AN UNSUPERVISED LEARNING ALGORITHM FOR IMAGE SEGMENTATION BASED ON FINITE MIXTURE MODELS

Yu lin-Sen Zhang Tian-wen

School of Computer Science and Technology,
Harbin Institute of Technology, China

ABSTRACT

There are two open problems for unsupervised learning of finite mixture models: model selection and initialization. To circumvent these problems in application of image segmentation, we integrate the filter technique into the EM algorithm. The proposed algorithm starts with the largest possible number of image regions. With the convergence of the algorithm, irrelevant components can be eliminated. It does not require careful initialization and also has the advantage of preserving the good features of EM while making use of the spatial information in a reasonable amount of time.

1. INTRODUCTION

Image segmentation plays a central role in low-level computer vision. It is a pre-requisite for solving many other computer vision problems, such as image classification, content-based image retrieval, and object recognition. Unsupervised image segmentation may be defined as the task of dividing an image into several homogeneous regions automatically based on some similarity measures. An efficient approach to measuring the similarity of regions is using probabilistic models. Perhaps the cleanest approach to segmenting points in feature space is based on mixture models. But there are two open problems for mixture models: model selection and initialization. The usual choice for obtaining ML or MAP estimates of the mixture parameters is the EM algorithm[3]. But the frequently used EM algorithm often converges to a local maximum that depends on the initial conditions. Additional computation has to be introduced to avoid the local maximum[1]. Estimating the number of mixture components, which is called model selection, is important and could have a significant effect on the quality of segmentation. The methods of model selection can be divided into two families[2]. The stochastic methods are still far too computationally demanding. The deterministic methods also need using EM algorithm to compute a set of candidate models. And for real data like images, sometimes the selected region number is very different from the natural number of groups present in the image. To circumvent these drawbacks, we integrate the filter technique into the EM algorithm. The low pass filtering operation can enlarge the volume of the mixture components, which promotes the competition among the components and make the EM algorithm has an ability to escape from a local maximum. The proposed algorithm starts with the largest possible number of image regions. By merging the components with similar parameters, our algorithm can select the proper region number automatically. Because it does not need model selection criterion, estimation and model selection can be integrated in a single algorithm. Moreover, the neighborhood operation of the filter implicitly imposes spatial constraint on the feature vectors. Thus it can reduce the situation where the result regions are spatially very mixed.

The paper is organized as follows. Section 2 reviews finite mixture models briefly. The proposed algorithm is presented in Section 3. Section 4 presents the experiment results. The paper's conclusions are summarized in section 5.

2. FINITE MIXTURE MODELS

It is said a d -dimensional random variable $x = [x_1, x_2, ...x_d]^T$ follows a K -component finite mixture distribution, if its probability density function can be written as

$$p(x \mid \Theta) = \sum_{m=1}^{K} \pi_m p(x \mid \theta_m) \qquad (1)$$

Where $\pi_m \in (0,1)$ ($\forall m = 1,2,\ldots,K$) are the mixing proportions subject to $\sum_{m=1}^{K} \pi_m = 1$, θ_m is the parameter of the m th density model and $\Theta = (\pi_1, \pi_2, \ldots, \pi_K, \theta_1, \theta_2, \ldots, \theta_K)$ is the parameter set of mixture models. Different descriptions of $p(x \mid \theta_m)$ can be assigned to different kinds of mixture models. We

focus on Gaussian finite mixture models and demonstrate the mechanism of our algorithm by means of Gaussian mixture models.

3. THE PROPSED ALGORITHM

In order to avoid using EM to compute a set of candidates and also to avoid the error that model selection criterion brings in, the proposed algorithm starts with the largest possible number of the components. Model selection can be implemented by eliminating irrelevant components in the iteration process gradually. The key step is how to eliminate irrelevant components in a reasonable way. In practice, the E-step corresponds to calculating the posterior probabilities of every pixel belonging to each of the component distributions by using the current estimate of the parameters. Then using these posterior probabilities as weights, the M-step corresponds to calculating a new maximum estimate for the parameters. For image data, every feature vector corresponds to one pixel. So rather than directly going to the M step after performing the E step, we perform lowpass filtering operation to these posterior probabilities. Low-pass filtering can enhance the robustness of estimation. And it can also enlarge the volume of the corresponding components. As a result, its competitive power is boosted. For standard EM, Local maxima of the likelihood arise when there are too many components in one region of the space, and too few in another because EM is unable to move components across low-likelihood regions. However the expansion of components by means of low-pass filtering endows the proposed algorithm with the ability to escape from local maxima.

3.1 Filtering EM algorithm

The algorithm can be expressed as follows:
(1) E-step: calculate the posterior probabilities for the Gaussian mixtures

$$p(m \mid x_k;\Theta) = \frac{\pi_m p(x_k \mid \theta_m)}{\sum_{l=1}^{K} \pi_l p(x_k \mid \theta_l)} \qquad (2)$$

Where each component density $p(x \mid \theta_m)$ is a normal probability distribution

$$p(x \mid \theta_m) = \frac{1}{(2\pi)^{1/2} \det(\sum_m)^{1/2}} \times \exp\left\{-\frac{1}{2}(x-\mu_m)^{\mathrm{T}} \sum_m^{-1}(x-\mu_m)\right\} \qquad (3)$$

Where $\theta_m = (\mu_m, \sum_m)$ is the parameters of each component.
(2) Filtering-step: perform low-pass filtering operation on these posterior probabilities

$$p_f(m \mid x_k;\Theta) = filter(p(m \mid x_k;\Theta), [w \times l]) \qquad (4)$$

Where $filter(.)$ is an operation of low-pass filtering, and the neighborhoods of size is w-by-l.
(3) M-step: calculate the new parameter estimates using the filtered posteriori probability as weights

$$\pi_m^{(t+1)} = \frac{1}{N}\sum_{k=1}^{N} p_f\left(m \mid x_k;\Theta^{(t)}\right) \qquad (5)$$

$$\mu_m^{(t+1)} = \frac{\sum_{k=1}^{N} x_k P_f\left(m \mid x_k;\Theta^{(t)}\right)}{\sum_{k=1}^{N} P_f\left(m \mid x_k;\Theta^{(t)}\right)} \qquad (6)$$

$$\sum_m^{(t+1)} = \frac{\sum_{k=1}^{N} P_f\left(m \mid x_k;\Theta^{(t)}\right)\left\{\left(x_k-\mu_m^{(t+1)}\right)\cdot\left(x_k-\mu_m^{(t+1)}\right)^{\mathrm{T}}\right\}}{\sum_{k=1}^{N} P_f\left(m \mid x_k;\Theta^{(t)}\right)} \qquad (7)$$

We use *component-wise EM for mixtures*(CEM^2)[4] to implement our algorithm by means of its feature of updating parameters sequentially. The proposed algorithm is very simple and performs as follows: Expectation→ Filtering→Maximum. So we name it as EFM algorithm for short.

3.2 Component annihilation mechanism

The low pass filtering operation can swell the components step by step. With the help of the competitive mechanism of EM algorithm, the redundant components tend to coincide with the other components. This is an ideal phenomenon, because it provides a reasonable judgment if merge operation should be performed or not. By merging the components with similar parameters, we can eliminate irrelevant components. Here, we adopt the merge criterion by the correlation coefficient of two components i and j:

$$J_{merge}(i,j;\Theta) = \frac{p_i(\Theta)^{\mathrm{T}} p_j(\Theta)}{\|p_i(\Theta)\| \|p_j(\Theta)\|} \qquad (8)$$

Where $p_i(\Theta) = (p(i \mid x_1;\Theta),..., p(j \mid x_N;\Theta))^{\mathrm{T}}$ is a N-dimensional vector consisting of the posterior probability for the i th component, and $\|\cdot\|$ denotes the Euclidean vector norm. When $J_{merge}(i,j;\Theta) \to 1$, two components of i and j are merged. Now the merge operation is very simple. Let

$$\pi_i = \pi_i + \pi_j \qquad (9)$$

And remain the other parameters of the i th component unchanged, and remove the j th component:

$$K = K - 1 \qquad (10)$$

4. EXPERIEMTNS

We start by mapping each pixel in the original image to a 6-dimensional feature vector, which consists of the same texture and color features in Blobword[1]. First, the image $I(x,y)$ is convolved with Gaussian smoothing kernels of several scales σ : $M_\sigma(x,y) = G_\sigma(x,y) * (\nabla I(x,y))(\nabla I(x,y))^{\mathrm{T}}$. Then computing the polarity at each pixel location: $P = |E_+ - E_-| / (E_+ - E_-)$, where E_+ and E_- represent the number of gradient vectors in the window $G_\sigma(x,y)$ that are on the positive and negative sides of the dominant orientation respectively. For each pixel, an optimal scale σ^* is selected. The three texture features are p_{σ^*}, anisotropy $\alpha = 1 - \lambda_2/\lambda_1$, where λ_1 and λ_2 are the eigenvalues of $M_{\sigma^*}(x,y)$, and normalized texture contrast: $c = 2\sqrt{\lambda_1 + \lambda_2}$. The three color features are the $L^*a^*b^*$ coordinates of the color image computed after smoothing the image with a Gaussian kernel at the selected optimal scale. In order to reduce their correlation, we condense the original 6-dimensional feature space to 3-dimensional by applying PCA. The size of the image in our experiments is 128×192 or 192×128 . For the present, we adopt an adaptive filter, which tailors itself to the local image variance [5]. Where the variance is large, the filter performs little smoothing. Where the variance is small, the filter performs more smoothing. At first, the filter estimates the local mean: $\mu_p = \frac{1}{N} \sum_{x_k \in \eta} p(m \mid x_k; \Theta^{(t)})$

and variance: $\delta_p^2 = \frac{1}{N} \sum_{x_k \in \eta} p^2(m \mid x_k; \Theta^{(t)}) - \mu_p^2$

around each pixel's feature vector x_k , where $\eta = [15 \times 15]$ is the local neighborhood of each pixel in the image in our experiments. The filter then creates a pixel-wise Wiener filter adaptively $p_f(m \mid x_k; \Theta^{(t)}) = \mu_p + \frac{\sigma_p^2 - \nu^2}{\sigma_p^2}$

$p(m \mid x_k; \Theta^{(t)} - \mu_p)$, where ν^2 is the noise variance. In our implementations, we chose the constant value $\nu^2 = 0.1$ for all the iteration. We demonstrate the performance of our algorithm through the following three examples.

In the first example, we demonstrate that EFM can escape from local maxima. The original image is shown in Fig.1(a). Fig.1(b) shows the segmentation result by K-means. It is a local optimum. Fig.1(c) shows the segmentation result of EM initialized by result of Fig. 1(b), and the result of EM gets trap in a local maximum. Using the parameters of Fig.1(c) as the initialization, Fig. 1(d) shows the segmentation result of our EFM algorithm, which converges to the global optimum. This is an exciting result because it is hard to escape the situation for the other extensions of EM algorithm. Fig.2 shows the clustering result in feature space, where (a),(b) and (c) correspond to Fig. 1 (b),(c) and (d) respectively.

The second example gives an example of reducing fragmentation phenomenon. The original image is shown in Fig. 3(a). Fig. 3(b) shows the result of EM. Fig. 3(c) shows the result of EFM. When the EM algorithm is used for partitioning spatial data, the result classes will often be spatially very mixed. This is because the algorithm relies on the assumption that all the underlying data class labels are independent. The independence assumption ignores the influence that one object exerts on its neighboring objects. Filtering technique provides a powerful tool to incorporate an *a priori* knowledge about the spatial statistics into the EM algorithm. So the spatially mixed situation is reduced.

At some times, model selection criterion cannot give proper image region number. In the Fig.4, we choose the two examples with bad segmentation results of Blobworld[1] and also give the results of our algorithm. The largest times of iteration of EM algorithm is set to be 15. For Blobworld, the threshold is set to be 10^{-2}. For our algorithm, the threshold is set to be 10^{-5} . In Blobworld, K ranges from 2 to 5. For each K , Blobworld starts with 4 different initializations, and selects the best result as candidate. At last, MDL criterion is used to choose the suitable number of region. In order to take account of spatial information, Blobworld integrates the coordinates of each pixel in feature vectors. So the dimension of feature vectors used by Blobworld is 8. To keep the covariance matrices from becoming singular, Blobworld adds some noise. EFM is initialized by means of K-means, and the initial K is set to be 6. Fig.4 (a), (d) show the original images. The segmentation results of Blobworld are given in Fig. 4(b), (e). Fig.4(c), (f) show the results of our algorithm. Under the segmentation results, we also give the region number and the computation time on 1.7G PC.

5. CONCLUSION

The paper presents a filtering EM (EFM) algorithm for image segmentation based on finite mixture models. By integrating low pass filtering operation into EM algorithm,

the proposed algorithm is less sensitive to initialization than the standard EM algorithm. By merging the components with similar parameters, the proposed algorithm can perform model selection automatically. Filtering EM algorithm presented in the paper is a simple and flexible extension to EM for clustering spatial data. It preserves the good features of EM and corrects the independence assumption of EM to some extent.

Acknowledgement: The authors would like to thank Chad Carson and the rest of the Blobworld team for making the Blobworld Matlab code publicly available. The authors would also like to thank UCB Computer Vision Group for making the Berkely Segmentation Dataset available.

5. REFERENCES

[1] Chad Carson, Serge Belongie, Hayit Greenspan and Jitendra Malik, Blobworld: Color- and Texture-Based Image Segmentation Using EM and Its Application to Image Querying and Classification， *IEEE Trans. Pattern Anal. and Mach.. Intell.* 24(2002):1026-1038.

[2]Figueiredo, M. A. T. and Jain, A. K. Unsupervised learning of finite mixture models, *IEEE Trans. Pattern Anal. and Mach. Intelli.,* 24(2002):381-396.

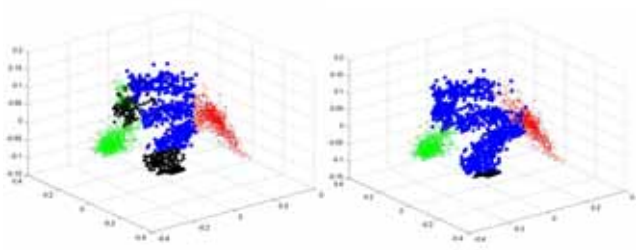
[3] A.P. Dempster, N.M. Laird, D.B. Rubin, maximum-likelihood from incomplete data via the EM algorithm, *J. R. Statist. Soc.B* .39(1977):1－38.

[4] G. Celeux, S. Chretien, F. Forbes, A. Mkhadri, A component-wise EM algorithm for mixtures, Technical report 3746, INRIA Rhone-Alpes, Frances, 1999.

[5] Lim, Jae S. Two-Dimensional Signal and Image processing. Englewood Cliffs, NJ: Prentice Hall, 1990. pp. 536-540.

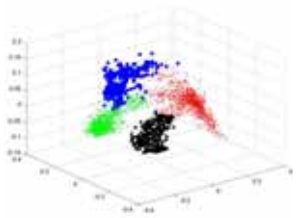
（a） (b) (c) (d)
Original image K-means EM EFM

Fig. 1 Example 1



(a) (b)
K-means EM

Fig.2 Clustering result in feature space



(c)
EFM

Fig.2 Clustering result in feature space

（a） （b） （c）
Original image EM EFM

Fig.3 Example 2

Original image K=5,t=55.98s Blobworld K=5,t= 22.26s Our algorithm Original image K=5,t=37.51s Blobworld K=4,t=14.79s Our algorithm
(a) (b) (c) (d) (e) (f)

Fig. 4 Example 3