels z and v, the link cost between them is set to be

$$V(z,v) = \begin{cases} \frac{\overline{f^e}(x_z,x_v)}{\overline{f^e}(x_z,x_v)+\overline{f^{ne}}(x_z,x_v)} & : \quad L_z = L_v \\ \frac{\overline{f^{ne}}(x_z,x_v)}{\overline{f^e}(x_z,x_v)+\overline{f^{ne}}(x_z,x_v)} & : \quad L_z \neq L_v \end{cases} \tag{6}$$

where $\overline{f^{e/ne}}(x_z,x_v) = \frac{f^{e/ne}(x_z)+f^{e/ne}(x_v)}{2}$.

The only problem left is that in the interactive segmentation scenario the user only marks some pixels inside the foreground and background, thus we only have negative samples to train the model f^{ne}. For the model f^e, we instead generate some synthetic foreground edge patches as training samples. As shown in Figure 4, we create eight 7×7 edge patch templates to model the smooth transitions from foreground (white) to background (black) in eight directions. The values in each patched are normalized between 0 and 1. given a pair of foreground and background patches, we can generate a synthetic edge patch by linearly combining these two patches using one of the edge templates. These synthetic edge patch are then used to fit the On-the-Edge model f^e.

4. RESULTS

In figure 1 we show DGMs significantly outperforms traditional GMMs for segmenting a very difficult image. Figure 5 visualizes the corresponding data and link costs computed from previous approaches and the proposed DGMs. It shows that DGMs can identify foreground pixels and edges much more accurately.

Two more segmentation results are shown in Figure 6. These results demonstrate that even the foreground and background contain similar textures and colors, our algorithm can still generate good results given the discriminant power embedded in the underlying statistical models.

Acknowledgement

The author is supported by a Microsoft Research Fellowship.

5. REFERENCES

[1] Y.Y. Boykov and M.P. Jolly, "Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images," in *Proc. ICCV*. IEEE, 2001, vol. I, pp. 105–112.

[2] A. Blake, C. Rother, M. Brown, P. Perez, and P. Torr, "Interactive image segmentation using an adaptive gmmrf model," in *Proc. ECCV*, 2004, pp. 428–441.

[3] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut - interactive foreground extraction using iterated graph cut," in *Proc. of ACM SIGGRAPH*, 2004, pp. 309–314.

[4] Y. Li, J. Sun, C.K. Tang, and H.Y. Shum, "Lazysnapping," in *Proc. of ACM SIGGRAPH*, 2004, pp. 303–308.

[5] H. Lombaert, Y. Sun, L. Grady, and C. Xu, "A multilevel banded graph cuts method for fast image segmentation," in *Proc. ICCV*. IEEE, 2005, vol. I, pp. 259–265.

[6] H. Permuter, J.M. Francos, and I.H. Jermyn, "Gaussian mixture models of texture and colour for image database retrieval," in *Proc. ICASSP*. IEEE, 2003, pp. 569–572.

[7] J. Chen, T.N. Pappas, A. Mojsilovic, and B.E. Rogowitz, "Image segmentation by spatially adaptive color and texture features," in *Proc. ICIP*. IEEE, 2003, pp. 1005–1008.

[8] K. Blekas, A. Likas, N.P. Galatsanos, and I.E. Lagaris, "A spatially constrained mixture model for image segmentation," *IEEE Trans Neural Network*, vol. 16, pp. 494–498, 2006.

[9] M. Varma and A. Zisserman, "Classifying images of materials: Achieving viewpoint and illumination independence," in *Proc. ECCV*, 2002, pp. 255–271.

[10] M. Varma and A. Zisserman, "Texture classification: Are filter banks necessary?," in *Proc. CVPR*, 2003, pp. 691–698.