Foreground/Background Segmentation with Probabilistic Graphical Models

Weijian Zhou[*]
997107018
keith.zhou@mail.utoronto.ca

ChenXing Zhao [†]
995965139
chenxing.zhao@mail.utoronto.ca

Wenzhangzhi Guo[‡]
997573353
wenzhi.guo@mail.utoronto.ca

Abstract

In this paper, we experiment with four different inference and learning models on an image segmentation problem. The goal of the project is to segment each image into foreground pixels and background pixels. The foreground and background objects in each image come from a known library. In this case, there are five different foreground objects and seven different background objects. The four inference algorithm we look at in this paper include iterated conditional modes (ICM), exact expectation-maximization (EM), Gibbs sampling EM, and variational (mean-field) EM. The basic idea is to approximate the joint distribution of the visible and hidden variables with a Q-distribution. Different inference algorithms put slightly different restrictions on the Q-distribution. This Q-distribution can be found by minimizing the free-energy (KL-divergent of the posterior distribution and the Q-distribution).

1 Introduction

In recent years, there is increasing interest in solving machine learning problems with probabilistic graphical models. Probabilistic graphical models not only provide a simple way to visualize the interactions between different variables in a model, they also allow complex probabilistic computations to be expressed as graphical manipulations [1]. In a graphical model, two of the main issues we are trying to solve are inference (compute the expectation of some nodes given some observed nodes) and learning (setting the parameters of the model given some data). In this paper, we are going to follow the approaches in [2] to compare four different inference and learning algorithms: iterated conditional modes (ICM), exact expectation-maximization (EM), Gibbs sampling EM, and variational (mean-field) EM. In order to compare the above four algorithms quantitatively, those four algorithms were tested against a simple image segmentation problem where we are trying to describe pictures as a composition of foreground and background objects, selected from a pre-defined library. Segmenting an image into foreground and background can be helpful to solve many other computer vision problems such as object classification [2].

All four inference and learning algorithms mentioned above have been widely used in many different image segmentation problems. The basic idea is to approximate the joint distribution of the visible and hidden variables with a Q-distribution. ICM is initially introduced by Besag in 1983 [3]

[*]Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON, M5S 3G4
[†]Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON, M5S 3G4
[‡]Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON, M5S 3G4

and since then, the original algorithm and its variants have been applied to many different image segmentation problems [4] [5] [6]. In general, ICM iterates between finding the optimal value of each hidden variable h_i and finding the optimal values for the model parameters [2]. On the other hand, the three EM algorithms accounts for uncertainty in the random variables in the update rules [2]. For the exact EM, there is no restrictions on the Q-distributions, for the Gibbs Sampling EM, we use sampling to update the hidden variables in the E-step, and mean-field EM assumes the Q-distribution can be fully factorized. EM algorithms are also widely used for image segmentation problems [7] [8] [9].

2 Algorithms

2.1 Data generation

A training image or testing image used in this paper is generated using the following procedure:

- Randomly select a foreground image. Each foreground image has a fixed mask which specifies the location of the foreground image within the background image.
- Randomly select a background image.
- Combine the foreground image and background image with the cut-out mask.
- Add some Gaussian noise. The Gaussian noise is added to model sensor noise and change in illumination so that we can have more data available in the training/testing stage. This is image that is shown to our model in the training and testing stage.

A graphical illustration of the above procedures can be seen in Fig 1.

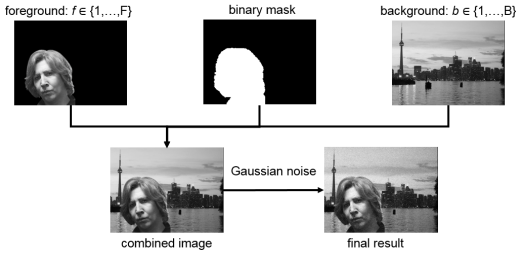

Figure 1: Overview of data generation

2.2 Model

In this project, we are using an occlusion model to describe each image generated in 2.1, where the foreground object and background object in each image come from the same library with J objects. If we assume each image has K pixels, then we can denote the pixel intensity in an image by $z_1, \ldots, z_K$. The probability of choosing f^{th} foreground object is $P(f)$. Then depending on the foreground object, a binary mask is constructed $m = (m_1, \ldots, m_K)$. m_i is either 0 (pixel z_i is a background pixel) or 1 (pixel z_i is a foreground pixel). The binary must cut out the foreground object, but given the foreground object, the mask RVs are independent. Similarly, the probability of choosing b^{th} background object is $P(b)$. Lastly, when given the mask, the foreground object class and background object class, the intensities of the pixels in an image are independent. Therefore, the joint distribution of the model can be written as:

$$P(z, m, f, b) = P(b)P(f)\left(\prod_{i=1}^{K} P(m_i|f)\right)\left(\prod_{i=1}^{K} P(z_i|m_i, f, b)\right) \tag{1}$$

Since the mask RVs (m_i) are binary, the last term in equation 1 can be factorized. When $m_i = 0$, the pixel intensity (z_i) only depends on the background object index b, whereas when $m_i = 1$, z_i only

depends on the foreground object index *f*. Therefore, equation 1 can be re-written as:

$$P(z,m,f,b) = P(b)P(f)\left(\prod_{i=1}^{K} P(m_i|f)\right)\left(\prod_{i=1}^{K} P(z_i|f)^{m_i}\right)\left(\prod_{i=1}^{K} P(z_i|b)^{1-m_i}\right) \tag{2}$$

To reduce the computation complexity, it is often useful to parametrize the model. In this case, we assume the pixel intensities in each image follows a Gaussian distribution, where given a foreground or background object index *k*, the pixel intensity z_i equals to μ_{ki} plus zero-mean Gaussian noise with a variance of φ_{ki}. This added noise would account for the noise we added to each of the images. We also denote the probability of class *k* by π_k. Lastly, we denote the probability of $m_i = 1$, given the foreground class is *f* be α_{fi}. Using this parametrization, equation 2 can be written as:

$$P(z,m,f,b) = \pi_b\pi_f\left(\prod_{i=1}^{K} (\alpha_{fi})^{m_i}(1-\alpha_{fi})^{1-m_i}\mathcal{N}(z_i;\mu_{fi},\varphi_{fi})^{m_i}\mathcal{N}(z_i;\mu_{bi},\varphi_{bi})^{1-m_i}\right) \tag{3}$$

where $\mathcal{N}$ represent the Gaussian distribution.

2.3 Finding model parameters by minimizing the free energy

The model parameters are found based on each training image. In each training image, we can group the variables into hidden RVs, visible RVs and parameters. The visible RVs in this case are the pixel intensities: $v^{(t)} = z^{(t)}$. The hidden RVs are $h^{(t)} = (f^{(t)}, b^{(t)}, m^{(t)})$. The model parameters are $\theta = (\mu, \varphi, \pi, \alpha)$. Then the joint distribution of the visible and hidden RVs can be written as:

$$P(h,v) = P(\theta)\prod_{t=1}^{T} P\left(h^{(t)}, v^{(t)}|\theta\right) \tag{4}$$

where (t) represents the index of the training cases.

Usually, it is very hard to find the posterior distribution P(*h*|*v*) directly since the calculation for the partition function is intractable. Therefore, we often use approximate inference techniques instead. The basic idea of those techniques is that we are trying to use a simpler *Q*-distribution Q(*h*) approximate the true posterior distribution P(*h*|*v*). The free energy is a measurement of similarity between the true posterior distribution and the *Q*-distribution we have currently. The free energy can be defined as:

$$\begin{aligned} F(Q,P) &= KL(Q,P) - \ln P(v) \\ &= \int_h Q(h)\ln\frac{Q(h)}{P(h|v)} - \int_h Q(h)\ln P(v) \\ &= \int_h Q(h)\ln\frac{Q(h)}{P(h,v)} \end{aligned} \tag{5}$$

where KL(*Q*,*P*) is the KL-divergent between the *Q*-distribution and the posterior distribution P(*h*|*v*). The term ln P(*v*) is added to make the calculation tractable (since we do not have a tractable form of P(*h*|*v*)), but it does not have a direct impact final solution for the *Q*-distribution (since P*v* does not depend on the *Q*-distribution). Since the KL divergence is a similarity measure of two distributions, minimizing the free energy F(*Q*,*P*)) makes the *Q*-distribution more similar to the posterior distribution P(*h*|*v*). Therefore, the goal of the inference problem is to search through the space to find Q(*h*) that resembles the true posterior distribution closely.

If we assume the training data is i.i.d, we can de-couple the free energy so that we have one term for each training case. In this case, we can substitute equation 4 into equation 5 and obtain:

$$F(Q,P) = \int_h Q(h)\ln Q(h) - \int_\theta Q(\theta)\ln P(\theta) - \sum_{t=1}^{T}\int_{h^{(t)}\theta} Q(h^{(t)},\theta)\ln P\left(h^{(t)},v^{(t)}|\theta\right) \tag{6}$$

2.4 Iterated Conditional Modes (ICM)

The ICM algorithm alternates between updating the hidden variables and the model parameters. In the ICM update procedure, we are only considering local information. We update one variable/parameter at a time by setting it to its Maximum a posterior (MAP) value while holding other

variables/parameters constant. Since we are using the MAP estimation where there is a hard-assignment for each pixel (it either belongs to the foreground or the background class), the Q-distribution is a product of delta functions:

$$\begin{aligned} Q = &\left(\prod_{k} \delta(\pi_k - \hat{\pi}_k)\right)\left(\prod_{k,i} \delta(\mu_{ki} - \hat{\mu}_{ki})\right)\left(\prod_{k,i} \delta(\varphi_{ki} - \hat{\varphi}_{ki})\right)\left(\prod_{k,i} \delta(\alpha_{ki} - \hat{\alpha}_{ki})\right) \\ &\left(\prod_{t} \left[b^{(t)} = \hat{b}^{(t)}\right]\right)\left(\prod_{t} \left[f^{(t)} = \hat{f}^{(t)}\right]\right)\left(\prod_{t}\prod_{i} \left[m_i^{(t)} = \hat{m}_i^{(t)}\right]\right) \end{aligned} \tag{7}$$

Substituting equation 7 and 3into equation 6, taking the derivative with respect to each parameter/ variable and set to zero, we can find the update rule for the ICM model as follows (Repeat until convergence):

E-step (hidden variable update)

for t = 1, . . . , T:

$$f \leftarrow \text{argmax}_f \left(\pi_f \prod_{i=1}^{K} (\alpha_{fi})^{m_i} (1-\alpha_{fi})^{1-m_i} \mathcal{N}(z_i; \mu_{fi}, \varphi_{fi})^{m_i}\right)$$

for i = 1, . . . , K:

$$m_i \leftarrow \text{argmax}_{m_i} = \begin{cases} \alpha_{fi}\mathcal{N}(z_i; \mu_{fi}, \varphi_{fi}), & \text{if } m_i = 1 \\ (1-\alpha_{fi})\mathcal{N}(z_i; \mu_{bi}, \varphi_{bi}), & \text{if } m_i = 0 \end{cases}$$

$$b \leftarrow \text{argmax}_b \left(\pi_b \prod_{i=1}^{K} \mathcal{N}(z_i; \mu_{bi}, \varphi_{bi})^{m_i}\right)$$

M-step (model parameter update)

For j = 1, . . . , J:

$$\pi_j \leftarrow \frac{1}{2T} \sum_{t=1}^{T} ([f^{(t)} = j] + [b^{(t)} = j])$$

For j = 1, . . . , J and For i = 1, . . . , K:

$$\alpha_{ji} \leftarrow \frac{\sum_{t=1}^{T} [f^{(t)} = j] m_i^{(t)}}{\sum_{t=1}^{T} [f^{(t)} = j]} \qquad \mu_{ji} \leftarrow \frac{\sum_{t=1}^{T} [f^{(t)} = j \text{ or } b^{(t)} = j] z_i^{(t)}}{\sum_{t=1}^{T} [f^{(t)} = j \text{ or } b^{(t)} = j]}$$

$$\varphi_{ji} \leftarrow \frac{\sum_{t=1}^{T} [f^{(t)} = j \text{ or } b^{(t)} = j](z_i^{(t)} - \mu_{ji})^2}{\sum_{t=1}^{T} [f^{(t)} = j \text{ or } b^{(t)} = j]}$$

The ICM algorithm is really easy to implement, but it can get stuck at local optima very easily. Thus, this algorithm does not work well in many cases in practice.

2.5 Exact Expectation-Maximization (EM)

Unlike ICM, the EM algorithm takes into account for uncertainty in the RVs in the update process. The Q-distribution for the i.i.d training cases can be written as:

$$Q(h) = \delta(\theta - \hat{\theta}) \prod_{t=1}^{T} Q(h^{(t)}) \tag{8}$$

Again, substituting equation equation 8 into equation 6, taking the derivative with respect to each parameter/ variable and set to zero, we can find the update rule for the ICM model as follows (Repeat until convergence):

E-step

for t = 1, . . . , T:

$$Q(b,f) \leftarrow c_2 \pi_b \pi_f \prod_{i=1}^{K} \left(\alpha_{fi} \mathcal{N}(z_i; \mu_{fi}, \varphi_{fi}) + (1-\alpha_{fi}) \mathcal{N}(z_i; \mu_{bi}, \varphi_{bi}) \right)$$

$$Q(b) \leftarrow \sum_f Q(b,f) \qquad Q(f) \leftarrow \sum_b Q(b,f)$$

for i = 1, . . . , K:

$$Q(m_i = 1|b,f) \leftarrow c_1 \alpha_{fi} \mathcal{N}(z_i; \mu_{fi}, \varphi_{fi}) \quad Q(m_i = 0|b,f) \leftarrow c_1 \alpha_{fi} \mathcal{N}(z_i; \mu_{fi}, \varphi_{fi})$$

for i = 1, . . . , K:

$$Q(m_i, b) \leftarrow \sum_f Q(m_i|b,f) Q(b,f) \quad Q(m_i, f) \leftarrow \sum_b Q(m_i|b,f) Q(b,f)$$

M-step

For j = 1, . . . , J:

$$\pi_j \leftarrow \frac{1}{2T} \sum_{t=1}^{T} \left(Q(f^{(t)} = j) + Q(b^{(t)} = j) \right)$$

For j = 1, . . . , J and For i = 1, . . . , K:

$$\alpha_{ji} \leftarrow \frac{\sum_{t=1}^{T} Q(m_i^{(t)} = 1, f^{(t)} = j)}{\sum_{t=1}^{T} Q(f^{(t)} = j)}$$

$$\mu_{ji} \leftarrow \frac{\sum_{t=1}^{T} \left(Q(m_i^{(t)} = 1, f^{(t)} = j) + Q(m_i^{(t)} = 0, b^{(t)} = j) \right) z_i^{(t)}}{\sum_{t=1}^{T} \left(Q(m_i^{(t)} = 1, f^{(t)} = j) + Q(m_i^{(t)} = 0, b^{(t)} = j) \right)}$$

$$\varphi_{ji} \leftarrow \frac{\sum_{t=1}^{T} \left(Q(m_i^{(t)} = 1, f^{(t)} = j) + Q(m_i^{(t)} = 0, b^{(t)} = j) \right) (z_i^{(t)} - \mu_{ji})^2}{\sum_{t=1}^{T} \left(Q(m_i^{(t)} = 1, f^{(t)} = j) + Q(m_i^{(t)} = 0, b^{(t)} = j) \right)}$$

2.6 Gibbs Sampling EM

Gibbs sampling EM is very similar to ICM except it uses Gibbs sampling to update the hidden variables in the E-step (The M-step in this case is the same as ICM, but we use Gibbs sampling method to update the hidden variables instead of using the MAP value).Since this sampling approach is a stochastic process, it helps the algorithm to jump out from local optima and thus improving the overall performance of the algorithm.

2.7 Variational (Mean field) EM

In the variational EM, we assume the hidden variables in the Q-distribution factorizes completely:

$$Q(h) = \prod_{i=1}^{L} Q(h_i) \tag{9}$$

In this case, the M-step is exactly the same as the exact EM algorithm, but the E-step can be simplified to:

E-step

for t = 1, . . . , T:

$$Q(f) \leftarrow c_3 \pi_f \prod_{i=1}^{K} \left(\left(\alpha_{fi} \mathcal{N}(z_i; \mu_{fi}, \varphi_{fi}) \right)^{Q(m_i=1)} (1 - \alpha_{fi})^{Q(m_i=0)} \right)$$

for i = 1, . . . , K:

$$Q(m_i = 1) \leftarrow c_1 \prod_{f} \left(\alpha_{fi} \mathcal{N}(z_i; \mu_{fi}, \varphi_{fi}) \right)^{Q(f)}$$

$$Q(m_i = 1) \leftarrow c_1 \left(\prod_{f} (1 - \alpha_{fi})^{Q(f)} \right) \left(\prod_{b} \mathcal{N}(z_i; \mu_{bi}, \varphi_{bi})^{Q(b)} \right)$$

$$Q(b) \leftarrow c_2 \pi_b \prod_{i=1}^{K} \mathcal{N}(z_i; \mu_{fi}, \varphi_{fi})^{Q(m_i=0)}$$

2.8 Other Inference Algorithms

Structured variational EM and sum-product EM can also be used to solve the image segmentation problem described in this paper. However, we did not have time to implement and test them due to time constraint. Unlike mean-field approximation, the structured variational EM takes into account some dependencies between the hidden RVs. In general, if more dependencies are included in the model, we will have more exact inference, but the computational complexity also increases. The sum-product algorithm performs inference by message passing. If the graphical model is a tree, all the posterior marginal distributions will be exact once we have passed the messages from the leaves to the root and then from the root back to the leaves. However, in out case, the graphical model constructed for this problem has cycles. The messages need to be passed through the model several times until convergence. The details of the update equations for these two models can be found in [2].

3 Experimental Results

3.1 Data set

The data set used for this project can be seen in Fig 2. There were five foreground objects and seven background objects used to create the training/testing images.

Figure 2: Data set used this project. first row: foreground images. second row: background images (resized). third row: example of training images

3.2 Results

4 References

[1] Bishop, C.M., "Graphical Models," in *Pattern recognition and machine learning,* New York, NY, USA: Springer Science+Business Media, LLC, 2006, ch. 8, pp. 359-360.

[2] Frey, B.J. and Jojic, N., "A comparison of algorithms for inference and learning in probabilistic graphical models," *IEEE Trans. Pattern Anal. Mach. Intell.,* vol. 27, no. 9, pp. 13921416, Sept. 2005.

[3] Besag, J.E. “Discussion of paper by P. Swiller,” *Bull. Inr. Stafisr. Inst.,* vol. 50, no. 3, pp. 422-425, 1983.

[4] Loog, M. and van Ginneken, B., “Supervised segmentation by iterated contextual pixel classification, *Image Signal Process., ser. Proc. 16th Int. Conf. Pattern Recognition (ICPR 2002),* pp. 925928, 2002.

[5] Owen, A., “Image segmentation via iterated conditional expectations,” Department of Statistics, Stanford University, Stanford, CA, Tech. Rep. 18 (94305-4065), June, 1989.

[6] Chen, C.W.; Luo, J.B. and Parker, K.J., “Image segmentation via adaptive K-mean clustering and knowledge-based morphological operations with biomedical applications, *IEEE Trans. Imag. Processing,* vol. 7, pp. 16731683, Dec. 1998.

[7] Gu, D.B.and Sun, J.X., “EM image segmentation algorithm based on an inhomogeneous hidden MRF model, *IEEE Proc. Vision, Image and Signal Processing,* vol. 152, Issue 2, pp. 184 190, April 2005.

[8] Geman, S. and Geman, D., “Stochastic Relaxation, Gibbs Distribution and the Bayesian Restoration of Images, *IEEE Trans. Pattern Anal. Mach. Intell.,* vol. 6, pp. 721-741, 1984.

[9] Cho, W.-H.; Kim, S.-H.; Park, S.-Y. and Park, J.-H., “Mean field annealing EM for image segmentation,” *Proc. 2000 Intl Conf. on Image Processing,* vol. 3, pp. 568-571, Sep. 2000.