10-708: Probabilistic Graphical Models 10-708, Spring 2016

Lecture 19: Indian Buffet Process

Lecturer: Matthew Gormley *Scribes: Kai-Wen Liang, Han Lu*

1 Dirichlet Process Review

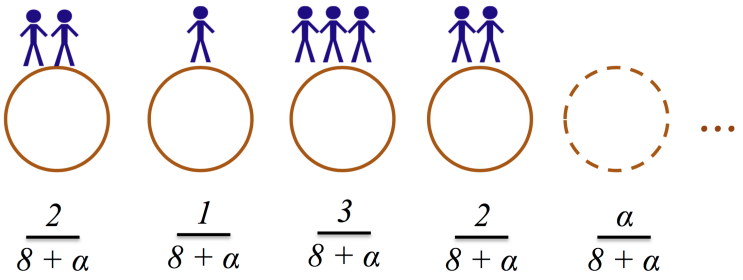
1.1 Chinese Restaurant Process

In probability theory, the Chinese restaurant process is a discrete-time stochastic process, analogous to seating customers at infinite number of tables in a Chinese restaurant. Assume that each customer enters and sits down at a table. The way they sit at the tables follows the process below:

- The first customer sits at the first unoccupied table
- Each subsequent customer chooses a table according to the following probability distribution:

$$p(k\text{th occupied table}) \propto n_k$$
$$p(\text{next unoccupied table}) \propto \alpha$$

In the end, we have the number of people sitting at each table. This corresponds to a distribution over clusterings, where $custermer = index$, and $table = cluster$. Although CRP gives potentially infinite number of clusters, the expected number of clusters given n customers is $O(\alpha \log(n))$. The number of clusters also indicates the rich-get-richer effect on clusters. Also as α goes to 0, the number of clusters goes to 1, while as α goes to $+\infty$, the number of clusters goes to n.



1.2 CRP Mixture Model

Here we denotes $z_1, z_2, \ldots z_n$ as a sequence of indices drawn from a Chinese Restaurant Process, where n is the number of customers. For each table/cluster we also draw a distribution θ_k^* from a base distribution H. Despite there are infinite number of tables/clusters we can have in CRP, the maximum number of clusters/tables is the number of the custumers (i.e. n). Finally, for each customer z_i (cluster indice), draw

a observation x_i from $p(x_i|\theta^*_{z_i})$. Here, in chinese restaurant story, we can view z_i as the table assignment of ith customer, θ^*_k is the table specific distribution over dishes, and finally x_i is the dishes that ith customer ordered follow the table specific dishes distribution.

The next thing we want to know is inference problem (i.e. computing the distribution of $\boldsymbol{z}$ and $\boldsymbol{\theta}$ given observation $\boldsymbol{x}$). Because the exchangeability of CRP, the Gibbs sampler is easy to do inference because for each observation, we can remove the customer/dish from the restaurant and resample as if the were the last to enter. Here we discribe three Gibbs Samplers for CRP Mixture Model.

- Algorithm. 1 (uncollapsed)
  - Markov chain state: per-customer parameters $\theta_1, \theta_2, \ldots, \theta_n$
  - For $i = 1, \ldots, n$: draw $\theta_i \propto p(\theta_i|\boldsymbol{\theta}_{-i}, \boldsymbol{x})$
- Algorithm. 2 (uncollapsed)
  - Markov chain state: per-customer cluster indices $z_1, \ldots, z_n$ and per-cluster parameters $\theta^*_1, \ldots, \theta^*_k$
  - For $i = 1, \ldots, n$: draw $z_i \propto p(z_i|\boldsymbol{z}_{-i}, \boldsymbol{x}, \boldsymbol{\theta}^*)$
  - Set K =numer of clusters in $\boldsymbol{z}$
  - For $k = 1, \ldots, K$:draw $\theta^*_k \propto p(\theta^*_k|x_i : z_i = k)$
- Algorithm. 3 (collapsed)
  - Markov chain state: per-customer cluster indices $z_1, \ldots, z_n$
  - For $i = 1, \ldots, n$:draw $z_i \propto p(z_i|\boldsymbol{z}_{-i}, \boldsymbol{x})$

For algorithm 1, if $\theta_i = \theta_j$, then $i, j \in samecluster$. For algorithm 2, since it is uncollapsed, it is hard to draw a new z_i under the conditional distribution.

1.3 Dirichlet Process

- Parameters of a DP:
  - Base distribution, H, is a probability distribution over Θ
  - Strength parameter, $\alpha \in R$
- We say $G \propto DP(\alpha, H)$, if for any partition $A_1 \cup A_2 \cup \cdots \cup A_K = \Theta$ we have: $(G(A_1), \ldots, G(A_K)) \propto Dirichlet(\alpha H(A_1), \ldots, \alpha H(A_K))$

The above definition is to say that the DP is a distribution over probability measures such taht marginlas on finite partitions are Dirichlet distributed. Given Dirichlet Process definition above, we have properties as follows,

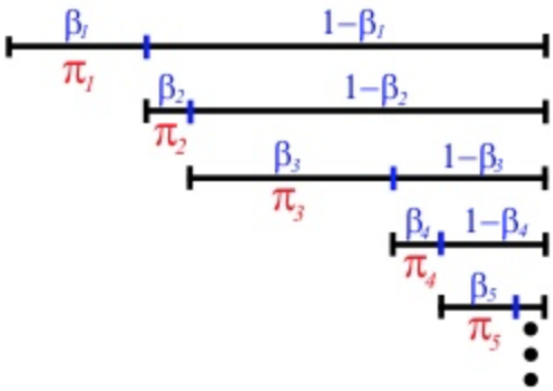
- Base distribution is the mean of the DP: $\mathcal{E}[G(A)] = H(A)$ for any $A_i \subset \Theta$
- Strength parameter is like inverse variance: $V[G(A)] = \frac{H(A)(1-H(A))}{\alpha+1}$
- Samples from a DP are discrete distributions (stick-breaking construction of $G \propto DP(\alpha, H)$ makes this clear)
- Posterior distribution of $G \propto DP(\alpha, H)$ given samples $\theta_1, \ldots, \theta_n$ from G is a DP, $G|\theta_1, \ldots, \theta_n \propto DP(\alpha + n, \frac{\alpha}{\alpha+n}H + \frac{n}{\alpha+n}\frac{\sum_{i=1}^n \delta_{\theta_i}}{n})$

1.4 Stick Breaking Construction

Stick breaking construction provides a constructive definition of the Dirichlet process as follows,

- Start with a stick of length 1, and break it at β_1. Then the length of the broken part of the stick is π_1
- Recursively break the rest of the stick to obtain $\beta_2, \beta_3 \ldots$ and $\pi_2, \ldots, \pi_3$

where $\beta_k \propto Beta(1, \alpha)$, $\pi_k \propto \beta_k \prod_{l=1}^{k-1} 1 - \beta_l$. Also we draw θ_k^* from a base distribution H. Then $G = \sum_{k=1}^{\infty} \pi_k \theta_k^* \propto DP(\alpha, H)$



2 Indian Buffet Process

2.1 Motivation

There are some latent feature models that are familiar to us. For example, they are factor analysis, probabilistic PCA, cooperative vector quantization, and sparse PCA. The applications are various, one of the application is as follows: we have images, and there are some set of objects in it, we want to get a vector, in which a one corresponds to the existence of the object in the image and zero if not. What latent feature models do is to help us assign our data instances to multiple classes, while a mixture model only assign one data instance to one class.

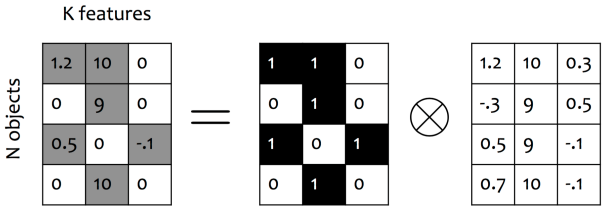
Another example is Netflix challenge, where we have a sparse data of the preference of users, and we want to find movie to recommend to users. They also allows infinite features so that we do not need to specify beforehand.

The formal description of latent feature models is as follows: let x_i be the i^{th} data instance, and $\mathbf{f}_i$ be its features. Define $X = [x_1^T, x_2^T, \ldots, x_N^T]$ be the list of data instances and $F = [\mathbf{f}_1^T, \mathbf{f}_2^T, \ldots, \mathbf{f}_N^T]$ be the list of features. The model is then specified by the joint distribution of $p(X, F)$, and by specifying some priors over the features, we further factorize it as $p(X, F) = P(X|F)p(F)$.

We can further decompose the feature matrix F into a sparse binary matrix Z and a value matrix V. That is, for a real matrix F, we have

$$F = Z \otimes V,$$

where $\otimes$ is elementwise product and $z_{ij} \in \{0, 1\}$ and $v_{ij} \in \mathcal{R}$. One example is shown as follows:



The reason that this is a powerful idea is that as the number of feature K, which is the number of column here, goes to infinity, we do not need to represent the entire matrix V (even if it might be dense), as long as the matrix Z is appropriately sparse. Therefore, the model becomes

$$p(X, F) = p(X|F)p(Z)p(V).$$

The main topic of this lecture, Indian Buffet Process, is going to provide a way to specify $p(Z)$ under the condition of infinite number of features k. Before going to the infinite latent feature models, we first review the basics of finite feature models.

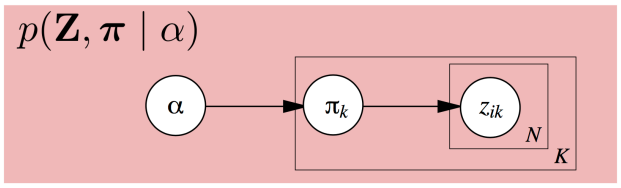
2.2 Finite Feature Model

The first example is Beta-Bernoulli Model. We have encountered this model before when we were talking about LDA. Here we restate the coin-flipping story: From a hyperparameter α, we sample a weighted coin for each column k, and for each row n we sample a head or tail.

Here we make things more formal. For each column we sample a feature π_k and for each row we sample an ON/OFF value based on π_k. That is,

- for each feature $k \in \{1, \ldots, K\}$:
  - $\pi_k \sim Beta(\frac{\alpha}{K}, 1)$ where $\alpha > 0$
  - for each object $i \in \{1, dots, N\}$:
    - $z_{ik} \sim Bernoulli(\pi_k)$

The graphical representation can be drawn as the plate diagram as follows. This gives us the probability of z_{ik} given π_k and α.



Because Beta distribution is the conjugate prior of Bernoulli distribution (this is the special case for Dirichlet distribution being the conjugate prior of Multinomial distribution, where the dimension decrease to 2), we

can analytically marginalize out the feature parameters π_k. The probability of just the matrix Z can be written as the product of the marginal probability of each column, that is,

$$\begin{aligned} P(Z) &= \prod_{k=1}^{K} \int \left(\prod_{i=1}^{N} P(z_{ik}|\pi_k) \right) p(\pi_k) d\pi_k \\ &= \prod_{k=1}^{K} \frac{\frac{\alpha}{K}\Gamma(m_k + \frac{\alpha}{K})\Gamma(N - m_k + 1)}{\Gamma(N + 1 + \frac{\alpha}{K})}, \end{aligned}$$

where $m_k = \sum_{i=1}^{N} z_{ik}$ is the number of features ON in column k, and Γ is the Gamma function.

The question that we are interested in is the expected number of non-zero elements in the matrix Z. To answer this question, we first recall that

$$\begin{aligned} \text{if } X \sim Beta(r, s), \text{ then } E[X] &= \frac{r}{r+s} \\ \text{if } Y \sim Bernoulli(p), \text{ then } E[Y] &= p \end{aligned}$$

Since $z_{ik} \sim Bernoulli(\pi_k)$ and $\pi_k \sim Beta(\frac{\alpha}{K}, 1)$, we have

$$E[z_{ik}] = \frac{\frac{\alpha}{K}}{1 + \frac{\alpha}{K}}$$

and we can calculate the expected number of ON element as

$$E[1^T Z 1] = E\left[\sum_{i=1}^{N} \sum_{k=1}^{K} z_{ik} \right] = \frac{N\alpha}{1 + \frac{\alpha}{K}}$$

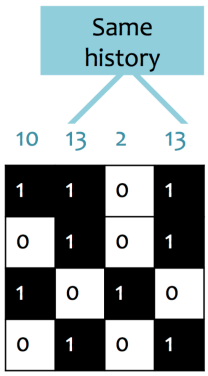
This value is upper-bounded by $N\alpha$. If we take $K \to \infty$, the value simply goes to $N\alpha$, which means that this particular model guarantees sparsity even if we have infinite set of features. However, a bad thing is that when $K \to \infty$, $p(Z)$ will also go to 0 because the first term $\frac{\alpha}{K}$ goes to 0. This is not a property we favor since we do not want to see the entire matrix Z become 0.

To tackle this problem, we first have to recognize the fact that the features are not identifiable, which means that the order of features does not matter to the model. To understand the concept of "not identifiable", recall that in topic model, we usually use MAP inference after a few run for the k^{th} topic from topic model since the order of which topic corresponds to which k does not matter. In a latent feature model, it is obvious that there is no difference between feature $k = 13$ and $k = 27$.

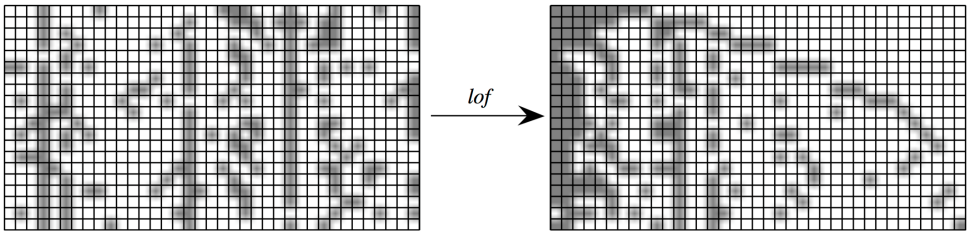
Having this in mind, we can further convert the matrix to Left-Ordered Form (lof). Define the history of feature k to be the magnitude of the binary value given by the column

$$h_k = \sum_{i=1}^{N} 2^{(N-i)} z_{ik}.$$

The figure below help us understand the concept of history:



With history at hand, we further define the $lof(Z)$ to be Z sorted left-to-right by the history of each feature. The figure below depicts the concept.



We define equivalence class $[Z] = \{Z' : lof(Z') = lof(Z)\}$, which is the collection of all the matrices Z's that have the same lof. By doing some counting, we can find out the cardinality of $[Z]$ to be $\frac{K!}{\prod_{h=0}^{2^N-1} K_h!}$, which is the number of matrices that have the same lof.

Now, instead of calculating the probability of a particular matrix $p(Z)$, we calculate the probability of the collection of matrices $p([Z])$. That is

$$\begin{aligned}\lim_{K\to\infty} p([Z]) &= \lim_{K\to\infty} \frac{K!}{\prod_{h=0}^{2^N-1} K_h!} p(Z) \\ &= \frac{\alpha^{K_+}}{\prod_{h=1}^{2^N-1} K_h!} \cdot exp\{-\alpha H_N\} \prod_{k=1}^{K_+} \frac{(N-m_k)!(m_k-1)!}{N!},\end{aligned}$$

where K_+ is the number of features with non-zero history, and $H_N = \sum_{j=1}^{N} \frac{1}{j}$ is the N^{th} harmonic number. By doing the algebra, we can see that the probability no longer goes to infinity. Now we have the enough background to go to the Indian Buffet Process.

2.3 The Indian Buffet Process

Imagine that there is an Indian restaurant with wonderful buffet containing an infinite number of dishes. Each customer walks in, takes as many dishes as possible and then sit down after they have enough food in their plate. The rule for them to select the dish is as follows:

- 1^{st} customer: Starts at the left and selects a $Poisson(\alpha)$ number of dishes.
- i^{th} customer:
  - Samples previously sampled dishes according to their popularity: (i.e. with probability $\frac{m_k}{i}$ where m_k is the number of previous customers who tried dish k)
  - Selects a $Poisson(\frac{\alpha}{i})$ number of new dishes

The example of this process is shown below:

The problem is that the process is not exchangeable, which means that dishes sampled as "new" depend on the customer order. The way to fix that is to modify the way the i^{th} customer selects dishes:

- Makes a single decision for dishes with same history, h: (i.e. if there are K_h dishes with history h sampled by m_h customers, then samples a $Binomial(m_h/i)$ number starting at the left)
- Selects a $Poisson(\frac{\alpha}{i})$ number of new dishes

This is equivalent to seeing them as a lof matrix. Therefore, we fix the problem and can thus calculate the probability $p([Z])$.

Next is to construct a Gibbs sampler for Indian Buffet Process. Specifically, we consider a "prior only" sampler of $p(Z|\alpha)$. For finite number K, we have

$$\begin{aligned} P(z_{ik} = 1|\mathbf{z}_{-i,k}) &= \int_0^1 P(z_{ik}|\pi_k)p(\pi_k|\mathbf{z}_{-i,k})d\pi_k \\ &= \frac{m_{-i,k} + \frac{\alpha}{K}}{N + \frac{\alpha}{K}} \end{aligned}$$

where $z_{-i,k}$ is the k^{th} column except row i, and $m_{-i,k}$ is the number of rows with feature k except i. For infinite K, since Indian Buffet Process is exchangeable, we can do the sampling just like CRP, in which we choose an order such that the i^{th} customer was the last to enter. For any k such that $m_{-i,k} > 0$, resample

$$P(z_{ik} = 1|\mathbf{z}_{-i,k}) = \frac{m_{-i,k}}{N},$$

then draw a $Poisson(\frac{\alpha}{i})$ number of new dishes.

There are some properties of Indian Buffet Process that should be noted:

- It is infinitely exchangeable.
- The number of ones in each row is $Poisson(\alpha)$.
- The expected total number of ones is αN.
- The number of nonzero columns grows as $O(\alpha \log N)$.
- It has a stick-breaking representation.
- It can be interpreted using Beta-Bernoulli process.

Finally, the posterior inference can be done using several different methods such as Gibbs sampling, Conjugate sampler, etc. And previous literatures have reported using this model for graph structures, protein complexes, etc.