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). Based on our experimental result, exact EM achieved the best result, but it also took longer to run. ICM took very little computational time, but it returned the worst result.

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

[*]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

visible and hidden variables with a Q-distribution. ICM is initially introduced by Besag in 1983 [3] 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] [10].

2 Algorithms

2.1 Data generation

A training 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 for training. This is image that is shown to our model in the training process.

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

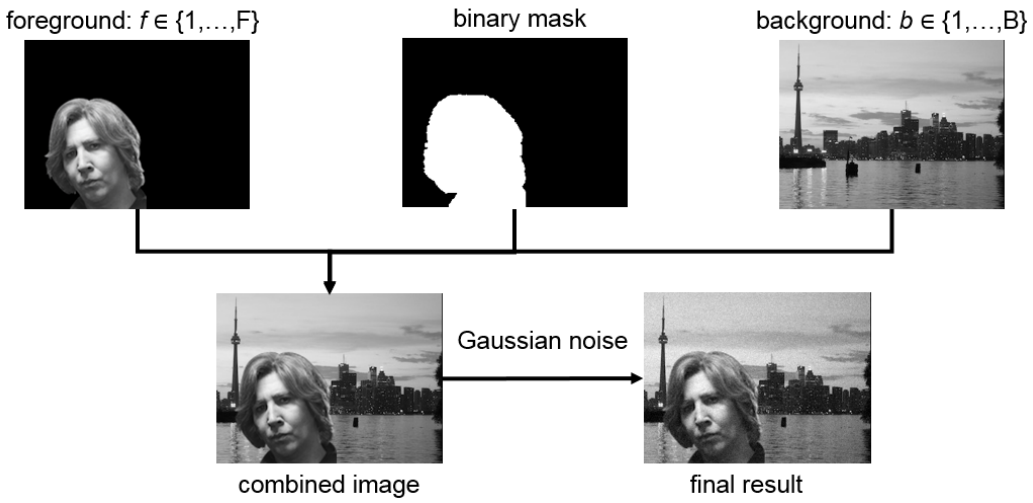

Figure 1: Overview of data generation

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 ψ_{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:

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

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 $\theta = (\mu, \psi, \pi, \alpha)$. Then the joint distribution of the visible and hidden RVs can be written as:

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

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:

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

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:

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

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:

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

for i = 1, . . . , K:

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

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

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:

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

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

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:

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

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

for i = 1, . . . , K:

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

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:

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

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

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

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:

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

for i = 1, . . . , K:

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

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

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

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 images. A total of 350 images (10 noisy images of each combination of the foreground and background images) were used as training images for this project. Each inference algorithm was run 10 times with different random initializations and the result with the highest likelihood was reported.

3.2 Parameter Learning Results

A sample of the learned parameters for each inference algorihtm can be seen in Fig 3. The model assumes there are 14 clusters (2 more than the sum of total number of foreground and background objects in the library) to give the model more flexibility. For the ICM, we can see that the mask

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

probablities are mostly random. The combination of the mask probabilities with the mean image (second column) does not produce any sensible foreground image. Thus, the ICM does not produce very good segmentation results. This is mainly due to the fact that ICM gets stuck at local optima easily. Once it gets stuck, there is no way to get out of it. For the exact EM, the performance is much better. The model learned accurate mask for class 1,6, and 7 and some partially accurate mask for class 2 and 3. Since there are 5 different foreground objects in the library, we expect that there are 5 classes with clean mask (classes with clean masks are foreground images, and the rest are background images). Thus, the result of the exact EM is fairly close to what we expected. For the mean-field EM, class 9 and class 13 produces mask that is roughly in the correct location (with some noise). Therefore, the performance of mean-field EM is worse than the exact EM. The mean-field EM simplifies the exact EM model by assuming the Q-distribution completely factorizes. This assumption ignores dependencies between hidden RVs, causing it to perform worse than the exact EM. However, in more complicated problems, this simplification can greatly reduce the computational complexity. Lastly, for Gibbs sampling EM, class 1 and class 5 each produces a very crude estimation of the mask. The masks produced in those two classes are not very accurate in terms of location and are very blurry. However, this result is much better than the result produced by ICM. This is because performing Gibbs takes uncertianties into account, which helps the model to get out of local optima it encounters in certain cases (whereas in ICM, the hidden RVs are updated based on their MAP value, so the model cannot esacpe those local optima). Therefore, based on the plot in Fig 3, twe can rank the performance of the four models as follows (from best to worst): exact EM, mean-field EM, Giibs sampling EM, and ICM.

Comparing to the result in [2], a similar trend is observed. ICM failed to find any sensible classes. The exact EM performs the best where it was able to successfully infer all foreground and background classes correctly. The mean-filed was able to infer the foreground and background classes correctly, except sometimes there are cutting holes in the learned mask. However, the results obtained in [2] for the exact EM and mean-field EM are better than what we were able to get in this project. The possible explanations for this difference could be because of the different dataset that was used and the differences in the implementation details.

3.3 Free Energy Results

The free energy across different iterations can be seen in Fig 4. Since the free energy is a measure of the similarity between the Q-distribution and the true posterior distribution, a well trained model should have low free energy. For all four inference algorithms, the free energy decreases very quickly in the first 5 interations, then converges to a relatively stable value afterwards. The initial drastic change in the free energy is due to the fact that all the variables were intialized randomly, so most of them are far from optimal values (causing the system to have a huge free energy intially). Thus, those variables need to be changed by a lot initially to get closer to their optimal values (as the Q-distribution gets closer to the true posterior distribution), causing a big decrease in the free

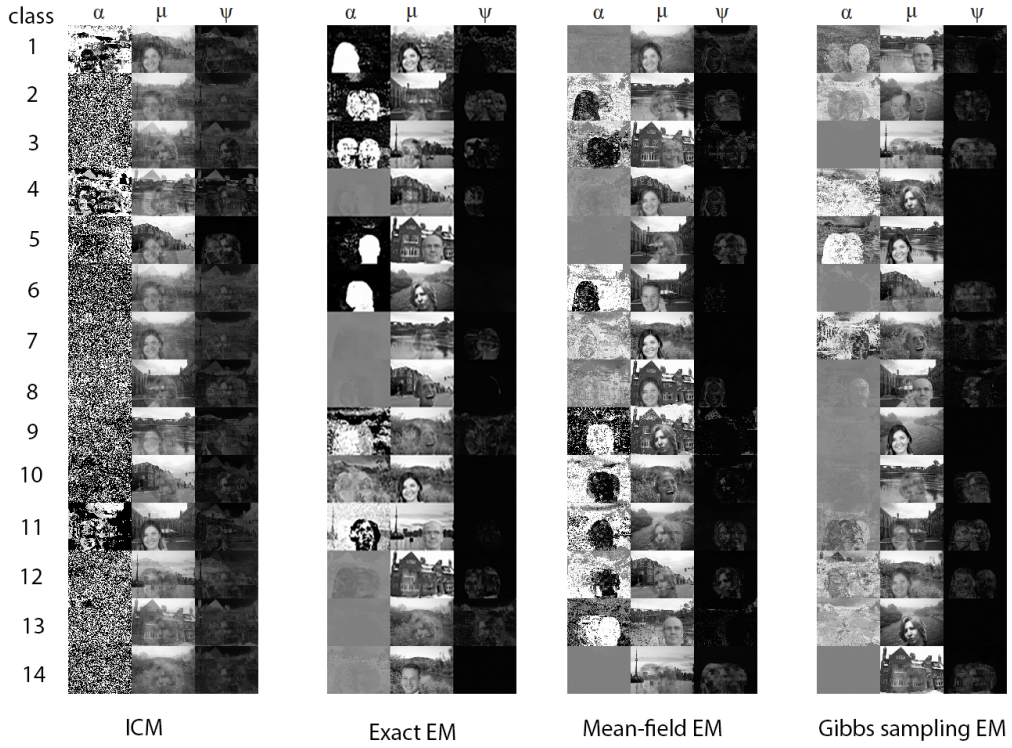

Figure 3: Comparison of the learned parameters using four different inference algorithms. α represents the mask probability for each pixel (white represents a high probability), μ represents the pixel means and ψ represents the pixel variances

energy. After 5 iterations, the hidden variables and parameters are at their optimal values mostly, so most of them will not change across iterations or only very tiny adjustments are made. As a resut, the free energy stays relatively stable at this point.

The free energy of ICM converges to its steady state value the quickest while the Gibbs sampling EM takes the longest to converge. This is because ICM algorithm quickly gets stuck at local optima where the Gibbs sampling EM tries to get out of local optima by taking uncertainties into account.

There are some oscillations seen in the free energy curve for Gibbs sampling EM. Those oscillations are cases where the hidden RVs were not updated based on their MAP value by taking the uncertainties into account. Even though taking uncertainties into account makes the free energy increases sometimes, those uncertainties enable the inference algorithm to escape from local optima, which lead to a better solution. However, the free energy of Gibbs sampling EM is still lower than that of ICM.

The trend seen in the free energy plot matches the trend seen in 3.2. The best performing model (exact EM) has the lowest free energy while the worst performing model (ICM) has the highest free energy after the model converges. The trend seen in our free energy plot matches the free energy plot in [2]. However, the exact range of the free energy curve is not the same. This is due to the fact that the training images in our case has different dimensions than the ones used in [2] and we used 350 training images instead of 300 training images.

3.4 Runtime Analysis

The runtime of each algorithm is high depedent on the implementation and hardware. For this project, the run time for each algorithm can be seen in Table 1. The ICM has the smallest runtime while exact EM has the largest runtime. This result matches the complexity analysis in [2] where

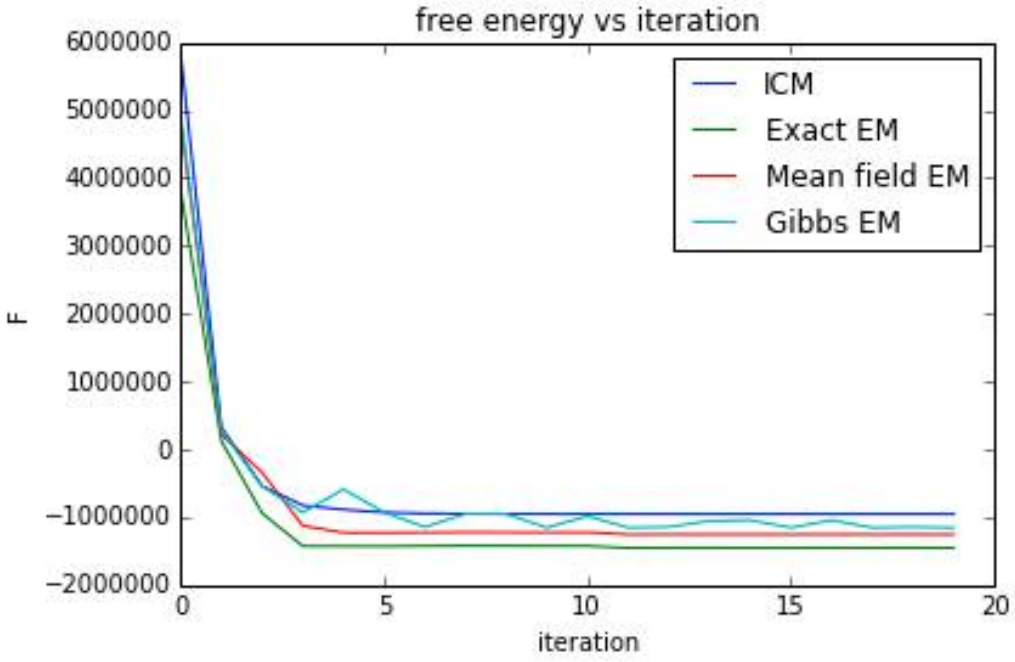

Figure 4: Comparison of the free energy for the four inference algorithms acorss iterations

the exact inference has the highest complexity and ICM has the lowest complexity. However, even though Gibbs sampling and ICM has the same complexity in [2], the runtime of Gibbs sampling is larger than that of ICM in the implementation. This is because it takes longer to collect samples in the Gibbs sampling EM than just using the MAP value.

Table 1: Runtime Comparison

	ICM	Exact EM	Mean-field EM	Gibbs Sampling EM
runtime per iteration (sec)	4.09	32.34	20.51	9.26

4 Conclusions

In this project, we looked at four different inference algorithms (ICM, exact EM, mean-filed EM, and Gibbs sampling EM) and compared their performance with a simple image segmentation problem. Exact EM has the best performance, but it also has the largest runtime. Mean-filed EM reduces the complexity of exact EM by assuming the Q-distribution can be completely factorized. This assumption reduces the runtime, but also worsen the performance a little bit. ICM has the fastest runtime, but it has very poor performace. The performance of ICM can be improved by taking uncertainty into account, which leads to the Gibbs sampling EM.

In the future, we plan to implement structural variational inference and sum-product EM as well and test their performance on this problem. Additionally, we plan to apply those inference algorithms to other image segmentation problems.

5 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] Zhou, Y.; Gong, Y. and Tao, H. "Background segmentation using spatial-temporal multi-resolution MRF, *Proc. 7th IEEE Workshop (WACV/MOTIONS '05),* vol. 1, pp. 813, Jan. 2005.

[10] 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.