Bayesian Hierarchical Cross-Clustering

Dazhuo Li
Dept. of Comp. Eng. and Comp. Sci.
University of Louisville

Patrick Shafto
Dept. of Psychological and Brain Sciences
University of Louisville

Abstract

Most clustering algorithms assume that all dimensions of the data can be described by a single structure. Cross-clustering (or multi-view clustering) allows multiple structures, each applying to a subset of the dimensions. We present a novel approach to cross-clustering, based on approximating the solution to a Cross Dirichlet Process mixture (CDPM) model [Shafto et al., 2006, Mansinghka et al., 2009]. Our bottom-up, deterministic approach results in a hierarchical clustering of dimensions, and at each node, a hierarchical clustering of data points. We also present a randomized approximation, based on a truncated hierarchy, that scales linearly in the number of levels. Results on synthetic and real-world data sets demonstrate that the cross-clustering based algorithms perform as well or better than the clustering based algorithms, our deterministic approaches models perform as well as the MCMC-based CDPM, and the randomized approximation provides a remarkable speed-up relative to the full deterministic approximation with minimal cost in predictive error.

1 INTRODUCTION

Standard approaches to clustering assume that there is a single clustering that describes all of the data. Consider, for example the Dirichlet process mixture (DPM) model, a widely used model for density estimation and for clustering. The model takes as input a set of data points, and their values on a set of dimensions. For each data point, the DPM infers a latent variable indicating an assignment of the data to a mixture component. A fundamental assumption underlying this approach is that all of the dimensions of the data are described from a single view, i.e., the data points were generated by a single underlying DPM.

However, there are many cases in which a single view does not describe all aspects of the data points. In some cases, we might expect some dimensions to be described by one model while others are merely "noise". More generally, any given data set may be generated by multiple different models, each applying to subsets of the observed dimensions. In these contexts, clustering algorithms typically identify a single dominant structure and the dimensions better explained by other models appear to be weakly structured.

Cross-clustering (or multi-view clustering) relaxes the single-DPM assumption, allowing the possibility that a data set may have multiple different views. Consider, for example, a generalization of the DPM, the Cross Dirichlet Process Mixture model (CDPM) [Shafto et al., 2006, Mansinghka et al., 2009]. This model allows that a single data set may be composed of data generated by multiple different DPMs. The model therefore infers, for each dimension, a latent variable indicating an assignment of that dimension to a view, and w.r.t. each view, an assignment of data points to mixture components of DPM. This provides the capability to separate structured features from noisy features and the ability to identify cases where different dimensions of the data are best described by different DPMs. Because the CDPM is a generalization of the DPM, this approach should lead to improved predictive performance on previously unobserved values. However, cross-clustering models admit a very large number of possible latent structures, and their success depends on reliable, efficient inference algorithms.

In this paper, we propose Bayesian Hierarchical Cross-Clustering (BHCC), a deterministic approach to approximate inference for a CDPM. We also propose Randomized BHCC (RBHCC), a much faster alternative approximation to the CDPM. Building off the work by Heller and Ghahramani [2005a], BHCC builds

Appearing in Proceedings of the 14^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2011, Fort Lauderdale, FL, USA. Volume 15 of JMLR: W&CP 15.

a hierarchical clustering of dimensions, where the posterior probability of merging dimensions in different views is estimated based on the marginal likelihood that the data are generated by a DPM.

2 RELATED WORK

In addition to the work by Shafto et al. [2006], Mansinghka et al. [2009], there has been growing interest in the problem of multi-view clustering. Rodriguez et al. [2008] proposed a very similar approach which they call the Nested Dirichlet Process. In terms of other approaches, there are those that allow for two views [Qi and Davidson, 2009, Gondek and Hofmann, 2004, Dang and Bailey, 2010], and those that allow many views. Because we typically do not know how many views there are *a priori*, approaches that allow potentially many views, and infer the correct number for a given data set are more appealing. Cui et al. [2007] use a sequential approach, iteratively clustering in subspaces that are orthogonal to existing solutions. Guan et al. [2010] propose a deterministic, variational approximation to CDPM. Their model differs in that they use a DP prior on categories via the stick-breaking construction. Unlike in their work, our approaches result in hierarchical clusterings of dimensions, which may be desirable in some situations. Additionally, we provide results on real-world prediction problems to provide objective validation for the approach.

3 Cross Dirichlet Process Mixture Model

The problem of learning cross-cutting category structure can be approached by generalizing standard category-learning approaches. Shafto et al. [2006] introduced the CDPM (which they called *CrossCat*), a generalization of standard DPMs [Neal, 1998]. The CDPM was formalized by assuming that dimensions are assigned to mixtures via the Chinese Restaurant Process (CRP) [Aldous, 1985].

Let X be an $I \times J$ data matrix, where the ith row $X_{i,\cdot}$ represents data point i and the jth column $X_{\cdot,j}$ represents dimension j. Let $\boldsymbol{u}$ be a vector of latent variables representing the partitioning of dimensions into views, where $u_j = v$ indicates that dimension j is assigned to view v. Let Z be a matrix of latent variables representing the partitioning of data points w.r.t. all views, where $Z_{i,v} = c$ indicates that, in view v, data point i is assigned to component c. The generative model for a CDPM is then,

$$\boldsymbol{u} \sim \mathrm{CRP}(\alpha), \tag{1}$$

$$Z_{\cdot,v} \sim \mathrm{CRP}(\alpha), \tag{2}$$

$$\Theta_{c,v} \sim H(\delta), \tag{3}$$

$$X_{i,\boldsymbol{u}=v} \sim F(X_{i,\boldsymbol{u}=v}|\Theta_{Z_{i,v},v}), \tag{4}$$

where α is the concentration hyperparameter of the CRPs (using a single parameter for simplicity), H is the prior distribution over component parameters $\Theta_{c,v}$, F is the component distribution (e.g. F is Binomial distribution and H is Beta distribution), and $\boldsymbol{u} = v$ returns a vector of indices: $(j|u_j = v \text{ for } j = 1, \ldots, J)$. Alternatively, by adopting conjugate models, one may substitute Equation 3 and 4 with

$$X_{Z_{\cdot,v}=c,\boldsymbol{u}=v} \sim G(X_{Z_{\cdot,v}=c,\boldsymbol{u}=v}|\delta). \tag{5}$$

As with the DPMs, analytic inference is intractable, but simple Gibbs sampling algorithms are no longer possible. Because mixing over possible views requires potentially creating new DPMs on data points, special-purpose MCMC algorithms are required (see [Mansinghka et al., 2009]). Developing computationally efficient samplers that mix well is time-consuming and challenging, and it is desirable to have alternatives to sampling-based methods.

4 Bayesian Hierarchical Cross Clustering

The BHCC algorithm takes the data matrix X and produces tree T, a hierarchical clustering of the dimensions. Each subtree in Tree T is represented by a 4-tuple $T_c = (c, T_a, T_b, r_c)$ where c is the identification number for the root node of T_c, T_a and T_b are the left and right subtrees of c, and r_c is the posterior probability of merging T_a and T_b to form T_c. Each node in T is associated with a set of dimensions and forms a view w.r.t. which the data are generated from a DPM.

Definition 4.1. Define $\mathcal{L}(T)$ as a function returning identification numbers for all the leaf nodes in tree T, i.e., if $T = (c, T_a, T_b, r_c)$, then

$$\mathcal{L}(T) = \begin{cases} \{c\} \text{ if } T_a = T_b = \emptyset \\ \mathcal{L}(T_a) \cup \mathcal{L}(T_b) \text{ otherwise.} \end{cases}$$

BHCC is described in Algorithm 1. The algorithm is initialized with each dimension forming a view by itself: it starts with J trees: $T_j = (j, \emptyset, \emptyset, 1)$ for $j = 1, \ldots, J$. The algorithm proceeds by repeatedly merging the pair of subtrees that, when joined, have the highest probability, and continues until all dimensions are joined in the same view. To estimate the posterior

probability of merging trees T_a and T_b, BHCC considers two hypotheses $\mathcal{H}_1^c$ and $\mathcal{H}_2^c$. The null hypothesis $\mathcal{H}_1^c$ states that the set of dimensions $\mathcal{L}(T_a) \cup \mathcal{L}(T_b)$ form one view, i.e., data points in $X_{\cdot,\mathcal{L}(T_a)\cup\mathcal{L}(T_b)}$ were generated by the same DPM,

$$p(X_{\cdot,\mathcal{L}(T_a)\cup\mathcal{L}(T_b)}|\mathcal{H}_1^c) = p(X_{\cdot,\mathcal{L}(T_a)\cup\mathcal{L}(T_b)}|\text{DPM}).$$

We follow [Heller and Ghahramani, 2005a] in approximating the marginal likelihood of the data under a DPM using BHC.

The alternative hypothesis states that the dimensions $\mathcal{L}(T_a) \cup \mathcal{L}(T_b)$ form two or more views, i.e., data points in $X_{\cdot,\mathcal{L}(T_a)\cup\mathcal{L}(T_b)}$ were generated by two or more DPMs. The number of possible ways of dividing n dimensions into two or more dimension clusters is $B_n - 1$ where B_n is the Bell number: $B_n = \sum_{k=0}^{n-1} \binom{n-1}{k} B_k$ and $B_0 = 1$. Thus summing over these possibilities is intractable. BHCC restricts itself to dimension partitionings consistent with the subtrees T_a and T_b (see Definition 5.1). The marginal likelihood of this restricted alternative hypothesis, $\mathcal{H}_2^c$, given the data, is a product over the subtrees:

$$p(X_{\cdot,\mathcal{L}(T_a)\cup\mathcal{L}(T_b)}|\mathcal{H}_2^c) = p(X_{\cdot,\mathcal{L}(T_a)}|T_a)p(X_{\cdot,\mathcal{L}(T_b)}|T_b),$$

where each term on the right-hand side of the equation is a probability of a data under a BHCC tree defined recursively as follows.

Let $T_c = (c, T_a, T_b, r_c)$ be the merged tree. The marginal probability of the data in tree T_c is a weighted sum of the probability of the data under both hypothesis:

$$\begin{aligned} &p(X_{\cdot,\mathcal{L}(T_c)}|T_c) = \pi_c p(X_{\cdot,\mathcal{L}(T_c)}|\mathcal{H}_1^c) + \\ &(1-\pi_c)p(X_{\cdot,\mathcal{L}(T_a)}|T_a)p(X_{\cdot,\mathcal{L}(T_b)}|T_b), \end{aligned} \tag{6}$$

where the weight π_c is the prior probability of this merge, i.e., $\pi_c \overset{\text{def.}}{=} p(\mathcal{H}_1^c)$. Heller and Ghahramani [2005a] proposed a bottom-up method for computing the prior under the CRP:

$$\begin{array}{lll} \pi_c = 1 & d_c = \alpha & \text{if } T_c \text{ is a leaf} \\ \pi_c = \frac{\alpha\Gamma(N_c)}{d_c} & d_c = \alpha\Gamma(N_c) + d_a d_b & \text{otherwise,} \end{array} \tag{7}$$

where α is the concentration hyperparameter, $N_c \overset{\text{def.}}{=} |\mathcal{L}(T_c)|$, and $\Gamma(\cdot)$ is the Gamma function.

The posterior probability of the merged hypothesis given the data is computed using Bayes rule,

$$r_c \overset{\text{def.}}{=} p(\mathcal{H}_1^c|X_{\cdot,\mathcal{L}(T_c)}) = \frac{\pi_c p(X_{\cdot,\mathcal{L}(T_c)}|\mathcal{H}_1^c)}{p(X_{\cdot,\mathcal{L}(T_c)}|T_c)}. \tag{8}$$

The quantity r_c is used to decide greedily which two trees to merge at the stage of inferring the BHCC tree; it also allows one to define posterior predictive distributions as discussed in Section 6.

Algorithm 1: Bayesian Hierarchical Cross-Clustering (BHCC) algorithm.

input : An $I \times J$ data matrix X
output : The final merged tree T

initialize: Each dimension j forms a view by itself, i.e. $T_j \leftarrow (j, \emptyset, \emptyset, 1)$ for $j = 1, \ldots, J$, where the 4-tuple has the format: (root node, left subtree, right subtree, posterior of merge). $S \leftarrow \{T_j | j = 1, \ldots, J\}$. The current largest root node id $c \leftarrow J$

while $|S| > 1$ **do**
  $c \leftarrow c + 1$
  Find the pair of T_a, T_b with the highest probability of the merged hypothesis:

$$r_c \leftarrow \frac{\pi_c p(X_{\cdot,\mathcal{L}(T_a)\cup\mathcal{L}(T_b)}|\text{DPM})}{\pi_c p(X_{\cdot,\mathcal{L}(T_a)\cup\mathcal{L}(T_b)}|\text{DPM}) + (1-\pi_c)p_a p_b}$$

  where $p_a \leftarrow p(X_{\cdot,\mathcal{L}(T_a)}|T_a)$, and $p_b \leftarrow p(X_{\cdot,\mathcal{L}(T_b)}|T_b)$
  $T_c \leftarrow (c, T_a, T_b, r_c)$, i.e. join trees T_a and T_b to form T_c with root node c, and the posterior of merge r_c
  $S \leftarrow S \cup \{T_c\} - \{T_a, T_b\}$
end
$T \leftarrow S$

5 Approximate Inference in a Cross Dirichlet Process Mixture Model

In this section, we show that the BHCC algorithm is an approximate inference algorithm for CDPM.

Definition 5.1. Define $\text{Ptns}(T)$ as a function returning the set of tree-consistent partitionings of the set $\mathcal{L}(T)$, i.e., if $T = (c, T_a, T_b, r_c)$, then

$$\text{Ptns}(T) = \begin{cases} (c) \text{ if } T_a = T_b = \emptyset \\ \big(\mathcal{L}(T)\big) \cup \text{Ptns}(T_a) \times \text{Ptns}(T_b) \text{ else.} \end{cases}$$

For example, assume a binary tree T with 3 leaf nodes: $T = (6, T_4, T_3, r_6), T_4 = (4, T_1, T_2, r_4), T_1 = (1, \emptyset, \emptyset, 1), T_2 = (2, \emptyset, \emptyset, 1), T_3 = (3, \emptyset, \emptyset, 1)$, then $\text{Ptns}(T) = \{(1,2,3), (1,2)(3), (1)(2)(3)\}$.

Lemma 5.2. *Let $\boldsymbol{u}$ be a vector of indicator variables representing a partitioning of N elements, $p(\boldsymbol{u})$ be the probability of $\boldsymbol{u}$ in a Dirichlet-Multinomial model, i.e., $p(\boldsymbol{u}) = \int p(\boldsymbol{u}|\boldsymbol{\theta})p(\boldsymbol{\theta}|\alpha)\, d\boldsymbol{\theta}$ where $p(\boldsymbol{u}|\boldsymbol{\theta})$ is the Multinomial distribution and $p(\boldsymbol{\theta}|\alpha)$ is the Dirichlet distribution, then*

$$p(\boldsymbol{u}) = \frac{\Gamma(\alpha)}{\Gamma(N+\alpha)}\, \alpha^{\max(\boldsymbol{u})} \prod_{v=1}^{\max(\boldsymbol{u})} \Gamma(|\boldsymbol{u} = v|),$$

where $\max(\boldsymbol{u})$ *is the number of clusters in partitioning* $\boldsymbol{u}$.

Lemma 5.3. *The marginal likelihood of a CDPM is:*

$$p(X_{\cdot,\mathcal{L}(T_c)}|\text{CDPM}) = \sum_{\boldsymbol{u}\in\mathcal{U}} p(\boldsymbol{u}) \prod_{v=1}^{\max(\boldsymbol{u})} p(X_{\cdot,\boldsymbol{u}=v}|\text{DPM}),$$

where $\mathcal{U}$ *is the set of all possible partitionings of the dimensions* $\mathcal{L}(T_c)$.

Lemma 5.2 follows from a standard Dirichlet integral. Lemma 5.3 follows from the definition of CDPM. Here the explicit dependence on N, α and δ has been dropped for simplicity.

Definition 5.4. Following from Equation 6, we define $p(\boldsymbol{u}|T_c)$ as

$$p(\boldsymbol{u}|T_c) = \frac{1}{d_c}\,\alpha^{\max(\boldsymbol{u})} \prod_{v=1}^{\max(\boldsymbol{u})} \Gamma(|\boldsymbol{u}=v|).$$

Theorem 5.5. *The quantity in Equation 6 computed by the BHCC algorithm is:*

$$\begin{aligned} &p(X_{\cdot,\mathcal{L}(T_c)}|T_c) \\ &= \sum_{\boldsymbol{u}\in\text{Ptns}(T_c)} p(\boldsymbol{u}|T_c) \prod_{v=1}^{\max(\boldsymbol{u})} p(X_{\cdot,\boldsymbol{u}=v}|\text{BHC}) \end{aligned}$$

Theorem 5.5 can be proven by induction, starting from the base case where T_c is a leaf node; proceeding to an arbitrary non-leaf node T_c, with inductive hypothesis that the Theorem holds for both subtrees T_a and T_b. The essential techniques for the proof are the same as in Heller and Ghahramani [2005a] and we omit the details here.

Corollary 5.6. *For any BHCC tree* $T_c = (c, T_a, T_b, r_c)$*, the following is a lower bound on the marginal likelihood of a CDPM:*

$$\frac{d_c\Gamma(\alpha)}{\Gamma(N_c+\alpha)} p(X_{\cdot,\mathcal{L}(T_c)}|T_c) \le p(X_{\cdot,\mathcal{L}(T_c)}|\text{CDPM})$$

recalling that $N_c = |\mathcal{L}(T_c)|$.

Proof. Notice $\text{Ptns}(T_c) \subseteq \mathcal{U}$, and $p(X|\text{DPM}) \ge p(X|\text{BHC})$, based on Lemma 5.2-5.3, Definition 5.4, and Theorem 5.5 we have

$$\begin{aligned} &p(X_{\cdot,\mathcal{L}(T_c)}|\text{CDPM}) \\ &\ge \sum_{\boldsymbol{u}\in\text{Ptns}(T_c)} p(\boldsymbol{u}) \prod_{v=1}^{\max(\boldsymbol{u})} p(X_{\cdot,\boldsymbol{u}=v}|\text{DPM}) \\ &\ge \sum_{\boldsymbol{u}\in\text{Ptns}(T_c)} p(\boldsymbol{u}) \prod_{v=1}^{\max(\boldsymbol{u})} p(X_{\cdot,\boldsymbol{u}=v}|\text{BHC}) \\ &= \frac{d_c\Gamma(\alpha)}{\Gamma(N_c+\alpha)} \sum_{\boldsymbol{u}\in\text{Ptns}(T_c)} p(\boldsymbol{u}|T_c) \prod_{v=1}^{\max(\boldsymbol{u})} p(X_{\cdot,\boldsymbol{u}=v}|\text{BHC}) \\ &= \frac{d_c\Gamma(\alpha)}{\Gamma(N_c+\alpha)}\, p(X_{\cdot,\mathcal{L}(T_c)}|T_c) \end{aligned}$$

□

6 PREDICTION

In this section we define approximate inference for the posterior predictive distribution of a new data point $p(\boldsymbol{x}|X,T)$, a new dimension $p(\boldsymbol{y}|X,T)$, and a missing value $p(X_{i,j}|X,T)$, for any BHCC tree T. Throughout this section, $\boldsymbol{x}$, a length-J row vector, represents a new data point; and $\boldsymbol{y}$, a length-I column vector, represents a new dimension.

To estimate the predictive distribution of new observation ($\boldsymbol{x}, \boldsymbol{y}$ or $X_{i,j}$) given X, the model sums the predictions for each of the possible CDPM hypotheses, weighted by the posterior probability of these hypothesis given the data. Our approach is to approximate these predictions by considering only tree-consistent hypotheses. We exploit the previously-built BHCC tree, and approximate the sum using only tree-consistent hypotheses. We present predictive distributions in Section 6.1 with recursive definitions, and we offer inductive proofs in Appendix A.

6.1 Predictive Distributions

For any tree $T_c = (c, T_a, T_b, r_c)$, all three types of predictive distribution are approximated recursively by summing over the probability of the new observation ($\boldsymbol{x}$, $\boldsymbol{y}$, or $X_{i,j}$) conditioned on the two hypothesis ($\mathcal{H}_1^c$ and $\mathcal{H}_2^c$) weighted by the posterior probability of the hypothesis given the data:

$$\begin{aligned} &p(\text{new observation}|X_{\cdot,\mathcal{L}(T_c)}, T_c) \\ &\stackrel{\text{def.}}{=} p(\mathcal{H}_1^c|X_{\cdot,\mathcal{L}(T_c)})p(\text{new observation}|\mathcal{H}_1^c) + \\ &(1 - p(\mathcal{H}_1^c|X_{\cdot,\mathcal{L}(T_c)}))p(\text{new observation}|\mathcal{H}_2^c). \end{aligned} \tag{9}$$

The base case of the recursion, for which T_c is a leaf node, is also defined by Equation 9 with $p(\mathcal{H}_1^c|X_{\cdot,\mathcal{L}(T_c)}) = r_c = 1$ by definition.

To predict a novel data point, note that while recursing down the tree, the new data point $\boldsymbol{x}$ is divided into two *independent* pieces according to the tree partitioning: $\boldsymbol{x}_{\mathcal{L}(T_a)}$ and $\boldsymbol{x}_{\mathcal{L}(T_b)}$. Thus for $\boldsymbol{x}$, Equation 9 becomes

$$p(\boldsymbol{x}|X_{\cdot,\mathcal{L}(T_c)}, T_c) = r_c p(\boldsymbol{x}|X_{\cdot,\mathcal{L}(T_c)}, \text{DPM}) + (1-r_c) \\ p(\boldsymbol{x}_{\mathcal{L}(T_a)}|X_{\cdot,\mathcal{L}(T_a)}, T_a) p(\boldsymbol{x}_{\mathcal{L}(T_b)}|X_{\cdot,\mathcal{L}(T_b)}, T_b). \tag{10}$$

When predicting a novel dimension, the new dimension $\boldsymbol{y}$ can be generated by each of the subtrees T_a and T_b. Thus for $\boldsymbol{y}$, Equation 9 becomes

$$p(\boldsymbol{y}|X_{\cdot,\mathcal{L}(T_c)}, T_c) = r_c p(\boldsymbol{y}|X_{\cdot,\mathcal{L}(T_c)}, \text{DPM}) + \\ (1-r_c)\Big(p(\boldsymbol{y}|X_{\cdot,\mathcal{L}(T_a)}, T_a) + p(\boldsymbol{y}|X_{\cdot,\mathcal{L}(T_b)}, T_b)\Big). \tag{11}$$

We define $p(\boldsymbol{y}|X_{\cdot,\mathcal{L}(T_k)}, \text{DPM})$ as the probability of the dimension $\boldsymbol{y}$ given that $X_{\cdot,\mathcal{L}(T_k)}$ is generated from a DPM w.r.t. the view containing the set of dimensions $\mathcal{L}(T_k)$:

$$\begin{aligned} & p(\boldsymbol{y}|X_{\cdot,\mathcal{L}(T_k)}, \text{DPM}) \\ & \stackrel{\text{def.}}{=} \sum_{\boldsymbol{z}\in\mathcal{V}} p(\boldsymbol{z}|X_{\cdot,\mathcal{L}(T_k)}, \text{DPM}) p(\boldsymbol{y}|\boldsymbol{z}) \\ & = \sum_{\boldsymbol{z}\in\mathcal{V}} p(\boldsymbol{z}|X_{\cdot,\mathcal{L}(T_k)}, \text{DPM}) \prod_{c=1}^{\max(\boldsymbol{z})} p(\boldsymbol{y}_{\boldsymbol{z}=c}), \end{aligned} \tag{12}$$

where $\mathcal{V}$ denotes the set of all partitionings of data points in the data, each partitioning $\boldsymbol{z} \in \mathcal{V}$ is represented in the form of a vector of indicator variables. $|\mathcal{V}|$ follows the Bell number and the quantity in Equation 12 is intractable. Again, it is approximated by recursing through the BHC tree.

For a missing value, $X_{i,j}$, those views (a.k.a. dimension clusters) in the tree not including dimension j will not contribute to the prediction. Thus Equation 9 becomes

$$p(X_{i,j}|X_{\cdot,\mathcal{L}(T_c)}, T_c) \stackrel{\text{def.}}{=} \\ r_c p(X_{i,j}|X_{\cdot,\mathcal{L}(T_c)}, \text{DPM})|j \cap \mathcal{L}(T_c)| + (1-r_c) \\ \Big(p(X_{i,j}|X_{\cdot,\mathcal{L}(T_a)}, T_a) + p(X_{i,j}|X_{\cdot,\mathcal{L}(T_b)}, T_b)\Big). \tag{13}$$

7 Randomized BHCC

Computational complexity is the primary limitation of the BHCC algorithm, which takes $O(I^2J^2)$ computation time to build the tree given a $I \times J$ data. This section presents one method that can dramatically decrease the complexity, based on a randomized filtering approach [see Heller and Ghahramani, 2005b].

The Randomized BHCC (RBHCC) algorithm is described in Algorithm 2. It takes in a data set X and randomly selects a subset of dimensions V_0 from the whole set of dimensions V. The original BHCC algorithm is run on $X_{\cdot,V_0}$, obtaining a tree T. Based on the priors of the two top subtrees of T, along with the predictive probabilities that a dimensions belongs to the left subtree and the right subtree (defined in Equation 11), the remaining dimensions, $V - V_0$ are then filtered individually down the tree.

Algorithm 2: Randomized Bayesian Hierarchical Cross-Clustering (RBHCC) algorithm

input : An $I \times J$ data matrix X. A threshold number B determining when to stop RBHCC
output : The final merged tree T

if $J < B$ **then return** $T \leftarrow \textbf{BHCC}(X)$
$V \leftarrow \{1, 2, \ldots, J\}, V_a = V_b = \emptyset$
Pick $V_0 \subset V$ randomly where $|V_0| \ll J$
$T \leftarrow \textbf{BHCC}(X_{\cdot,V_0})$, which, as defined in Algorithm 1, returns a 4-tuple (c, T_a, T_b, r_c) with c the root node id, T_a and T_b the left and right subtrees of the root, and r_c the posterior of merging T_a and T_b
foreach $j \in V - V_0$ **do**
    $\boldsymbol{y} \leftarrow X_{\cdot,j}$
    $p_a \leftarrow p(\boldsymbol{y}|X_{\cdot,\mathcal{L}(T_a)}, T_a)$, $p_b \leftarrow p(\boldsymbol{y}|X_{\cdot,\mathcal{L}(T_b)}, T_b)$
    if $\pi_a p_a > \pi_b p_b$ **then**
        $V_a \leftarrow V_a \cup \{j\}$
    else
        $V_b \leftarrow V_b \cup \{j\}$
    end
end
$T_a \leftarrow \textbf{RBHCC}(X_{\cdot,V_a \cup \mathcal{L}(T_a)}, B)$
$T_b \leftarrow \textbf{RBHCC}(X_{\cdot,V_b \cup \mathcal{L}(T_b)}, B)$
$p_a \leftarrow p(X_{\cdot,\mathcal{L}(T_a)}|T_a)$, $p_b \leftarrow p(X_{\cdot,\mathcal{L}(T_b)}|T_b)$,

$$r_c \leftarrow \frac{\pi_c p(X_{\cdot,\{\mathcal{L}(T_a),\mathcal{L}(T_b)\}}|\text{DPM})}{\pi_c p(X_{\cdot,\{\mathcal{L}(T_a),\mathcal{L}(T_b)\}}|\text{DPM}) + (1-\pi_c) p_a p_b}$$

$T \leftarrow (c, T_a, T_b, r_c)$, where c is a node id unique to T

For RBHCC, the cost is composed of three parts. The upper levels of the tree are constructed using randomized BHCC, which includes recursively running normal BHCC to construct the initial tree and then filtering the remaining dimensions based on the tree. The lower levels (the threshold number B in Algorithm 2 is reached) are built using normal BHCC. The total number of dimension comparisons can be expressed recursively as:

$$Comp(J) = |V_0|^2 + J + Comp(aJ) + Comp((1-a)J),$$

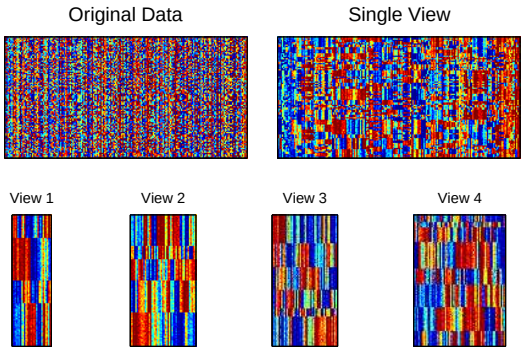

Figure 1: Comparison of the clustering(s) between BHC and BHCC. The data matrix and the results from BHC (Single View) and BHCC (View 1-4) are shown in heatmaps, where each row denotes a data point and each column denotes a dimension. Note that the number of views and the partitioning of dimensions are unknown *a priori* to BHCC and are inferred from the data.

recalling that V_0 is the set of dimensions randomly chosen for running BHCC, $0 \leq a \leq 1$ is the proportion of dimensions on one side of the tree. Note that $|V_0|$ can be considerably smaller than J when J is large; Meanwhile, the depth of a balanced binary tree is $\log(J)$. Thus the number of dimension comparisons $comp(J)$ has the computational complexity $O(J\log(J))$. If the DPM w.r.t. the views is approximated by the randomized BHC algorithm, the overall complexity of RBHCC is $O(IJ\log I\log J)$.

Generally, capturing higher level structures is sufficiently informative. Thus we could restrict the algorithm to running only the top L levels, either a priori or interactively. With these much smaller dimensions and data points cut-off levels (L for dimensions and K for data points), the truncated RBHCC algorithm is linear, $O(IJLK)$.

8 RESULTS

8.1 An Example

We illustrate the performance of the algorithms using a synthetic dataset. We generated a Binomial dataset with 100 data points and 200 dimensions. The data set has four views of data point clusterings. i.e. a CDPM with four DPMs each w.r.t. a subset of the original 200 dimensions. The number of dimensions within Views 1-4 are 30, 50, 50 and 70 respectively.

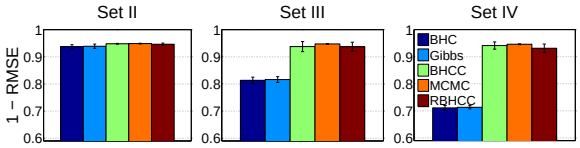

Figure 2: Comparison of predictive performance among DPM approximations (BHC and Gibbs sampling) and CDPM approximations (BHCC, MCMC and RBHCC) on the same synthetic datasets. Set II has 2 views; Set III has 3 views and Set IV has 4 views. Each DPM w.r.t. its view contains four well separated Binomial mixture components.

The number of Binomial mixture components [1] under Views 1-4 are 4, 5, 6 and 8 respectively.

We applied BHC and BHCC on the data. Figure 1 shows the original data, the result from BHC (Single View) and the results from BHCC (Views 1 - 4). Heatmaps are used to display the data matrix and the results where each row denotes a data point and each column denotes a dimension. For BHC and BHCC, the data points are rearranged according to the inferred hierarchical tree over data points to reflect the clustering.

Note that in BHCC, the partitioning of dimensional space is represented by an inferred hierarchical tree over dimensions. It yields four dimension subsets when choosing 0.5 as the threshold of posterior probability of merging. In each view, clear category structure is evident by the horizontal striations.

8.2 Comparison of Predictive Performance

We compared the predictive performance of BHCC and RBHCC to those from Markov Chain Monte Carlo (MCMC) for CDPM, and two inference methods for DPM, i.e. the BHC approximation and collapsed Gibbs sampling [Neal, 1998]. We first compared these algorithms on 3 synthetic datasets. We then compared them on 4 real-world datasets. In each experiment, a one time 10-fold cross-validation is performed on predicting missing values, i.e., the set of entries in the data matrix is randomly partitioned into 10 subsets, of which each subset is hold out once as the validation data and the remaining 9 subsets are used as training data.

The number of filtering levels for RBHCC is set to

[1] Let $\boldsymbol{x}_k$ be a data point in the kth component, and $\boldsymbol{\theta}_k$ be the mean of the kth component. Then $\boldsymbol{x}_k \sim \text{Binomial}(\boldsymbol{N}, \boldsymbol{\theta}_k)$ and $\boldsymbol{\theta}_k \sim \text{Beta}(\boldsymbol{\alpha}, \boldsymbol{\beta})$. We set $\boldsymbol{N} = \mathbf{50}, \boldsymbol{\alpha} = \boldsymbol{\beta} = \mathbf{0.5}$.

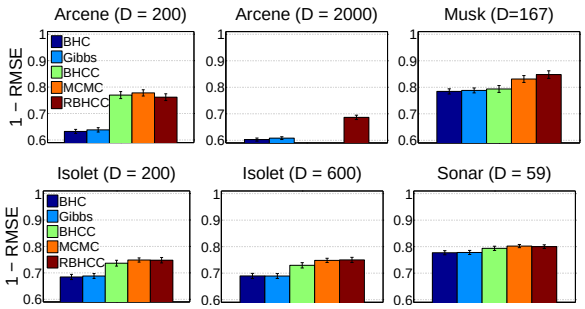

Figure 3: Comparison of predictive performance among DPM approximations (BHC and Gibbs sampling) and CDPM approximations (BHCC, MCMC and RBHCC) on the same real datasets. The 4 real datasets are Arcene, Isolet, Musk and Sonar. The number of dimensions used for the test is D (e.g. D = 200) and the number of data points chosen is fixed to 100. For test Arcene (D=2000), the results from BHCC and MCMC are not available because of long running time.

$L = 6$. In each experiment, the collapsed Gibbs sampling for DPM was performed for 100 iterations [2], the first half was used for burn-in and every fifth sample from the second half was used to estimate posterior quantities. Each iteration consisted of sampling all the latent indicator variables associating mixture components with data points. To facilitate mixing, we interleaved split-merge proposals [Jain and Neal, 2004] between Gibbs sweeps. In each experiment, MCMC for CDPM was performed for 100 iterations, first half of which were used for burn-in and every fifth of the second half for estimating posterior quantities. To improve the mixing, Metropolis coupled MCMC [Geyer, 1991] was performed in which two chains were run, one of which was "heated" by raising the posterior probability to a power 0.8. Due to space limitations, we do not review the MCMC for CDPM and the collapsed Gibbs sampling for DPM here.

We generated 3 sets of 200-dimensional Binomial datasets, each with 100 data points. Set II has 2 views; Set III has 3 views and Set IV has 4 views. Each view contains a DPM with four well separated Binomial mixture components. Figure 2 shows the prediction results. We found that the predictive accuracy of inference methods for DPM decreased dramatically as the number of views in the data increased, while methods for CDPM maintained a consistently high accuracy regardless of the number of views. Meanwhile, RBHCC, BHCC and MCMC yield equally high accuracy.

[2] Tests showed that increasing the number of iterations did not lead to better predictive accuracy.

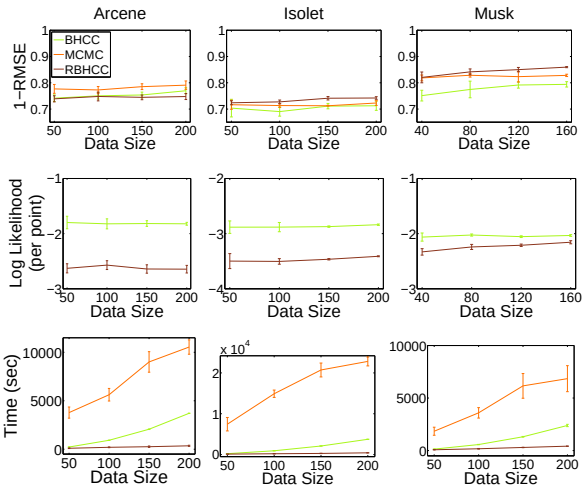

Figure 4: Comparison of the prediction accuracy, the estimated marginal likelihood lower bound (not applicable to MCMC), and the runtime among RBHCC, BHCC and MCMC. X-axis represents the number of dimensions used. The number of data points is fixed to 100.

The real datasets used were the Arcene data (900 data points, 10000 dimensions) from NIPS 2003 feature selection challenge, Isolet (7797 data points, 617 dimensions), Musk Version 1 (476 data points, 168 dimensions), and Sonar mines vs. rocks (208 data points, 60 dimensions) data. All data are from the UCI repository [Asuncion and Newman, 2007]. In each experiment, we chose a subset of 100 data points and varied the number of dimensions between 60 and 2000. Figure 3 shows the results. The performance of each algorithm varied across datasets. Comparing the predictive accuracy between inference algorithms for CDPM and for DPM, we found the former gave a substantial improvement on the Arcene and the Isolet data and a slight improvement on the Musk and the Sonar data. Meanwhile, RBHCC, BHCC and MCMC yield generally equal accuracy.

8.3 Prediction, Estimated Marginal Likelihood, and Runtime Comparisons among CDPM Inference Algorithms

To investigate the relationship between speed and accuracy among the CDPM algorithms, we contrasted prediction accuracy, the estimated log marginal likelihood (per point, not applicable to MCMC) and the runtime among RBHCC, BHCC and MCMC on the real datasets of varying sizes. To do so, we selected a subset of the total dimensions, varying the number of dimensions between 40 and 200 (see Fig-

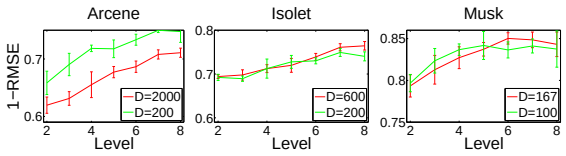

Figure 5: Comparison of prediction accuracy among RBHCC with different filtering levels. The number of dimensions used for the test is D (e.g. $D = 2000$) and the number of data points are fixed to 100.

ure 4). Although the estimated marginal likelihood lower bounds differ for RBHCC and BHCC, the algorithms yield quite similar prediction accuracy. The runtime of MCMC depends on the structure (i.e. number of views and DPMs) underlying the data, which leads to quite variable runtime. Furthermore, we note that, consistent with our complexity analyses, RBHCC offers a significant speedup compared to BHCC: RBHCC scales linearly w.r.t. the number of dimensions in the data, while BHCC is quadratic.

8.4 Varying the Filtering Levels for RBHCC

To investigate the tradeoff between runtime and prediction accuracy in the RBHCC, we varied the number of filtering levels between 2 and 8 for RBHCC and ran it on dataset Arcene, Isolet and Musk datasets. Figure 5 shows the prediction accuracy. For all tests, the accuracy increases as the level increases. Further, the gain in accuracy gets smaller as L reaches to a certain level, indicating the performance reaches a potential upper bound and it is not necessary to continue increasing L.

9 CONCLUSIONS

We described Bayesian Hierarchical Cross-Clustering (BHCC), a novel approach for approximate inference of multi-view data. The algorithm provides a deterministic, agglomerative, approximate approach to inference in a Cross Dirichlet Process Mixture (CDPM) model. We have also introduced a fast, randomized algorithm (RBHCC) that scales linearly in the number of levels of the hierarchy. We have contrasted predictive performance on synthetic and real-world data, with clustering models that adopt a single view of the data, the DPM with Gibbs sampling-based inference and Bayesian Hierarchical Clustering, and cross-clustering models that adopt multiple views of the data, the CDPM with MCMC-based inference, BHCC, and RBHCC. Our results show that algorithms based on inferring multiple views have greater predictive accuracy, that the deterministic approaches perform comparably to MCMC-based inference, and the RBHCC provides a remarkable speed-up relative to BHCC, with little cost in predictive accuracy. We consider these results promising, and future work will explore applications of the model to other real-world data sets, and extensions to more richly structured models.

Acknowledgements

Dazhuo Li is supported by the Department of Energy (DE-EM0000197). The contents of this manuscript are solely the responsibility of the authors and do not necessarily represent the official views of the DOE.

References

D. Aldous. Exchangeability and related topics. 1985.

A. Asuncion and D. J. Newman. UCI machine learning repository, 2007.

Y. Cui, X. Z. Fern, and J. G. Dy. Non-redundant multi-view clustering via orthogonalization. In *Proceedings of the 7th IEEE International Conference on Data Mining*, pages 133–142, 2007.

X. H. Dang and J. Bailey. Generation of alternative clusterings using the cami approach. In *Proceedings of the SIAM International Conference on Data Mining*, pages 118–129, 2010.

C. Geyer. Markov chain monte carlo maximum likelihood. In *Computing Science and Statistics: Proceedings of the 23rd Symposiumon the Interface*, pages 156–163, 1991.

D. Gondek and T. Hofmann. Non-redundant data clustering. In *Proceedings of the 4th IEEE International Conference on Data Mining*, pages 75–82, 2004.

Y. Guan, J. G. Dy, D. Niu, and Z. Ghahramani. Variational inference for nonparametric multiple clustering. In *MultiClust: 1st International Workshop on Discovering, Summarizing and Using Multiple Clusterings held in conjunction with the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2010.

K. A. Heller and Z. Ghahramani. Bayesian hierarchical clustering. In *Proceedings of the 22nd international conference on Machine learning*, pages 297–304, 2005a.

K. A. Heller and Z. Ghahramani. Randomized algorithms for fast bayesian hierarchical clustering. In *PASCAL Workshop on Statistics and Optimization of Clustering*, 2005b.

S. Jain and R. M. Neal. A split-merge markov chain monte carlo procedure for the dirichlet process mix-

ture model. *Journal of Computational and Graphical Statistics*, 13:158–182, 2004.

V. K. Mansinghka, E. Jonas, C. Petschulat, B. Cronin, P. Shafto, and J. B. Tenenbaum. Cross-categorization: A method for discovering multiple overlapping clusterings. In *Advances in Neural Information Processing Systems, Workshop on Nonparametric Bayesian Statistics*, 2009.

R. M. Neal. Markov chain sampling methods for dirichlet process mixture models. Technical Report no.9815, Department of Statistics University of Toronto, 1998.

Z. Qi and I. Davidson. A principled and flexible framework for finding alternative clusterings. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 717–726, 2009.

A. Rodriguez, D. B. Dunson, and A. E. Gelfand. The nested dirichlet process. *Journal of the American Statistical Association*, 103(483):1131–1154, 2008.

P. Shafto, C. Kemp, V. K. Mansinghka, M. Gordon, and J. B. Tenenbaum. Learning cross-cutting systems of categories. In *Proceedings of the Twenty-Eighth Annual Conference of the Cognitive Science Society*, 2006.

A SUPPLEMENTAL MATERIALS

We begin with some preliminary definitions, then proceed to prove the consistency of the predictive distributions.

Definition A.1. Define $\text{Nodes}(T)$ as a function returning the identification numbers of all the nodes in tree T, i.e., if $T = (c, T_a, T_b, r_c)$, then

$$\text{Nodes}(T) = \begin{cases} \emptyset \text{ if } T = \emptyset, \\ \{c\} \cup \text{Nodes}(T_a) \cup \text{Nodes}(T_b) \text{ else.} \end{cases}$$

Definition A.2. Define $\text{Parent}(T, k)$ as a function returning identification number of the immediate parent of node k in a BHCC tree T, i.e.,

$$\text{Parent}(T, k) = \begin{cases} \emptyset \text{ if } k \text{ is root of } T, \\ \{c | T_c = (c, T_k, T_b, r_c) \vee T_c = (c, T_a, T_k, r_c)\} \text{ else.} \end{cases}$$

Definition A.3. Define $\text{Path}(T, k)$ as a function returning identification numbers of all the nodes in the path from the root to node k in a BHCC tree T, i.e.,

$$\text{Path}(T, k) = \begin{cases} \emptyset \text{ if } \{k\} \cap \text{Nodes}(T) = \emptyset, \\ \{k\} \cup \text{Path}(T, \text{Parent}(T, k)) \text{ else.} \end{cases}$$

Definition A.4. Define $\omega(T, k)$ as the posterior probability that $\mathcal{L}(T_k)$ forms a view and merging other subtrees to T_k does not yield a view:

$$\omega(T, k) = r_k \prod_{c \in \text{Path}(T,k) - \{k\}} (1 - r_c).$$

Lemma A.5. *The quantity defined in Equation 10 has a lower bound:*

$$\sum_{\boldsymbol{u} \in \text{Ptns}(T_c)} \prod_{v=1}^{\max(\boldsymbol{u})} \omega(T_c, k) p(\boldsymbol{x}_{\boldsymbol{u}=v} | X_{\cdot, \boldsymbol{u}=v}, \text{DPM}), \quad (14)$$

where the partitioning $\boldsymbol{u}$ is represented in the form of a vector of indicator variables, $\max(\boldsymbol{u})$ is the number of clusters in partitioning $\boldsymbol{u}$, and k is the node in T_c such that the two vectors $\boldsymbol{u} = v$ and $\mathcal{L}(T_k)$ have the same set of dimensions.

Proof. We show a proof by induction. If c is the leaf node, $\text{Ptns}(T_c) = (c)$, $\max(\boldsymbol{u}) = 1$, $k = c$ and $\omega(T_c, c) = r_c = 1$, thus Equation 14 becomes $p(\boldsymbol{x} | X_{\cdot, \mathcal{L}(T_c)}, \text{DPM})$ which is equal to the quantity in Equation 10. Thus the lemma is true in the base case.

Our inductive hypothesis is that the lemma holds for the two subtrees T_a and T_b. That is,

$$p(\boldsymbol{x}_{\mathcal{L}(T_a)} | X_{\mathcal{L}(T_a)}, T_a) \geq \sum_{\boldsymbol{u}' \in \text{Ptns}(T_a)} \prod_{v'=1}^{\max(\boldsymbol{u}')} \omega(T_a, k') p(\boldsymbol{x}_{\boldsymbol{u}'=v'} | X_{\cdot, \boldsymbol{u}'=v'}, \text{DPM})$$

and same for $p(\boldsymbol{x}_{\mathcal{L}(T_b)} | X_{\mathcal{L}(T_b)}, T_b)$; Also note that

$$\omega(T_a, k') \geq (1 - r_c)\omega(T_a, k') = \omega(T_c, k'),$$

and same for $\omega(T_b, k'')$. Therefore

$$\begin{aligned} &(1 - r_c) p(\boldsymbol{x}_{\mathcal{L}(T_a)} | X_{\mathcal{L}(T_a)}, T_a) p(\boldsymbol{x}_{\mathcal{L}(T_b)} | X_{\mathcal{L}(T_b)}, T_b) \geq \\ &\sum_{\boldsymbol{u}' \in \text{Ptns}(T_a)} \prod_{v'=1}^{\max(\boldsymbol{u}')} \omega(T_c, k') p(\boldsymbol{x}_{\boldsymbol{u}'=v'} | X_{\cdot, \boldsymbol{u}'=v'}, \text{DPM}) \times \\ &\sum_{\boldsymbol{u}'' \in \text{Ptns}(T_b)} \prod_{v''=1}^{\max(\boldsymbol{u}'')} \omega(T_c, k'') p(\boldsymbol{x}_{\boldsymbol{u}''=v''} | X_{\cdot, \boldsymbol{u}''=v''}, \text{DPM}) \\ &= \sum_{\boldsymbol{u} \in \text{Ptns}(T_a) \times \text{Ptns}(T_b)} \prod_{v=1}^{\max(\boldsymbol{u})} \omega(T_c, k) p(\boldsymbol{x}_{\boldsymbol{u}=v} | X_{\cdot, \boldsymbol{u}=v}, \text{DPM}) \end{aligned} \quad (15)$$

Meanwhile, for the trivial partitioning $(\mathcal{L}(T_c))$ (recalling that $(\mathcal{L}(T_c))$ represents the partitioning where all dimensions in $\mathcal{L}(T_c)$ are assigned to the same cluster), we have $\max(\boldsymbol{u}) = 1$, $k = c$ and $\omega(T_c, c) = r_c$. Thus for $\boldsymbol{u} = (\mathcal{L}(T_c))$

$$\begin{aligned} &\prod_{v=1}^{\max(\boldsymbol{u})} \omega(T_c, k) p(\boldsymbol{x}_{\boldsymbol{u}=v} | X_{\cdot, \boldsymbol{u}=v}, \text{DPM}) \\ &= r_c p(\boldsymbol{x}_{\mathcal{L}(T_c)} | X_{\cdot, \mathcal{L}(T_c)}, \text{DPM}) \end{aligned} \quad (16)$$

By definition, $\text{Ptns}(T_c) = (\mathcal{L}(T_c) \cup \text{Ptns}(T_a) \times \text{Ptns}(T_b)$. Therefore combining the results from Equation 15 and 16, we see the lemma is true. □

Lemma A.6. *The quantity defined in Equation 11 is equal to the quantity:*

$$\sum_{k \in \text{Nodes}(T_c)} \omega(T_c, k) p(\boldsymbol{y} | X_{\cdot, \mathcal{L}(T_k)}, \text{DPM}) \quad (17)$$

which sums over the prediction w.r.t. all the nodes in T_c weighted by the posterior of the nodes.

Proof. We show a proof by induction. If c is a leaf node, then $T_a = T_b = \emptyset$ and $r_c = 1$. By definition, $p(\boldsymbol{y} | X_{\cdot, \mathcal{L}(T_c)}, T_c) = p(\boldsymbol{y} | X_{\cdot, \mathcal{L}(T_c)}, \text{DPM})$; Meanwhile, $\text{Nodes}(T_c) = \{c\}$, $\omega(T_c, c) = r_c = 1$, $\mathcal{L}(T_c) = \{c\}$, thus Equation 17 becomes $p(\boldsymbol{y} | X_{\cdot, \mathcal{L}(T_c)}, \text{DPM})$. Thus the lemma is true in the base case.

Our inductive hypothesis is that Equation 17 holds for the two subtrees T_a and T_b, i.e.,

$$p(\boldsymbol{y} | X_{\cdot, \mathcal{L}(T_a)}, T_a) = \sum_{k \in \text{Nodes}(T_a)} \omega(T_a, k) p(\boldsymbol{y} | X_{\cdot, \mathcal{L}(T_k)}, \text{DPM})$$

and same for T_b. Meanwhile, by definition, $(1 - r_c)\omega(T_a, k) = \omega(T_c, k)$ and same for T_b; and $r_c =$

$\omega(T_c, c)$. Therefore,

$$p(\boldsymbol{y}|X_{\cdot,\mathcal{L}(T_c)}, T_c) = \omega(T_c, c)p(\boldsymbol{y}|X_{\cdot,\mathcal{L}(T_c)}, \text{DPM}) + \sum_{k\in\text{Nodes}(T_a)} \omega(T_c, k)p(\boldsymbol{y}|X_{\cdot,\mathcal{L}(T_k)}, \text{DPM}) + \sum_{k\in\text{Nodes}(T_b)} \omega(T_c, k)p(\boldsymbol{y}|X_{\cdot,\mathcal{L}(T_k)}, \text{DPM})$$

Further notice $\text{Nodes}(T_c) = \{c\} \cup \text{Nodes}(T_a) \cup \text{Nodes}(T_b)$, thus the lemma is true. □

Lemma A.7. *The quantity defined in Equation 13 is equal to the quantity:*

$$\sum_{k\in\text{Path}(T_c,j)} \omega(T_c, k)p(X_{i,j}|X_{\cdot,\mathcal{L}(T_k)}, \text{DPM}) \tag{18}$$

which sums over the prediction w.r.t. the nodes on the path from root to dimension j, weighted by the posterior of the nodes.

Proof. We show a proof by induction. If c is a leaf node, $\text{Path}(T_c, j) = \{c\} \cap \{j\}$, $\omega(T_c, c) = r_c = 1$, $\mathcal{L}(T_c) = \{c\}$, thus Equation 18 becomes $p(X_{i,j}|X_{\cdot,\mathcal{L}(T_c)}, \text{DPM}) \times |\{c\} \cap \{j\}|$ which is also the case in Equation 13 ($T_a = T_b = \emptyset$ and $r_c = 1$). Thus the lemma is true in the base case.

Our inductive hypothesis is that Equation 18 holds for the two subtrees T_a and T_b, i.e.,

$$p(X_{i,j}|X_{\cdot,\mathcal{L}(T_a)}, T_a) = \sum_{k\in\text{Path}(T_a,j)} \omega(T_a, k)p(X_{i,j}|X_{\cdot,\mathcal{L}(T_k)}, \text{DPM})$$

and same for T_b. Meanwhile, by definition, $(1 - r_c)\omega(T_a, k) = \omega(T_c, k)$ and same for T_b; and $r_c = \omega(T_c, c)$. Therefore,

$$p(X_{i,j}|X_{\cdot,\mathcal{L}(T_c)}, T_c) = \omega(T_c, c)p(X_{i,j}|X_{\cdot,\mathcal{L}(T_c)}, \text{DPM}) \times |\{j\} \cap \mathcal{L}(T_c)| + \sum_{k\in\text{Path}(T_a,j)} \omega(T_c, k)p(X_{i,j}|X_{\cdot,\mathcal{L}(T_k)}, \text{DPM}) + \sum_{k\in\text{Path}(T_b,j)} \omega(T_c, k)p(X_{i,j}|X_{\cdot,\mathcal{L}(T_k)}, \text{DPM})$$

Assume j is a leaf node of subtree T_a, then $\text{Path}(T_b, j) = \emptyset$, $\text{Path}(T_c, j) = \{c\} \cup \text{Path}(T_a, j)$ and $|\{j\} \cap \mathcal{L}(T_c)| = 1$, thus the lemma is true. □