Nested Hierarchical Dirichlet Processes

John Paisley, Chong Wang, David M. Blei, and Michael I. Jordan

Review by David Carlson
Overview

- Dirichlet process (DP)
- Nested Chinese restaurant process topic model (nCRP)
- Hierarchical Dirichlet process topic model (HDP)
- Nested Hierarchical Dirichlet process topic model (nHDP)
- Outline of stochastic variational Bayesian procedure
- Results
Dirichlet Process

In general, we can write a that a distribution $G$ drawn from a Dirichlet process can be written as:

$$G \sim \text{DP}(\alpha G_0)$$ (1)

$$G = \sum_{i=1}^{\infty} p_i \delta_{\theta_i}$$ (2)

where $p_i$ is a probability and each $\theta_i$ is an atom. We can construct a Dirichlet process mixture model over data $W_1, ..., W_N$:

$$W_n|\varphi_n \sim F_W(\varphi_n)$$ (3)

$$\varphi_n|G \sim G$$ (4)
Generating the Dirichlet process

There are two common methods for generating the Dirichlet process. The first is the *Chinese restaurant process*, where we integrate out $G$ to get the a distribution for $\varphi_{n+1}$ given the previous values as:

$$\varphi_{n+1} | \varphi_1, \ldots, \varphi_n \sim \frac{\alpha}{\alpha + n} G_0 + \sum_{i=1}^{n} \frac{1}{\alpha + n} \delta_{\varphi_i}$$  \hspace{1cm} (5)$$

The second commonly used method is a *stick-breaking construction*. In this case, one can construct $G$ as:

$$G = \sum_{i=1}^{\infty} V_i \left[ \prod_{j=1}^{i-1} (1 - V_j) \right] \delta_{\theta_i}, \quad V_i \overset{iid}{\sim} \text{Beta}(1, \alpha), \quad \theta_i \overset{iid}{\sim} G_0$$  \hspace{1cm} (6)$$

Because the stick-breaking construction maintains the independence among $\varphi_1, \ldots, \varphi_N$ is has advantages over the CRP during mean-field variational inference.
The CRP (or DP) is a flat model. Often, it is of interest to organize the topics (or atoms) hierarchically to have subcategories of larger categories in a tree-structure. One way to construct such a hierarchical data structure is through the *nested Chinese restaurant process* (nCRP).
As an analogy, consider an extension of the CRP analogy. Each customer selects a table (parameter) according to the CRP. From that table, the customer chooses a restaurant accessible only from the table, where he/she chooses a table from that restaurant specific CRP.

As shown in the image, each customer (document) that draws from the CRP chooses a single path down the tree.
Modeling the nCRP

Let \( \boldsymbol{i}_l = (i_1, ..., i_l) \) be a path to a node at level \( l \) of the tree. Then we can define the DP at the end of this path as:

\[
G_{\boldsymbol{i}_l} = \sum_{j=1}^{\infty} V_{(\boldsymbol{i}_l, j)} \prod_{m=1}^{j-1} (1 - V_{(\boldsymbol{i}_l, m)}) \delta_{\theta(\boldsymbol{i}_l, j)}
\]  

(7)

If the next node is child \( j \), then the nCRP transitions to the DP \( G_{\boldsymbol{i}_{l+1}} \), where we define \( \boldsymbol{i}_{l+1} = (\boldsymbol{i}_l, j) \)
Nested CRP topic models

We can use the nCRP to define a path down a shared tree, but we want to use this tree to model the data. One application of the tree-structure is a topic model, where we would define each atom $\theta_{i_l,j}$ defines a topic.

$$\theta_{i_l,j} \sim \text{Dir}(\eta)$$  \hspace{1cm} (8)

Each document in the nCRP would choose one path down the tree according to a Markov process, and the path provides a sequence of topics $\phi_d = (\phi_d,1, \phi_d,2, \ldots)$ which we can use to generate the words in the document. The distribution over these topics is provided by a new document-specific stick-breaking process:

$$G^{(d)} = \sum_{j=1}^{\infty} U_{d,j} \prod_{m=1}^{j-1} (1 - U_{d,m}) \delta_{\phi_d,j}, \quad U_{d,j} \overset{iid}{\sim} \text{Beta}(\gamma_1, \gamma_2)$$  \hspace{1cm} (9)
Problems with nCRP

There are several problems with the nCRP, including:

- Each document is only allowed to follow one path down the tree, limiting the number of topics for each document to the number of levels (typically $\leq 4$), which can force topics to blend (have less specificity).
- Topics are often repeated on many different parts of the tree if they appear as random effects in documents.
- The tree is shared, but very few topics are shared between a set of documents because they each follow independent single paths down the tree.

We would like to be able to learn a distribution over the entire shared tree for each document to give a more flexible modeling structure. The solution given to this problem is the nested hierarchical Dirichlet process.
Hierarchical Dirichlet processes

The HDP is a multi-level version of the Dirichlet process. This is described as the hierarchical process:

\[ G_d | G \sim \text{DP}(\beta G), \quad G \sim \text{DP}(\alpha G_0) \] (10)

In this case, we have that each document has its own DP \( G_d \) which is drawn from a shared DP \( G \). In this way, the weights on each topic (atom) are allowed to vary smoothly from document to document, but still share statistical strength.

This can be represented as a stick-breaking process as well:

\[ G_d = \sum_{i=1}^{\infty} V_i^d \prod_{j=1}^{i-1} (1 - V_j^d) \delta_{\phi_i}, \quad V_i^d \overset{iid}{\sim} \text{Beta}(1, \beta), \quad \phi_i \overset{iid}{\sim} G \] (11)
The nHDP formulation allows (i) each word to follow its own path to a topic, and (ii) each topic its own distribution over a shared tree.

To formulate the nHDP, let a tree $\mathcal{T}$ be a draw from the global nCRP with stick-breaking construction. Instead of drawing a path for each document, we use each Dirichlet process in $\mathcal{T}$ as a base for a second level DP drawn independently for each document. In order words, each document $d$ has tree $\mathcal{T}_d$, where for each $G_{i_l} \in \mathcal{T}$ we draw:

$$G_{i_l}^{(d)} \sim \text{DP}(\beta G_{i_l})$$

(12)
We can write the second level DP as:

\[
G_{i_l}^{(d)} = \sum_{j=1}^{\infty} V_{i_l,j}^{(d)} \prod_{m=1}^{j-1} (1 - V_{i_l,m}^{(d)}) \delta_{\phi_{i_l,j}^{(d)}}, \quad V_{i_l,j}^{(d)} \overset{iid}{\sim} \text{Beta}(1, \beta), \quad \phi_{i_l,j}^{(d)} \overset{iid}{\sim} G_{i_l}
\] (13)

However, we would like to maintain the same tree structure in $T_d$ as in $T$. To do this, we can map the probabilities, so that the probability of being on node $\theta_{(i_l,j)}$ in document $d$ is:

\[
G_{i_l}^{(d)}(\{\theta_{(i_l,j)}\}) = \sum_m G_{i_l}^{(d)}(\{\phi_{i_l,m}^{(d)}\}) \mathbb{I}(\phi_{i_l,m}^{(d)} = \theta_{(i_l,j)})
\] (14)
Generating a Document

After generating the tree $T_d$ for document $d$, we draw document-specific beta random variables that act as a stochastic switch. I.E. if a word is at node $i_l$, it determines the probability that the word uses the topic at that node or continues down the tree. So we stop at node $i_l$ with probability:

$$U_{d,i_l}^{idd} \sim \text{Beta}(\gamma_1, \gamma_2)$$

From the stick-breaking construction, the probability that the topic $\varphi_{d,n} = \theta_{i_l}$ for word $W_{d,n}$ is:

$$Pr(\varphi_{d,n} = \theta_{i_l}| T_d, U_d) = \left[ \prod_{i_m \subset i_l} G_{i_m}^{(d)}(\{\theta_{i_{m+1}}\}) \right] \left[ U_{d,i_l} \prod_{m=1}^{l-1} (1 - U_{d,i_m}) \right]$$
Algorithm 1 Generating Documents with the Nested Hierarchical Dirichlet Process

Step 1. Generate a global tree $\mathcal{T}$ by constructing an nCRP as in Section II-B1.

Step 2. Generate document tree $\mathcal{T}_d$ and switching probabilities $U^{(d)}$. For document $d$,
   a) For each DP in $\mathcal{T}$, draw a second-level DP with this base distribution (Equation 8).
   b) For each node in $\mathcal{T}_d$ (equivalently $\mathcal{T}$), draw a beta random variable (Equation 10).

Step 3. Generate the documents. For word $n$ in document $d$,
   a) Sample atom $\varphi_{n,d} = \theta_{i_t}$ with probability given in Equation (11).
   b) Sample $W_{n,d}$ from the discrete distribution with parameter $\varphi_{d,n}$.
Inference

In a large amount of data, it is difficult to use an MCMC algorithm to efficiently learn the parameters in the model. To solve this problem, the authors developed a *stochastic variational Bayesian* inference scheme that updates over a sub-batch of documents denoted by $C_s$

\[
\text{for } s = 1, \ldots, \infty \text{ do }
\]
\[
\text{for } d \in C_s \text{ do }
\]
\[
\text{Update all local parameters for document } d:
\]
\[
(z_{i,j}^{(d)}, c_{d,n}, V_{i,j}^{(d)}, U_{d,i}) \text{ while holding global variables constant }
\]

**Stochastic updates for corpus variables:**

Find a noisy estimate for the Dirichlet parameters $\lambda'_i$ of $q(\theta_i)$, and then update the global parameters $\lambda_{i,w}^{s+1} = \lambda_0 + (1 - \rho_s)\lambda_{i,w}^s + \rho_s\lambda'_i$.

Likewise update the parameters for $q(V_{i,j})$.
Initialization: A good initialization greatly benefits the stochastic VB algorithm. For a small set of documents, the authors iteratively use k-means through a hierarchical k-means clustering to define an initial tree, with $n_1$ clusters at the top level, $n_2$ clusters at the next level, and $n_3$ clusters at the last level.

In the small experiments a truncated tree with widths of (10,7,5) was used in the inference results to give 430 possible nodes whereas in the "big data" experiments the tree was truncated to (20,10,5).

To test the hold-out set the authors completely held out a set of documents, and then learned their local parameters on 75% of the held-out words and tested on the remaining 25%. Predictive log-likelihood values are reported.


**TABLE II**

**Comparison of the nHDP with the nCRP on Three Smaller Problems.**

<table>
<thead>
<tr>
<th>Method</th>
<th>Data set</th>
<th>JACM</th>
<th>Psych. Review</th>
<th>PNAS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variational nHDP</td>
<td>-5.405 ± 0.012</td>
<td>-5.674 ± 0.019</td>
<td>-6.304 ± 0.003</td>
<td></td>
</tr>
<tr>
<td>Variational nCRP</td>
<td>-5.433 ± 0.010</td>
<td>-5.843 ± 0.015</td>
<td>-6.574 ± 0.005</td>
<td></td>
</tr>
<tr>
<td>Gibbs nCRP</td>
<td>-5.392 ± 0.005</td>
<td>-5.783 ± 0.015</td>
<td>-6.496 ± 0.007</td>
<td></td>
</tr>
</tbody>
</table>
Fig. 2. The New York Times: Average per-word log likelihood on a held-out test set as a function of training documents seen.
Fig. 3. The New York Times: Per-document statistics from the test set using the tree at the final step of the algorithm. (a) A histogram of the size of the subtree selected for a document. (b) The average number of nodes by level within the subtree (white), and the average number by level that have at least one expected observation (black). (c) The average number of words allocated to each level of the tree per document.
Fig. 4. Tree-structured topics from The New York Times. The shaded node is the top-level node and lines indicate dependencies within the tree. In general, topics are learning in increasing levels of specificity. For clarity, we have removed grammatical variations of the same word, such as “scientist” and “scientists.”
Fig. 5. Tree size: The smallest number of nodes containing 90%, 99% and 99.9% of all paths as a function of documents seen for (a) The New York Times, and (b) Wikipedia.
Fig. 6. Wikipedia: Average per-word log likelihood on a held-out test set as a function of training documents seen.
Results

Fig. 7. Wikipedia: Per-document statistics from the test set using the tree at the final step of the algorithm. (a) A histogram of the size of the subtree selected for a document. (b) The average number of nodes by level within the subtree (white), and the average number by level that have at least one expected observation (black). (c) The average number of words allocated to each level of the tree per document.
Fig. 8. Examples of subtrees for three articles from Wikipedia. The three sizes of font indicate differentiate the more probable topics from the less probable.
Conclusions

The nHDP provides a way to eliminate some of the constraints of the nCRP and provide a more informative tree that gives a higher predictive log-likelihood.

Using the complete tree is a method explored in the paper ”Tree-Structured Stick Breaking for Hierarchical Data” by Adams et. al, but this method allows documents to share statistical strength for preferences on the tree structure between documents.

The stochastic variational Bayesian algorithm allows for efficient inference on this complicated model that seems to perform well.