Dirichlet Process Mixture Model


Algorithm 3: DPMM with conjugate priors

(Review from July 3)

We have the following Hierarchical Bayes model, and we wish to sample from the posterior of the labels $z_j$ given the observe data $y$.

\[
\begin{align*}
\varphi_k &\sim g(\eta_0) \\
\theta &\sim \text{Dir}(\frac{\alpha}{K}, \cdots, \frac{\alpha}{K}) \\
z_j &\sim \text{Categorical}(\theta) \\
y_j &\sim f(\varphi_{z_j})
\end{align*}
\]

where

\[
g(\varphi_k; \eta_0) = h_g(\varphi_k) \exp\{\eta_0 \cdot t(\varphi_k) - A(\eta_0)\}
\]

\[
p(z) \propto \prod_k \Gamma\left(\frac{\alpha}{K} + n_k\right), \quad n_k \equiv \sum_j 1[z_j = k]
\]

\[
p(y_j|z_j, \varphi) = h_f \exp\{\eta(y_j) \cdot t(\varphi_{z_j})\}
\]

We assume that the prior $g$ is conjugate to the likelihood $f$. Then we can integrate out the $\varphi$-s and $\eta_0$ to sample from the posterior of the labels given the data. Algorithm 3 from
Neal accomplishes this via Gibbs sampling on $z$: $z_j | z_{\backslash j}, y$.

From Bayes Rule,

$$p(z_j | z_{\backslash j}, y) \propto p(z_j | z_{\backslash j}, y_j)p(y_j | z, y_{\backslash j})$$

$$= p(z_j | z_{\backslash j})p(y_j | z, y_{\backslash j})$$

How can we compute each term? First, check exchangeability.

$$p(z) = \prod_{j=1}^{J} p(z_j | z_{1:j-1}) = \prod_{j} \frac{\alpha}{K} + \frac{\sum_{i=1}^{j-1} 1[z_i = z_j]}{\alpha + j - 1} \propto (\prod_j (\alpha + j - 1))^{-1} \prod_k \frac{\Gamma\left(\frac{\alpha}{K} + n_k\right)}{n_k}$$

From exchangeability, without loss of generality look at the last index $J$.

1) First term:

$$p(z_J | z_{1:J-1}) \propto \frac{\alpha}{K} + \sum_{j=1}^{J-1} 1[z_j = z_J]$$

2) Second term: from last lecture,

$$p(y | z) = \left(\prod_j h_f(y_j)\right) \prod_k \exp\left\{-A(\eta_0) + A(\eta_0 + \sum_j 1[z_j = k] \eta(y_j))\right\}$$

Thus,

$$p(y_J | z, y_{1:J-1}) = \frac{p(y | z)}{p(y_1, \cdots, y_{J-1} | z)}$$

$$= \frac{(\prod_j h(y_j)) \prod_k \exp\left\{-A(\eta_0) + A(\eta_0 + \sum_j 1[z_j = k] \eta(y_j))\right\}}{(\prod_{j=1}^{J-1} h(y_j)) \prod_k \exp\left\{-A(\eta_0) + A(\eta_0 + \sum_{j=1}^{J-1} 1[z_j = k] \eta(y_j))\right\}}$$

$$= h(y_J) \exp\left\{A(\eta_0 + \sum_{j=1}^{J-1} 1[z_j = z_J] \eta(y_J)) - A(\eta_0 + \sum_{j=1}^{J-1} 1[z_j = z_J] \eta(y_J))\right\}$$

The Infinite Limit of Finite Mixture Model

What if $K \to \infty$?

We will flip the labels $(k)$ arbitrarily if it’s convenient to think. Or, we will not distinguish $k$ if no $z_j$ are assigned to it. Consider $p(z_j = k |$ other one is already assigned).

Define $n_k^{-j}$ as the same definition of $n_k$ ignoring the impact of $z_j$. 

2
\[ p(z_j|z_{\setminus j}, y) \propto \begin{cases} \left( \frac{\alpha}{K} + n_k^{-j} \right)p(y_j|y_{\setminus j}, z) & \text{if } n_k^{-j} > 0 \\ \frac{\alpha}{K}p(y_j) & \text{otherwise} \end{cases} \]

Note that the second term is from
\[ \frac{\alpha}{K}p(y_j|y_{\setminus j}, z) = \frac{\alpha}{K}h(y_j)\exp\left\{ A(\eta_0 + \eta(y_j)) - A(\eta_0) \right\} = \frac{\alpha}{K}p(y_j) \]

Now,
\[ p(z_j = k, k \in \{ l : n_l^{-j} > 0 \}|z_{\setminus j}, y) \propto \left( \frac{\alpha}{K} + n_k^{-j} \right)p(y_j|y_{\setminus j}, z) \]  \hspace{1cm} (1)

\[ p(z_j \notin \{ l : n_l^{-j} > 0 \}|z_{\setminus j}, y) \propto \frac{\alpha}{K}|\{ l : n_l^{-j} = 0 \}|p(y_j) \]  \hspace{1cm} (2)

As \( K \to \infty \) and \( J \) remains finite, since \(|\{ l : n_l^{-j} > 0 \}| \leq J\),
\[ (1) \to n_k^{-j}p(y_j|y_{\setminus j}, z) \]
\[ (2) \to \alpha p(y_j) \]

\[ \therefore p(z_j|z_{\setminus j}, y) \propto \begin{cases} n_k^{-j}p(y_j|y_{\setminus j}, z) & \text{if } n_k^{-j} > 0 \\ \frac{\alpha}{p}(y_j) & \text{otherwise} \end{cases} \quad \text{as } K \to \infty \]

**Algorithm 8: Sampling from a general DPMM**

The above algorithm assumes that the prior \( g \) is conjugate to the likelihood \( f \). In a more general setting we cannot generally integrate out \( \varphi \). Algorithm 8 from Neal is a general MCMC method to sample from the Dirichlet Process Mixture Model.

Using the Chinese Restaurant metaphor, a state of the algorithm consists of the table assignments, and dishes. For instance: tables 1, 2, 3, with dishes \( \varphi_1, \varphi_2, \varphi_3 \), and customers 1, 3, 4; 2; 5 (ie \( z_1 = 1, z_2 = 2, \text{etc..} \)).

Then the algorithm can be summarized as follows;

Given initial values,
1) Resample \( \varphi \)'s given \( y, z \).
2) Resample \( z_j \)'s from \( p(z_j|\varphi, y, z_{\setminus j}) \).
Step (1) is relatively easy, one can design a Metropolis-Hastings algorithm for this purpose.

Step (2) is somewhat harder, since one must ensure reversibility. If we naively resample $z_j$, and $z_j$ belongs to a unique cluster, then the probability of coming back is 0. The idea is to consider a few extra tables at each step.

For instance, suppose we resample person 1 above. Then there remain 2 people at table 1. Sample two new iid tables with parameters $\varphi_4, \varphi_5$. Assign them weight proportional to $\alpha/2$. Then reassign person 1 with the usual Chinese restaurant probabilities, ie each table with weight proportional to the number of people sitting there, and the last two tables with their weight $\alpha/2$ above.

Note that the whole process is not reversible, but each step is so: for the first step of sampling new tables, we must think that these tables are hidden. Then in the second step of moving from a table to another one, moving back presents the same choices and probabilities. Finally the step of deleting the extra tables is the inverse of the first one.

Note also that we never resample a person sitting at a table alone. Finally, using more than one table at each resampling step is a computational convenience - this way it is more likely to sample a $\varphi$ consistent with the data.

Mixed Membership Mixture Modelling

Reference: Latent Dirichlet Allocation by Blei, Ng, and Jordan

Latent Dirichlet Allocation

We consider a text analysis setting: $w_{jn} \in \{1, 2, \ldots, w\}$ are words in $J$ documents indexed by $j$. Each document has $N_j$ words.

The following is a DP Mixture model for this problem:

$$\pi \sim \text{Dir}(\frac{\alpha}{K}, \ldots, \frac{\alpha}{K})$$

$$z_j \sim \text{Categorical}(\pi)$$

$$w_{jn} \sim \text{Categorical}(\varphi_{z_j})$$

$$\varphi_k \sim \text{Dir}(\eta)$$

where $\pi$ is the frequencies of topics, $z_j$ are the topics, $\varphi_k$ are the topic word frequencies.
Each document is associated with a topic, and the words $w_{jn}$ are drawn with the frequencies $\varphi_{z_j}$ associated to the topic in their document $z_j$.

This model does not capture that a document may have multiple topics. Latent Dirichlet Allocation allows this. The LDA model is:

$$
\begin{align*}
\pi_j &\sim \text{Dir}(\alpha, \cdots, \alpha) \\
z_{jn} &\sim \text{Categorical}(\pi_j) \\
w_{jn} &\sim \text{Categorical}(\varphi_{z_{jn}}) \\
\varphi_k &\sim \text{Dir}(\eta)
\end{align*}
$$

Here each document $j$ gets a mixing weight $\pi_j$ across topics. Then with $z_{jk}$ we draw a topic for each word, from the categories specific to each document. In $w_{jn}$ every word is drawn from the dictionary corresponding to its topic $z_{jn}$. Finally $\varphi_k$ are drawn from the conjugate prior to the categorical distribution.

How to make this non-parametric? We could try letting $K \to \infty$. However the different $\pi_j$
will have no overlap, and that is not what we want. In the current setting nothing is tying them together. Later, we will see that Hierarchical Dirichlet Processes can enforce that the \( \pi_j \) are related.

The intuition for LDA is that it makes use of words shared across documents. The best explanation of the data is that co-occurring words get their own topic. These topics assign high probabilities to those specific words. In real examples LDA identifies reasonable topics, and often works very well.

An application of LDA is source separation from audiospectrogram data. Here we have a matrix where each entry is a measure of the energy in a given frequency at a given time point. In the LDA model time slices correspond to documents, and frequencies correspond to words. Applying LDA to a mixed recording separates the sources. This is a very hard problem and LDA is a good solution.

Another application of LDA-like models is in computer vision. Here the data are images, and the problem is to obtain informative summaries and features for later processing and comparison. It is easy to extract a large number of features. An LDA-like model may be able to select important features, and assign weights to them. Then one can use these features for further processing in numerous algorithms. Alternative ways to process the images exist - for instance one can vectorize them directly.

Finally, we would like to discuss the advantages of Bayesian non-parametric modeling. In general it makes it easy to select hyperparameters (such as the number of clusters). Further, hierarchical models let us deal in a principled way with non-iid data. This is especially true if the group sizes are small. Finally, it is relatively easy to add modeling assumptions, for instance by adding extra hierarchies.