Project4: image segmentation using MRF

Zeming Fang
661958922

1. Introduction

In the previous homework and projects, we extensively discussed the Bayesian network (BN), the directed network, as one of two types of probabilistic graphic network (PGM). In this new project, we are going to explore an alternative undirected PGM, the Markov Random Field (MRF).

An MRF can be defined as a two-tuple: $\{\mathcal{G}, \Theta\}$, where $\mathcal{G}$ defines the structure of the network, including node-set $\mathcal{V} = \{X_1, X_2, ..., X_N\}$ and edge-set $\mathcal{E} = \{E_{ij}\}$ represents the undirected edges between arbitrary nodes i and j. Θ defines the parameters for the node-set, the unary function $\phi(X_i)$, and edge-set, the potential function $\psi(X_i, X_j)$.

For a given node X_i, we can call the set of nodes that have directed edge to X_i as the neighbor of X_i, denoted as N_{X_i}. The basic assumption of MRF is the Markov condition, the mathematical expression of which is (Ji 2019, equation 4.1):

$$X_i \perp X_j | N_{X_i}, \quad \forall X_j \in \mathrm{X} \backslash N_{X_i} \backslash X_i. \tag{1}$$

In another word, given the neighbour of a node X_i, we can exactly describe X_i.

Calculating the joint probability distribution is one fundamental question in PGM. According to the Hammersley-Clifford (HC) theorem (Zhang and Ji, 2009), the joint probability distribution of MRF can be calculated as:

$$p(x_1, x_2, ..., x_N) = \frac{1}{Z} \prod_{c \in C} \psi_c(\mathrm{x}_c) \tag{2}$$

where each maximal clique c is a element of C, the set of maximal cliques (see more detail in Ji (2019)). x_c means all the nodes in maximal clique c, and $psi_c(\mathrm{x}_c)$ is the potential function for clique c. Z is the partition function that normalizes the product of all maximal clique potential function, $Z = \sum_{x_1, x_2, ..., x_N} \prod_{c \in C} \psi_c(\mathrm{x}_c)$.

The computation complexity of the MRF joint probability distribution mainly depends on the form of potential functions. To ensure a practical MRF, we usually need to make assumptions to simplify MRF potential functions.

1.1 Log-linear potential function

One simple and practical potential functions simplification approach is to restrict them as log-linear function:

$$\psi_c(\mathrm{x}_c) = \exp(-w_c E_c(\mathrm{x}_c)) \tag{3}$$

where $E_c(\mathrm{x}_c$ is called the energy function for clique c. The log-linear function includes an exponential operation and a linear weighed energy function. Due to the good property (conjugate property) of the log-linear function, the joint probability distribution of MRF is still in the form of exponential linear (Gibbs distribution):

$$p(x_1, x_2, ..., x_N) = \frac{1}{Z} \exp[-\sum_{c \in C} \mathrm{w}_c E_c(\mathrm{x}_c)] \tag{4}$$

1.2 Pairwise Markov Random Field

Another common simplification method is to carefully design the model structure to ensure the maximal clique size is two. This kind of model is called pairwise MRF. In a pairwise model, we can further rewrite the equation 4 as:

$$p(x_1, x_2, ..., x_N) = \frac{1}{Z} \exp[-\sum_{(x_i, x_j) \in \mathcal{E}} w_{ij} E_{ij}(x_i, x_j)] \tag{5}$$

In practice, a bias potential function, called unary function, is often added as the prior knowledge of the MRF (Ji, 2019). Because of that, we should further extend equation 5 as:

$$p(x_1, x_2, ..., x_N) = \frac{1}{Z} \exp[-\sum_{(x_i, x_j) \in \mathcal{E}} w_{ij} E_{ij}(x_i, x_j) - \sum_{x_i \in \mathcal{V}} \alpha_i E_i(x_i)] \tag{6}$$

1.3 Potts model

The Potts model is special pairwise MRF, which rigidly defines pairwise energy function (Ji, 2019, equation. 4.9):

$$E(x_i, x_j) = 1 - \delta(x_i, x_j) \tag{7}$$

where δ is the Kronecker function, which equals to 1 when the inputs are equal, otherwise equals to 0. The pairwise energy of the Potts model encourages local smoothness (Ji, 2020b, page. 19).

1.4 label-observation model

The label-observation model is one typical model among the special designed pairwise models. The structure of the label-observation model (see figure 1) ensure that all the potential

functions take two nodes as input. However, the edges between nodes X_i and its corresponding Y_i is different from those between different nodes. Instead of being undirected edges, edges between X_i and Y_i are directed edges pointing from X_i to Y_i (Ji, 2019, page. 137). Thus, we usually treated the edges between pair (X_i, Y_i) as the unary function of X_i. The MRF joint probability distribution of Potts model can be calculated as (Ji, 2019, equation. 4.15):

$$p(x_1, x_2, ..., x_N) = \frac{1}{Z} \exp[- \sum_{(x_i,x_j)\in\mathcal{E}} w_{ij} E_{ij}(x_i, x_j) - \sum_{x_i\in\mathcal{V}} \alpha_i E_i(y_i|x_i)] \tag{8}$$

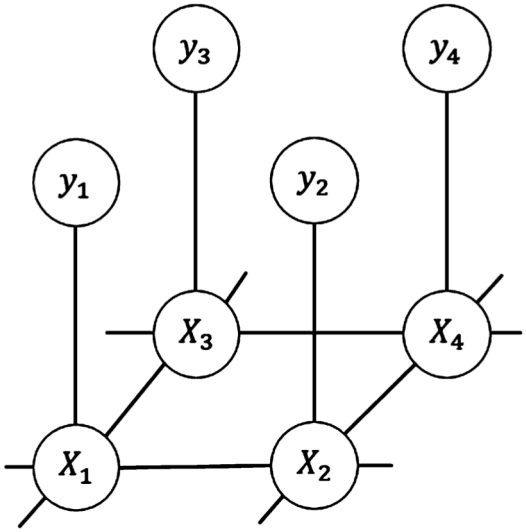


Figure 1: (Adopted from (Ji, 2019, figure. 4.5)) the structure of the label-observation model

2. Inference in pairwise label-observation model

There are three types of inference: posterior probability inference, MAP inference, and likelihood inference. Image segmentation task falls into MAP inference, where we need to find the most possible explanation of what type a pixel is given our observation (the value of the node). Thus, we will focus on MAP inference discussion in this section.

The objective function of MAP inference is,

$$\mathrm{x}^* = \arg\max_{\mathrm{x}} p(\mathrm{x}|\mathrm{y}) \tag{9}$$

For a label-observation model using log-linear potential function, equation 9 is equivalent to (see equation 15 in appendix for details),

$$\mathrm{x}^* = \arg\min_{\mathrm{x}} E(\mathrm{x}, \mathrm{y}) \tag{10}$$

To tackle equation 10, we can use directed methods, which are variables elimination (VE), belief propagation (BP), the junction tree, and the graph cuts; and approximation methods, iterated conditional modes (ICM), Gibbs sampling, loopy belief propagation (Ji, 2019, page. 144-150). To the interest of simplicity, we only discuss ICM and Gibbs sampling.

2.1 Iterated conditional modes

One of the best properties of Bayesian network (BN) is its factorization of the joint probability distribution: we can compute its joint distribution using chain rule. In MRF, we analogy the HC-theorem as chain rule. However, the partition function Z (see equation 2) prevents us from fully factorization.

The ICM algorithm (Wu and Ji, 2019) was proposed to find a fully factorization approximation that allows us to do MAP estimation (Ji, 2019, equation. 4.33),

$$\begin{aligned} p(\mathrm{x}|\mathrm{y}) &\approx \prod_{i=1}^{N} p(x_i|x_{-i}, \mathrm{y}) \\ &= \prod_{i=1}^{N} p(x_i|N_{x_i}, \mathrm{y}) \end{aligned} \tag{11}$$

We can calculate the local MAP inference given equation 12, but the gathering the local MAP results does not mean we have the correct MPE inference of $p(\mathrm{x}|\mathrm{y})$ (Ji, 2020c, page. 67). The ICM method apply the idea coordinate ascent (Ji, 2020b, page. 68) and uses a iterative way to approach MPE inference. Given the observation y and current values of its neighbours $N_{x_i}^t$, the algorithm do the MAP inference until converge:

$$x_i^{t+1} = \arg\max_{x_i} p(x_i|N_{x_i}^t, \mathrm{y}) \tag{12}$$

Given the Potts model, we can get a more concrete equation by introducing the log-linear and pairwise assumption to equation 12,

$$x_i^{t+1} = \arg\min_{x_i} \sum_{x_j^t \in N_{x_i}^t} w_{ij} E_{ij}(x_i, x_j^t) + \alpha_i E_i(x_i, y_i) \tag{13}$$

Algorithm 1 is the pseudo of this method (adopted with modifications from (Ji, 2019, algorithm. 4.1)).

2.2 Gibbs sampling

Gibbs sampling methods samples from the stationary distribution of a Markov chain to approximate the real posterior distribution $p(\mathrm{x}|\mathrm{y})$. In this Markov chain, each configuration of the nodes in MRF is believed to be a state $s^t = \{x_1, x_2, ..., x_N\}^\top$. The transition function of the state is defined as $p(x_i^t|\mathrm{x}_{-i}^{t-1}, \mathrm{y})$ where x_{-i} represents all the node variables apart from

Algorithm 1 Pesudo-code for the ICM algorithm in label observation model

Input the observation nodes y and label nodes $\mathcal{V} = \{X_1, X_2, ..., X_N\}$
Initialize x to x^0
t=0
while not converging **do**
  for Each node x_i in $\mathcal{V}$ **do**
    $x_i^{t+1} = x_i^{t+1} = \arg\min_{x_i} \sum_{x_j^t \in N_{x_i}^t} w_{ij} E_{ij}(x_i, x_j^t) + \alpha_i E_i(x_i, y_i)$
    $\mathrm{x}^{t+1} = \{x_1^t, x_2^t, ..., x_i^{t+1}, ..., x_N^t\}$
    $t = t + 1$
Output x^t

the target x_i, which is equivalent to $p(x_i|N_{x_i}, \mathrm{y})$, due to the Markov property. The close-form solution of this probability $p(x_i|N_{x_i}, \mathrm{y})$ is,

$$p(x_i|N_{x_i}, \mathrm{y}) = \frac{1}{Z} \exp(- \sum_{x_j^t \in N_{x_i}^t} w_{ij} E_{ij}(x_i, x_j^t) - \alpha_i E_i(x_i, y_i)) \tag{14}$$

The Gibbs sampling requires a burin-in period to allow the constructed Markov chain to reach stationary. Usually, we treat the burn-in length as a hyper-parameter.

After burn-in, we start to collect samples. When the size of node is large, finding the mode is challenging. Instead, we case use the sample mean as a approximation of the mode (Ji, 2020b, page. 71).

Algorithm 2 shows the pseudo-code of Gibbs sampling (adopted and modified based on (Ji, 2019, algorithm. 4.2).

Algorithm 2 Pesudo-code for the Gibbs sampling algorithm in label-observation model

Input the observation nodes y and label nodes $\mathcal{V} = \{X_1, X_2, ..., X_N\}$ and the expected number of sample T
Initialize x to x^0
while not burn-in **do**
  for Each node x_i in $\mathcal{V}$ **do**
    $x_i^{t+1} \sim p(x_i|N_{x_i}^t, \mathrm{y})$
    $\mathrm{x}^{t+1} = \{x_1^t, x_2^t, ..., x_i^{t+1}, ..., x_N^t\}$
    $t = t + 1$
$t = 0$
while $t < T$ **do**
  for Each node x_i in $\mathcal{V}$ **do**
    $x_i^{t+1} \sim p(x_i|N_{x_i}^t, \mathrm{y})$
    $\mathrm{x}^{t+1} = \{x_1^t, x_2^t, ..., x_i^{t+1}, ..., x_N^t\}$
    $t = t + 1$
Output $\mathrm{x}^* = \frac{1}{T} \sum_{t=1}^{T} \mathrm{x}^t$ as an approximation of MAP inference

3. Experiment setting

In this experiment, we need to solve an image segmentation problem using MRF. We are instructed to identify the foreground and background of a given image (see 2). The image is 640×360

Figure 2: (Adopted from (Ji, 2020a)) the test image

Meanwhile, we are given a label-observation model using a Pott model pairwise energy $E_ij = 1 - \delta(x_i, x_j)$. Meanwhile, we are given unary potential function as $E_i(x_i, y_i) = -\ln N(\mu_x, \sigma_x^2)$ where $N(\mu, \sigma^2)$ means the pdf of a Gaussian distribution with mean as μ and variance as σ^2.

4. Results

In this section, we discuss the experimental result using ICM method and Gibbs sampling method.

4.1 ICM results

In using the ICM method to infer the MPE of the labels, I varies the initialization method of the label $\mathcal{V} = \{X_1, ..., X_N\}$. The first method I used is called *uniform-bernoulli initialization*, which sample from $\{0, 255\}$ follow a uniform Bernoulli distribution ($Ber(p = .5)$). The second method is called *all-zeros initialization*, which fixed the initiating state of all labels as 0. For both initialization conditions, the ICM algorithms were run until converging (when no more pixels changes).

Figure 3 showed the converging conditions of the ICM method. Th all-zeros initiate algorithm converged at 27th iteration, while the uniform-bernoulli initialization converges at 17th (range from 16-18 depends on initialization) iteration. Although the converging curve is steeper at the early iteration stage, it has a heavier tail at the late iterating stage.

Figure 4 displayed some examples of the labels that allow us to compare the performance of uniform-bernoulli initialization and all-zeros initialization side by side. Both methods had segmentation right after the 1st iteration. We may understand the task for the uniform-bernoulli method to get rid of the noise introduced by the random initialization. Meanwhile, a semantic interpretation of the task for all-zeros initiation is to correct the misinterpreted texture in the foreground (e.g. windows, the logo, etc). Along with the number of iterations, the label image was less noisy with uniform-bernoulli initialization and less misinterpreted

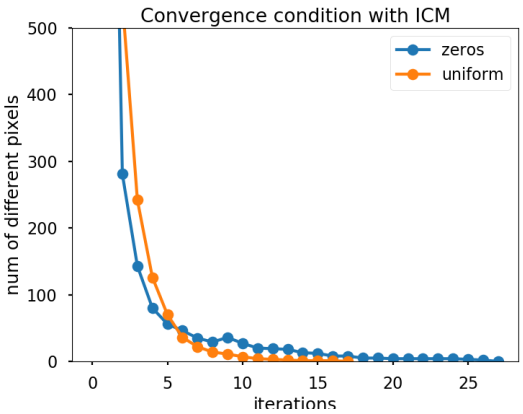


Figure 3: Converging conditions of ICM

foreground texture in the all-zeros initialization condition. For both initialization, human observers may fail to tell the difference between the label images at the iterations after 5 with only a few pixels change.

4.2 Gibbs sampling results

Like the ICM method, we used both uniform-bernoulli initialization and all-zeros initialization in Gibbs sampling. Meanwhile, we examined the impact of the burn-in length. Here we can analogy the burn-in length as the iterations discussed in the ICM method. We use the term "iteration" to describe a sample for each label (pixel) in an image and m iterations means sampling n samples for each label. The number of sweeps is treated as a hyperparameter in this project, which ranges from $\{1, 2, 5, 10, 15, 20\}$. The hyperparameter candidate list is designed to match the iterations in ICM.

While varying the burn-in length, we fixed the number of samples we collected for inference as 10,000. We use the mean of the samples $\bar{\mathrm{x}} = \sum_{t=1}^{T} \mathrm{x}^t$ as an approximation of the model image and passed this $\bar{\mathrm{x}}$ into a nonlinear function that return 255, if $x_i \geq 255/2, x_i \in \bar{\mathrm{x}}$ to ensure the inferred lables is within the value space $\{0, 255\}$.

Figure 5 showed the experiment results. Like what we observed in the ICM results, the main challenge for uniform-beroulli initialization is denoising and that for all-zeros is to correct the system mislabeled foreground textures. The difference is that with few iterations, Gibbs sampling tended to be noisier (see figure 5a, 5c)) and to have more mislabeling (see figure 5b, 5d)), compared with the ICM method. The tradeoff is as the labels inferred by Gibbs sampling were more accurate when we can afford a long burn-in period (iterations $>$ 15). Figure 5g and 5h has less mislabeled texture pixels compared to figure 4g and 4h.

4.3 ICM and Gibbs sampling

Through the discussion in section 4.1 and section 4.2, we knew that Gibbs sampling and random initialization require more iterations to get better asymptotic performances, while the ICM method and all-zeros initialization reach a quite satisfying performance within a few iterations. One explanation is that both the ICM method and all-zeros method are

biased methods and Gibbs sampling and random initialization introduced minimum biases. Algorithms with good biases are more sample efficient, but less likely to reach close to the optimal solution.

There is no deterministic answer to the question: which methods are better, ICM or Gibbs sampling. When the computation resource is a non-negligible factor, we can resort to ICM+all-zeros initialization+early stopping to get a tolerable solution. Otherwise, when we consider more about the accuracy, we can use Gibbs sampling + uniform random initialization to polish inference results.

5. Conclusions

In this project, we were instructed to run a simple foreground-background segmentation task using both ICM and Gibbs sampling method. We concluded that both ICM method and Gibbs sampling can complete the image segmentation task. Among them, Gibbs sampling is more computational expensive but returns a better asymptotic segmentation performance.

References

Qiang Ji. Probabilistic Graphical Models for Computer Vision. Academic Press, 2019.

Qiang Ji. Fall20 project 4 instruction. 2020a.

Qiang Ji. Fall20 undirected pgm. 2020b.

Qiang Ji. Fall20 bn inference. 2020c.

Yue Wu and Qiang Ji. Facial landmark detection: A literature survey. International Journal of Computer Vision, 127(2):115–142, 2019.

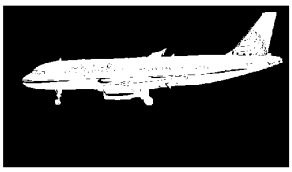
Lei Zhang and Qiang Ji. Image segmentation with a unified graphical model. IEEE Transactions on Pattern Analysis and Machine Intelligence, 32(8):1406–1425, 2009.

6. Appendix

6.1 Simplify the MAP inference with a log-linear potential function

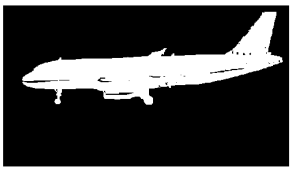
$$
\begin{aligned}
& \arg\max_{\mathrm{x}} p(\mathrm{x}|\mathrm{y}) \\
=& \arg\max_{\mathrm{x}} \frac{1}{Z} \exp[-E(\mathrm{x},\mathrm{y})] \\
\Rightarrow& \arg\max_{\mathrm{x}} \exp[-E(\mathrm{x},\mathrm{y})] \\
\Rightarrow& \arg\max_{\mathrm{x}} -E(\mathrm{x},\mathrm{y}) \\
\Rightarrow& \arg\min_{\mathrm{x}} E(\mathrm{x},\mathrm{y})
\end{aligned}
\tag{15}
$$

(a) At iteration 1, infer with the uniform bernoulli initialization



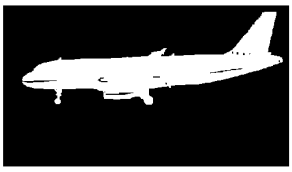
(b) At iteration 1, infer with the all zeros initialization

(c) At iteration 2, infer with the uniform bernoulli initialization

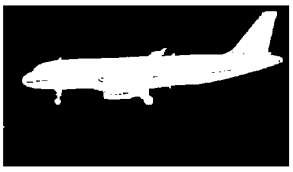


(d) At iteration 2, infer with the all zeros initialization

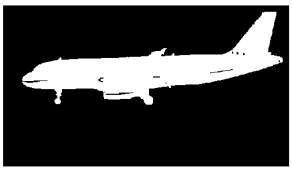
(e) At iteration 5, infer with uniform bernoulli initialization



(f) At iteration 5, infer with the all zeros initialization

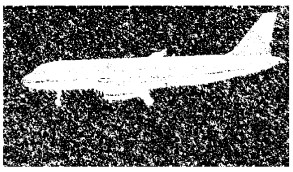


(g) At iteration 17(converged), infer with the uniform bernoulli initialization

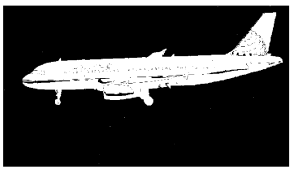


(h) At iteration 28 (converged), infer with all zeros initialization

Figure 4: ICM results at different iterations. The left column was initiated with a uniform bernoulli distribution. The right column was initiated with all labels are fixed to be zero

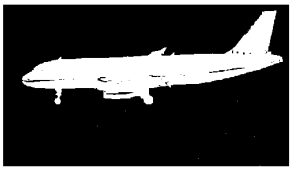


(a) At iteration 1, infer with the uniform bernoulli initialization

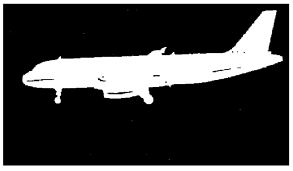


(b) At iteration 1, infer with the all zeros initialization

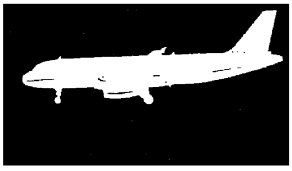
(c) At iteration 2, infer with the uniform bernoulli initialization



(d) At iteration 2, infer with the all zeros initialization

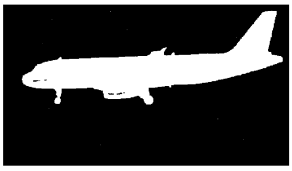


(e) At iteration 5, infer with uniform bernoulli initialization



(f) At iteration 5, infer with the all zeros initialization

(g) At iteration 20, infer with the uniform bernoulli initialization



(h) At iteration 20, infer with all zeros initialization

Figure 5: The Gibbs sampling results at different iterations. The left column was initiated with a uniform-bernoulli distribution. The right column was initiated with all-zeros initialization.