Markov Beta Processes for Time Evolving Dictionary Learning

Amar Shah
Machine Learning Group
University of Cambridge
as793@cam.ac.uk

Zoubin Ghahramani
Machine Learning Group
University of Cambridge
zoubin@eng.cam.ac.uk

Abstract

We develop Markov beta processes (MBP) as a model suitable for data which can be represented by a sparse set of latent features which evolve over time. Most time evolving nonparametric latent feature models in the literature vary feature usage, but maintain a constant set of features over time. We show that being able to model features which themselves evolve over time results in the MBP outperforming other beta process based models. Our construction utilizes Poisson process operations, which leave each transformed beta process marginally beta process distributed. This allows one to analytically marginalize out latent beta processes, exploiting conjugacy when we couple them with Bernoulli processes, leading to a surprisingly elegant Gibbs MCMC scheme considering the expressiveness of the prior. We apply the model to the task of denoising and interpolating noisy image sequences and in predicting time evolving gene expression data, demonstrating superior performance to other beta process based methods.

1 INTRODUCTION

Latent variable models provide an intuitive way to study the structure of observed data. Pioneering examples of latent variable models are factor analyzers [Bartholomew, 1987] and finite mixture models [MacLachan and Peel, 2000]. Nonparametric Bayesian priors provide an elegant solution to the problem of inferring the number of latent features by sampling over latent feature representations with varying number of features a posteriori.

Griffiths and Ghahramani [2011] develop the *Indian buffet process* (IBP), a stochastic process on features which can be thought of as a factorial analog of the Chinese restaurant process. The IBP is a nonparametric Bayesian prior on binary matrices with an unbounded number of columns, which is exchangeable over the customers (or rows). The underlying measure which results in this exchangeability is of particular interest. Thibaux and Jordan [2007] show that the *beta process* is the underlying de Finetti mixing distribution which generates the Indian buffet process.

The beta process (BP) can be drawn as a Poisson process (with a particular Lévy measure), which opens many doors since Poisson processes have been studied in great depth [Kingman, 1993]. In particular, there exist operations which one can apply to a Poisson process draw such that the resultant object remains a draw from a Poisson process marginally. This construction has been exploited by Lin et al. [2010] and Chen et al. [2012] to develop dependent Dirichlet processes and dependent normalized random measures more generally.

In this work we apply Poisson process preserving operations to construct a Markov chain of beta processes. Each beta process is then used as a base measure for a sequence of Bernoulli process draws. Sampling an entire beta process at each 'time' step seems like a daunting task, however, the elegant consequence of the Poisson process preserving operations is that each beta process in the chain actually is marginally a draw from a beta process with an evolved form of base measure. This fact along with the conjugacy of the beta and Bernoulli processes permits us to marginalize over the entire chain of beta processes analytically.

Whilst previous attempts have been made to encode dependencies between feature usage i.e. between the binary matrices over multiple time steps [Williamson et al., 2010, Foulds et al., 2011], very little work has

been done on making features themselves evolve over time. Not only is our model capable of modeling a vast array of Markov dependencies amongst features, it permits an elegant Gibbs based inference algorithm which is efficiently able to learn structure amongst data. A key insight is that our model permits the analytic integration of a Dirichlet process used to model the time evolving features, and the entire Markov beta process chain. We believe that this is the first instance of dependent beta process work where such analytic integration is possible.

We utilize a Markov chain of beta processes for image denoising and inpainting tasks as well as for modeling time evolving gene expression data. Sparse methods have been successfully used for image analysis and gene data modeling. Our model, a natural extension of the beta Bernoulli model for time-evolving datasets, often outperforms other beta process based models on the tasks we consider.

2 BETA AND BERNOULLI PROCESSES

In this section, we review the beta, Bernoulli and Indian buffet processes following Thibaux and Jordan [2007], and proceed to discuss Poisson process properties Kingman [1993].

A *beta process* $B \sim \mathrm{BP}(c, B_0)$ is a positive random measure on a space Ω, where c is a positive function on Ω, and B_0 is a fixed measure on Ω, called the base measure. In our work, we assume c is constant. When B_0 is continuous, then a draw B can be represented as $B = \sum_{k=1}^{\infty} p_k \omega_k$, where ω_k are i.i.d. draws from $B/B(\Omega)$ and p_k are independent draws from a degenerate beta distribution with parameter c. If B_0 is discrete of the form $B_0 = \sum_k q_k \delta_{\omega_k}$, then $B = \sum_k p_k \delta_{\omega_k}$, with $p_k \sim \mathrm{Beta}(cq_k, c(1-q_k))$ independently. When B_0 is mixed discrete-continuous, B is generated as the sum of independent contributions from the discrete and continuous parts.

We now consider a draw $X \sim \mathrm{BeP}(B)$ from a *Bernoulli process*, for measure B on Ω. If B is continuous, then $X = \sum_{k=1}^{K} \delta_{\omega_k}$, where $K \sim \mathrm{Poisson}(B(\Omega))$, and ω_k are i.i.d. draws from $B/B(\Omega)$. If B is discrete and of the form $B = \sum_k p_k \delta_{\omega_k}$, then $X = \sum_k b_k \delta_{\omega_k}$, where the $b_k \sim \mathrm{Bernoulli}(p_k)$ independently.

Now consider the generative process $B \sim \mathrm{BP}(c, B_0)$ and $X_i | B \sim \mathrm{BeP}(B)$, for $i = 1, ..., n$. The posterior distribution of B is

$$B|\{X_i\}_{i=1:n} \sim \mathrm{BP}\Big(c+n, \frac{c}{c+n}B_0 + \frac{1}{c+n}\sum_{i=1}^{n} X_i\Big).$$

The BP is the conjugate prior for BeP and we can therefore integrate out B analytically when we consider sequential draws from the beta Bernoulli process

$$X_{n+1}|\{X_i\}_{i=1:n} \sim \mathrm{BeP}\Big(\frac{c}{c+n}B_0 + \frac{1}{c+n}\sum_{i=1}^{n} X_i\Big).$$

Note that $\frac{1}{c+n}\sum_{i=1}^{n} X_i \equiv \sum_k \frac{m_{n,k}}{c+n}\delta_{\omega_k}$, where ω_k represent the unique atoms selected by the first n data points and $m_{n,k}$ represents how many of the first n data points selected the k^{th} atom. Note that $X_{n+1}|\{X_i\}_{i=1:n}$ is sampled from the sum of two contributions: one from $\mathrm{BeP}(\frac{c}{c+n}B_0)$ and the other from $\mathrm{BeP}(\frac{1}{c+n}\sum_{i=1}^{n} X_i)$. This sampling process is equivalent to that of the Indian buffet process prior [Griffiths and Ghahramani, 2011].

When B_0 is a non-atomic measure, a draw from a beta process $B \sim \mathrm{BP}(c, B_0)$ is equivalent to a Poisson process (PP) draw with base measure ν, where

$$\nu(d\omega, dp) = cp^{-1}(1-p)^{c-1}dpB_0(d\omega). \qquad (1)$$

The Poisson process draw will consist of points of the form $(\omega_k, p_k) \in [0,1] \times \Omega$, which should be used to define $B = \sum_k p_k \delta_{\omega_k}$. Poisson process preserving operations are operations one can apply to a draw from a Poisson process such that the resultant draw remains marginally a draw from a Poisson process, with a modified base measure. We utilize such operations to construct dependent beta process, in particular, we consider *subsampling, point transition* and *superposition.* See Kingman [1993] for a review of Poisson process properties.

3 MARKOV BETA PROCESSES

3.1 CONSTRUCTION

Given a draw from a beta process, we construct a related beta process by 1) partitioning atoms into clusters, 2) applying stochastic transformations to atoms in each cluster and 3) recombining all the atoms to give a new atomic base measure. We shall show that the transformed beta process draw is marginally a beta process, by applying the theory of Poisson processes.

Definition 1. *A probabilistic transition function is a function* $\boldsymbol{\mathcal{T}} : \Omega \to \mathbb{R}_+$*, such that for each* $\omega \in \Omega$*,* $\boldsymbol{\mathcal{T}}(\omega)$

is a probability measure on Ω. A measure, μ, over Ω is transformed by $\boldsymbol{\mathcal{T}}$ to give measure $\boldsymbol{\mathcal{T}}[\mu]$, defined as

$$(\boldsymbol{\mathcal{T}}[\mu])(A) := \int_{\Omega} \boldsymbol{\mathcal{T}}(\omega)(A)\mu(d\omega). \tag{2}$$

for each measurable $A \subseteq \Omega$. We denote a sample from $\boldsymbol{\mathcal{T}}(\omega)$ as $\mathcal{T}(\omega)$.

Our approach is to consider parameteric probabilistic transition functions, $\boldsymbol{\mathcal{T}}_\theta$, for parameters θ from some measurable parameter space Θ. An example of a probabilistic transition function is a Gaussian probability density function e.g. $\boldsymbol{\mathcal{T}}_\theta(\omega) = \mathcal{N}(\omega; 0, \theta)$.

In order to partition the set of atoms and decide how they should be transitioned, we use a draw from a Dirichlet process, $\boldsymbol{\mathcal{D}} = \sum_j \lambda_j \delta_{\theta_j} \sim \mathrm{DP}(\alpha, H)$, for a DP concentration parameter α and a base measure, H, on Θ. The λ_j are non-negative and sum to 1. We define the probabilistic transition $\boldsymbol{\mathcal{T}}_{\boldsymbol{\mathcal{D}}} = \sum_j \lambda_j \boldsymbol{\mathcal{T}}_{\theta_j}$, which we can think of as a convex combination of probabilistic transition functions. The probabilities, λ_j, are used to define a multinomial distribution which is used to decide to which cluster each atom would be assigned to. Once clustered, atoms in cluster j are transitioned using $\boldsymbol{\mathcal{T}}_{\theta_j}$.

Let B_0 be a base measure on Ω, and $B_1 = \sum_k q_k \delta_{\omega_k} \sim \mathrm{BP}(c, B_0)$. For each k, sample $u_k \sim \mathrm{Mult}(1, \boldsymbol{\lambda})$, and $\mathcal{T}_{\theta_{u_k}}(\omega_k) \sim \boldsymbol{\mathcal{T}}_{\theta_{u_k}}(\omega_k)$. Finally, set $B_1 = \sum_k q_k \delta_{\mathcal{T}_{\theta_{u_k}}(\omega_k)}$. We prove that marginally, B_1 is a draw from a beta process.

Lemma 2. *If B_1 is constructed as above, then marginally, $B_1 \sim \mathrm{BP}(c, \boldsymbol{\mathcal{T}}_*[B_0])$ for measure $\boldsymbol{\mathcal{T}}_*[B_0]$ defined such that for $A \subseteq \Omega$, $\boldsymbol{\mathcal{T}}_*[B_0](A) \equiv \int (\boldsymbol{\mathcal{T}}_{\boldsymbol{\mathcal{D}}}[B_0])(A)\mathrm{DP}(\boldsymbol{\mathcal{D}}; \alpha, H)d\boldsymbol{\mathcal{D}}$*

Proof. The distribution of q_k remain unchanged. Also $\boldsymbol{\mathcal{T}}_{\boldsymbol{\mathcal{D}}}[B_0](\Omega) = \sum_j \lambda_j \boldsymbol{\mathcal{T}}_{\theta_j}[B_0](\Omega) = B_0(\Omega)$. Integrating over $\boldsymbol{\mathcal{D}}$ and s_k, the new location of atom k is marginally distributed as $\boldsymbol{\mathcal{T}}_*[B_0]/B_0(\Omega)$. Hence B_1 is a PP draw with base measure $\tilde{\nu}(d\omega, dp) = cp^{-1}(1-p)^{c-1}dp\boldsymbol{\mathcal{T}}_*[B_0](d\omega)$, making it marginally a beta process draw. □

Whilst we have not explicitly invoked theorems regarding Poisson preserving operations, they provide some intuition as to why the transformed beta processes are marginally beta process draws. The operation of clustering the atoms of the original beta process is analogous to the Poisson process operation of *subsampling*. A subsampled Poisson process is marginally a Poisson process. *Point transition* is a Poisson process preserving operation, as is the *superposition* of Poisson process draws. The composition of these three Poisson preserving operations accurately describes the nature of our construction of dependent beta processes. An illustration of the formulation is seen in Figure 1.

Note that the Dirichlet process prior can be replaced with other priors without affecting the proof. We choose a DP in our experiments for its modeling flexibility.

Repeating the above approach to construct $B_2, B_3, ...$ results in a *Markov chain of beta processes*. Define $\boldsymbol{\mathcal{T}}^t_{\boldsymbol{\mathcal{D}}}[B_0]$ to be the measure resulting in applying $\boldsymbol{\mathcal{T}}_{\boldsymbol{\mathcal{D}}}$ t times to measure B_0. Then marginally, $B_t \sim \mathrm{BP}(c, \boldsymbol{\mathcal{T}}^t_*[B_0])$, where for $A \subseteq \Omega$, $\boldsymbol{\mathcal{T}}^t_*[B_0](A) = \int (\boldsymbol{\mathcal{T}}^t_{\boldsymbol{\mathcal{D}}}[B_0])(A)\mathrm{DP}(\boldsymbol{\mathcal{D}}; \alpha, H)d\boldsymbol{\mathcal{D}}$. Henceforth we refer to our construction of dependent beta processes as *Markov beta processes*. When $B_1, ..., B_T$ is a draw from a Markov beta process, we write $B_1, ..., B_T, \boldsymbol{u}_1, ..., \boldsymbol{u}_T | \boldsymbol{\mathcal{D}} \sim \mathrm{MBP}(c, B_0, \boldsymbol{\mathcal{D}}, \boldsymbol{\mathcal{T}}, T)$.

To the best of out knowledge, our construction of dependent beta processes is the first to leave each transformed beta process marginally a beta process draw. Most existing models all keep features (atoms) constant but vary the feature probabilities q_k over time or space. We discuss the related models in the next subsection.

3.2 RELATED DEPENDENT BETA PROCESS CONSTRUCTIONS

In our construction of Markov dependent beta processes, the atom locations evolve over time, whilst the atom weights remain constant. This is in contrast to the majority of existing constructions of dependent beta processes. Existing research has tended to focus on maintaining a time invariant set of atoms, whilst varying the weights.

The Markov IBP introduced by Van Gael et al. [2008] is a model where features remain constant over time, but feature allocations follow a Markov chain on the space $\{0, 1\}$ (1 indicates presence of a feature and 0 indicates absence). The transition matrix governing the chain also remains constant over time, suggesting that the Markov IBP would be unfit to model non-stationary data. Most importantly, this model does not have the ability to jointly model multiple related time series unlike the model we propose in this work. The authors propose several inference schemes for the Markov IBP, but comment that adequate performance is only observed with more complicated algorithms.

Foulds et al. [2011] use a similar idea to the Markov

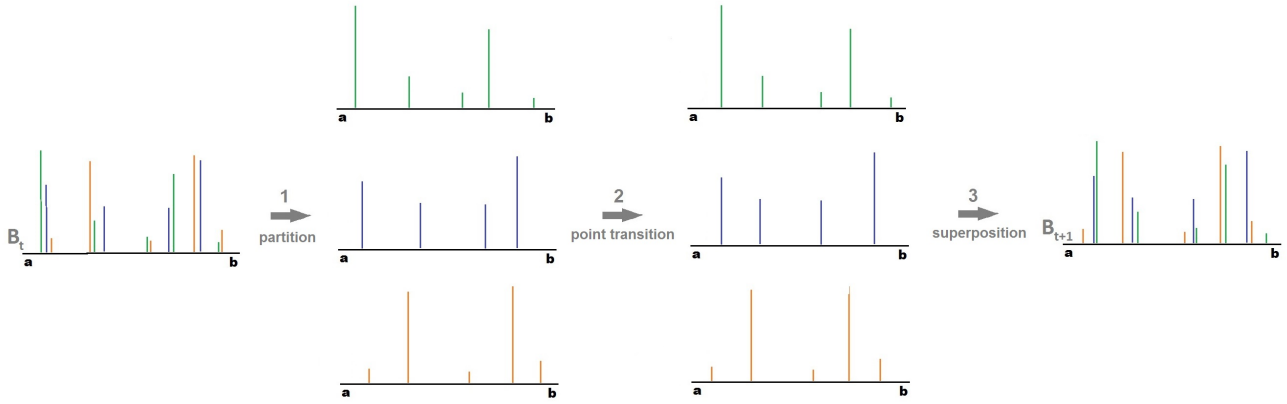


Figure 1: Example illustrating how B_{t+1} is constructed from B_t. First the atoms are partitioned or clustered into, in this case, 3 clusters. Next, atoms undergo probabilistic point transition based on the cluster they belong to. Finally atoms are superimposed to create B_{t+1}.

IBP for modeling time evolving social networks. Their DRIFT model incorporates a Markov process to decide whether or not features are present at each time step whilst maintaining constant features over time. Unlike the Markov IBP, DRIFT is able to model an arbitrary number of items.

Williamson et al. [2010] construct dependent IBPs using Gaussian process draws and the stick breaking construction of the IBP [Teh et al., 2007] to introduce dependencies between customers for each feature as well as temporal dependencies. The dIBP has high flexibility due to the nonparametric temporal dependence structure, but this comes with a high computational cost. Features also remain constant over time in the dIBP model, unlike in our MBP model.

The kernel beta process [Ren et al., 2011] is a generalization of the beta process and has the form $B_x = \sum_k \pi_k K(x, x^*|\psi)\delta_{\omega_k}$, where x belongs to a covariate space $\mathcal{X}$ and K is a kernel function on $\mathcal{X} \times \mathcal{X}$ taking values in $[0, 1]$ for parameters x^* and ψ. The use of the kernel allows weights to vary over a covariate space, whilst the atom locations remain fixed.

The dependent hierarchical beta process Zhou et al. [2011] is constructed as a convex combination of i.i.d. draws from a hierarchical beta process. The convex weights are made to depend on a kernel over a covariate space. More specifically, a kernel function is defined between the covariates (each data point has one covariate), inducing correlation of feature usage between data points. The covariate being used is the location of the patch within the entire image, the idea being that neighbouring patches are more likely to use similar dictionary elements for image denoising and interpolation. The way it has been constructed by the authors, it would not explicitly be able to model temporal structure amongst image sequences.

3.3 COLLAPSED GIBBS SAMPLER

None of the models discussed above result in simple marginal distributions, and consequently each use an uncollapsed Gibbs sampler to perform posterior inference. Nonparametric Bayesian priors are usually easy to sample from, but posterior inference is typically difficult because of the flexibility of the models and the presence of poor local optima in the posterior conditional distributions of the latent variables. Whilst an uncollapsed Gibbs sampler algorithms are usually easy to implement, they are prone to get stuck in poor modes and converge much more slowly than collapsed samplers [Doshi-Velez and Ghahramani, 2009]. In our Markov beta processes, having marginal beta process distributions permits a marginalized posterior sampler which has a better chance of mixing well and has an elegant restaurant analogy.

At each time step, t, we draw $x_t^i|B_t \sim \text{BeP}(B_t)$ iid for $i = 1, ..., n_t$. Let $X_t \equiv \{x_t^i\}_{i=1}^{n_t}$. We show how to sample each x_t^i marginalizing out the Markov beta process chain in the following lemma and corollary. For $t > 1$, we let ω_t^j denote a sample from (a) $\boldsymbol{\mathcal{T}}_{\boldsymbol{\mathcal{D}}}(\omega_{t-1}^j)$ if $j \leq k_{t-1}$, or (b) $\boldsymbol{\mathcal{T}}_{\boldsymbol{\mathcal{D}}}^{t-1}[B_0]/B_0(\Omega)$ if $j > k_{t-1}$.

Lemma 3. *For* $c_t = c + \sum_{s=1}^t n_s$,

$$B_t|\boldsymbol{\mathcal{D}}, B_0, X_{1:t} \sim \text{BP}\left(c_t, \frac{c}{c_t}\boldsymbol{\mathcal{T}}_{\boldsymbol{\mathcal{D}}}^{t-1}[B_0] + \sum_{j=1}^{k_t} \frac{m_t^j}{c_t}\delta_{\omega_t^j}\right)$$

Proof. We prove the claim by induction. Note that

$B_1|\boldsymbol{\mathcal{D}}, B_0 \sim \mathrm{BP}(c, \boldsymbol{\mathcal{T}}_{\boldsymbol{\mathcal{D}}}[B_0])$ by Lemma 2. By conjugacy of the beta and Bernoulli processes, we then have $B_1|\boldsymbol{\mathcal{D}}, B_0, X_1 \sim \mathrm{BP}\big(c_1, \frac{c}{c_1}\boldsymbol{\mathcal{T}}_{\boldsymbol{\mathcal{D}}}[B_0] + \sum_{j=1}^{k_1} \frac{m_1^j}{c_1}\delta_{\omega_1^j}\big)$. Now let $t > 1$. By induction and Lemma 2, $B_t|\boldsymbol{\mathcal{D}}, B_0, X_{1:t-1} \sim \mathrm{BP}\Big(c_{t-1}, \frac{c}{c_{t-1}}\boldsymbol{\mathcal{T}}_{\boldsymbol{\mathcal{D}}}^{t-1}[B_0] + \sum_{j=1}^{k_{t-1}} \frac{m_{t-1}^j}{c_{t-1}}\delta_{\omega_t^j}\Big)$. The final result follows from the conjugacy of the beta and Bernoulli processes. □

Corollary 4.

$$x_t^{n_t+1}|\boldsymbol{\mathcal{D}}, B_0, X_{1:t} \sim \mathrm{BeP}\left(\frac{c}{c_t}\boldsymbol{\mathcal{T}}_{\boldsymbol{\mathcal{D}}}^{t-1}[B_0] + \sum_{j=1}^{k_t}\frac{m_t^j}{c_t}\delta_{\omega_t^j}\right)$$

Proof. Since $x_t^{n_t+1}|B_t \sim \mathrm{BeP}(B_t)$ and $p(x_t^\circ|\boldsymbol{\mathcal{D}}, B_0, X_{1:t}) = \int p(x_t^\circ|B_t)p(B_t|\boldsymbol{\mathcal{D}}, B_0, X_{1:t})dB_t$, the result follows from conjugacy and Lemma 3. □

The result in Corollary 4 motivates a sampling scheme with a restaurant analogy similar to the one developed for the Indian buffet process [Griffiths and Ghahramani, 2011]. On day $t = 1$, customer $i_1 = 1$ tastes a number of dishes sampled from Poisson$(cB_0(\Omega))$. He tries dishes in Ω sampled independently from probability measure $B_0/B_0(\Omega)$. Suppose $i_1 > 1$ and let m^j be the number of people who have tasted the j^{th} dish so far. Customer i_1 tries each of the existing dishes independently with probability $\frac{m^j}{c+i_1-1}$, and samples Poisson$\big(\frac{c}{c+i_1-1}B_0(\Omega)\big)$ new dishes. On a new day, $t > 1$, each dish from the previous day evolves stochastically using the probabilistic transition function $\boldsymbol{\mathcal{T}}_{\boldsymbol{\mathcal{D}}}$. Customer i_t tries each existing dish with independent probability $\frac{m^j}{c_{t-1}+i_t-1}$, and samples Poisson$\big(\frac{c}{c_{t-1}+i_t-1}B_0(\Omega)\big)$ new dishes independently from $\boldsymbol{\mathcal{T}}_{\boldsymbol{\mathcal{D}}}^{t-1}[B_0]/B_0(\Omega)$.

As is the case for the IBP, dishes which are popular are more likely to be tried over time. The key difference in our Markov beta process framework, is that the dishes evolve stochastically over time, and the base distribution from which new dishes are drawn also evolves stochastically over time.

4 TIME EVOLVING DICTIONARY LEARNING

In this section we illustrate how a draw from a Markov beta process prior may be used to perform time evolving dictionary learning. Each $\omega_k \in \Omega$ which appears in a beta process draw is a dictionary element, with each associated q_k being the probability that a data point uses ω_k. We consider the case $\Omega = \mathbb{R}^P$, and refer to dictionary element k at time t as a column vector, $\boldsymbol{d}_t^k$. The entire dictionary at time t is a matrix denoted $\mathbf{D}_t \equiv [\boldsymbol{d}_t^1, ..., \boldsymbol{d}_t^{k_t}] \in \mathbb{R}^{P\times k_t}$. We may model datapoint i at time t, $\boldsymbol{x}_t^i$, as

$$\boldsymbol{x}_t^i = \mathbf{D}_t(\boldsymbol{s}_t^i \circ \boldsymbol{z}_t^i) + \boldsymbol{\epsilon}_t^i, \tag{3}$$

where $\circ$ represents the Hadamard product, $\boldsymbol{s}_t^i, \boldsymbol{\epsilon}_t^i \in \mathbb{R}^{k_t}$ and $\boldsymbol{z}_t^i \in \{0,1\}^{k_t}$. z_t^{ik} represents whether or not $\boldsymbol{x}_t^i$ uses dictionary element k and is a draw from Bernoulli(q_k). s_t^{ik} determines how much of dictionary element k to use and $\boldsymbol{\epsilon}_t^i$ is simply additive noise.

A natural probabilistic transition function in the case $\Omega = \mathbb{R}^P$, is a Gaussian distribution. We set $\Theta = \mathbb{R}_+^P$ and define $\boldsymbol{\mathcal{T}}_{\boldsymbol{\theta}}(\boldsymbol{d})$ to be a multivariate Gaussian probability density function with mean $\boldsymbol{d}$ and diagonal covariance matrix with diagonal $\boldsymbol{\theta}^{-1}$.

We can generate data from $t = 1, ..., T$, $k = 1, ..., k_t$ and $i = 1, ..., n_t$ as follows,

$$\boldsymbol{\mathcal{D}} = \sum_j \lambda_j \delta_{\boldsymbol{\theta}_j} \sim \mathrm{DP}(\alpha, H), \tag{4}$$

$$\begin{aligned}\mathbf{D}_1, ..., \mathbf{D}_T, \boldsymbol{q}, \boldsymbol{u}_1, ..., \boldsymbol{u}_{T-1}|\boldsymbol{\mathcal{D}} &\sim \mathrm{MBP}(c, B_0, \boldsymbol{\mathcal{D}}, \boldsymbol{\mathcal{T}}, T),\\ z_t^{ik}|q_k &\sim \mathrm{Bernoulli}(q_k),\\ s_t^{ik} &\sim \mathcal{N}(0, \gamma_s^{-1})\\ \boldsymbol{x}_t^i|\mathbf{D}_{\boldsymbol{t}}, \boldsymbol{s}_t^i, \boldsymbol{z}_t^i &\sim \mathcal{N}\big(\mathbf{D}_{\boldsymbol{t}}(\boldsymbol{s}_t^i \circ \boldsymbol{z}_t^i), \gamma_\epsilon^{-1}\mathbf{I}_P\big),\end{aligned}$$

where H is a measure on $\Theta = \mathbb{R}_+^P$ which is the product of P Gamma measures of the form Gamma$(1, P)$ on each component and B_0 is a multivariate Gaussian measure of the form $\mathcal{N}\big(\mathbf{0}, \frac{1}{P}\mathbf{I}\big)$. We place Gamma priors on γ_s and γ_ϵ.

4.1 INFERENCE

We train the model based on a collapsed Gibbs based MCMC scheme, where the probabilities q_k are marginalized out. The update equations are summarized below.

Update $\mathbf{D}_t$. The posterior density for $\boldsymbol{d}_t^k$ is

$$\begin{aligned}p(\boldsymbol{d}_t^k|-) \sim \exp\Big(&-\frac{1}{2}\sum_{p=1}^P \theta_{u_{t-1}^k,p}\big(d_{t,p}^k - d_{t-1,p}^k\big)^2\\ &-\frac{1}{2}\sum_{p=1}^P \theta_{u_t^k,p}\big(d_{t+1}^k - d_t^k\big)^2\\ &-\frac{\gamma_\epsilon}{2}\sum_{i=1}^N \big\|\boldsymbol{\psi}_t^{i,-k} - \boldsymbol{d}_t^k(s_t^{i,k}z_t^{i,k})\big\|^2\Big)\end{aligned}$$

where $\boldsymbol{\psi}_t^{i,-k} = \boldsymbol{x}_t^i - \sum_{k' \neq k} s_t^{ik} z_t^{ik} \boldsymbol{d}_t^k$.
Hence $\boldsymbol{d}_t^k | - \sim \mathcal{N}(\boldsymbol{\phi}, \boldsymbol{\Phi})$, where

$$\boldsymbol{\Phi} = \Big(\operatorname{diag}\Big(\boldsymbol{\theta}_{u_{t-1}^k} + \boldsymbol{\theta}_{u_t^k} \Big) + \gamma_\epsilon \sum_{i: z_t^{i,k}=1} {s_t^{i,k}}^2 \Big)^{-1} \mathbf{I}_P$$

$$\boldsymbol{\phi} = \boldsymbol{\Phi} \Big(\operatorname{diag}\Big(\boldsymbol{\theta}_{u_{t-1}^k} \Big) \boldsymbol{d}_{t-1}^k + \operatorname{diag}\Big(\boldsymbol{\theta}_{u_t^k} \Big) \boldsymbol{d}_{t+1}^k + \gamma_\epsilon \sum_{i: z_t^{i,k}=1} s_t^{i,k} \boldsymbol{\psi}_t^{i,-k} \Big)$$

Update u_t^k. The posterior density for $\boldsymbol{u}_t^k$ is

$$p(\boldsymbol{u}_t^k = j | -) \propto \lambda_j \prod_{p=1}^{P} \Big[\sqrt{\theta_{j,p}} \exp\Big(-\frac{1}{2} \theta_{j,p} {\delta_{t,p}^k}^2 \Big) \Big]$$

where $\delta_{t,p}^k = \big(d_{t+1,p}^k - d_{t,p}^k \big)$, a multinomial distribution.

Update $z_t^{i,k}$. The posterior odds of $z_t^{i,k}$ is

$$\frac{p(z_t^{i,k}=1|-)}{p(z_t^{i,k}=0|-)} = \frac{\exp\Big(-\frac{\gamma_\epsilon}{2} \big\| \boldsymbol{\psi}_t^{i,-k} - s_t^{i,k} \boldsymbol{d}_t^k \big\|^2 \Big) \pi_t^{i,k}}{\exp\Big(-\frac{\gamma_\epsilon}{2} \big\| \boldsymbol{\psi}_t^{i,-k} \big\|^2 \Big) (1 - \pi_t^{i,k})},$$

where

$$\pi_t^{i,k} = \frac{\sum_{s=1}^{T} \sum_{j=1}^{k_s} z_s^{i,j} - z_t^{i,k}}{c + \sum_{s=1}^{T} n_s - 1}.$$

Update $s_t^{i,k}$. The posterior density for $s_t^{i,k}$ is

$$p(s_t^{i,k}|-) \propto \exp\Big(-\frac{\gamma_\epsilon}{2} \big\| \boldsymbol{\psi}_t^{i,-k} - s_t^{i,k} \boldsymbol{d}_t^k \big\|^2 - \frac{\gamma_s}{2} {s_t^{i,k}}^2 \Big),$$

hence $s_t^{i,k} \sim \mathcal{N}(\varphi, \kappa)$, where

$$\kappa = \Big(\gamma_\epsilon {\boldsymbol{d}_t^k}^\top \boldsymbol{d}_t^k + \gamma_s \Big)^{-1}$$
$$\varphi = \kappa \Big(\gamma_\epsilon {\boldsymbol{d}_t^k}^\top \boldsymbol{\psi}_t^{i,-k} \Big).$$

Update $\boldsymbol{\lambda}$. The posterior distribution of $\boldsymbol{\lambda}$ is

$$p(\boldsymbol{\lambda}|-) \propto \prod_j {\lambda_j}^{\alpha + \sum_{s=t}^{T-1} \sum_{k=1}^{k_s} \mathbb{I}[u_t^k = j] - 1}.$$

Update $\boldsymbol{\theta}_j$. The posterior distribution of $\boldsymbol{\theta}_j$ is

$$p(\boldsymbol{\theta}_j|-) \propto \exp\Big(-\frac{P}{2} {\boldsymbol{\theta}_j}^\top \boldsymbol{\theta}_j - \frac{1}{2} \sum_{t=1}^{T-1} \sum_{k=1}^{k_t} \mathbb{I}[u_t^k = j] {\boldsymbol{\theta}_j}^\top \big(\boldsymbol{\delta}_t^k \circ \boldsymbol{\delta}_t^k \big) \Big),$$

hence $\boldsymbol{\theta}_j | - \sim \mathcal{N}\big(\boldsymbol{\varsigma}_j, \frac{1}{P} \mathbf{I} \big)$, where

$$\boldsymbol{\varsigma}_j = \frac{1}{2P} \sum_{t=1}^{T-1} \sum_{k=1}^{k_t} \mathbb{I}[u_t^k = j] \big(\boldsymbol{\delta}_t^k \circ \boldsymbol{\delta}_t^k \big).$$

5 EXPERIMENTS

In this section we describe the findings of various experiments we performed using Markov beta processes to induce a chain of evolving latent features. We investigated two tasks, the problem of denoising and inpainting a sequence of images and time evoloving gene expression data. First we try to answer various questions with synthetic experiments.

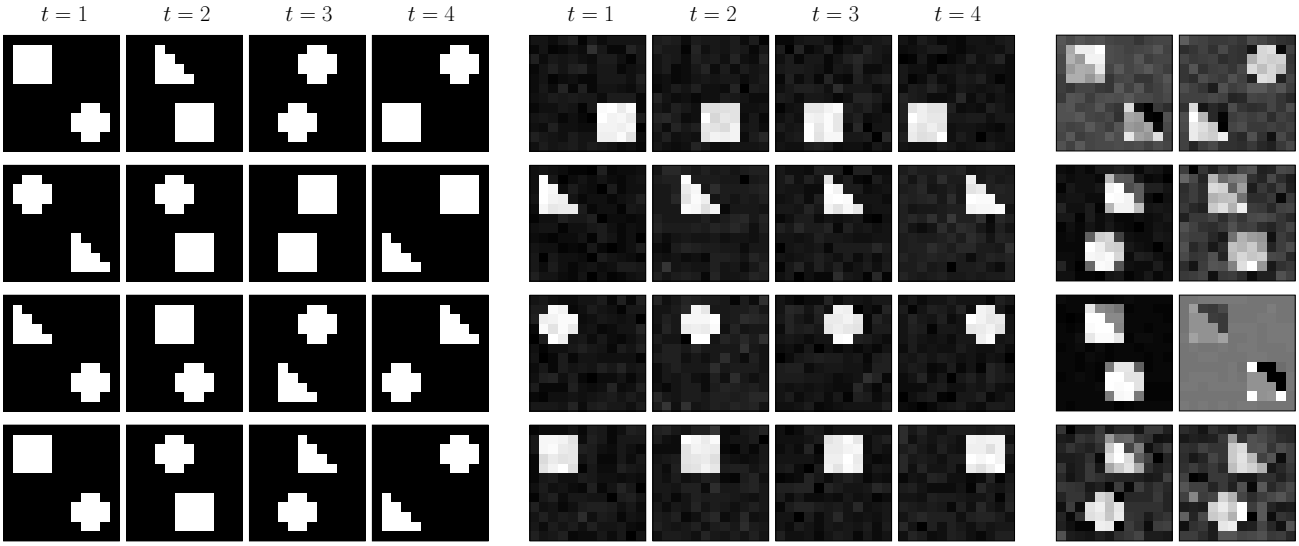
5.1 SYNTHETIC EXPERIMENTS

A key quesion we set out to address in this work is 'are Markov evolving features more useful for multiple sequence modeling than Markov evolving feature probabilities?' When there are no clear task specific reasons to choose one approach over the other, one would want to choose the framework that is generally more reliable to make useful predictions. Setting $N = 500, D = 40, K = 50$, we generated Synth-MBP from the MBP prior, and Synth-DRIFT from the DRIFT model. The MBP, DRIFT and BP models were trained on these datasets with 10% of the points randomly chosen and held out for prediction. Table 1 summarizes the test mean squared errors of predicitons of the three models.

We initialize features using random subsets of the observable data and use a k-means initialization for the clustering of the feature transitions for the MBP model. The Z and S matrices are subsequently initialized with a linear least squares estimate given the features. The same approach is used in all subsequent experiments.

Both the MBP and DRIFT models outperform the BP model, suggesting that each of them are able to learn some level of temporal structure, as we would have liked. However, it is interesting to see that the MBP model's outperformance versus the DRIFT model on Synth-MBP is much larger than the DRIFT model's outperformance versus the MBP model on Synth-DRIFT. The Dirichlet process transition process between features appears to be responsible for the MBP's predictive power on the DRIFT-Synth data set. We replaced the DP with a single Gaussian transition function for all features, and found the mean squared errors dropped to 2.04 and 2.11 on MBP-Synth and DRIFT-Synth respectively.

The fact that we are able to utilize a flexible DP based transition function using a collapsed Gibbs sampler which mixes fast is key to the high performance of the MBP model. Inference in models with complicated dependencies between binary feature usage variables is more difficult. Our MBP model is useful because of the efficient Gibbs based sampler we are able to derive



(a) Synthetic data (b) Time-evolving features learned by MBP (c) DRIFT features

Figure 2: Synthetic image data experimental results. (a) Each row represents an image sequence used for the synthetic image experiment. (b) Time-evolving features learned by the MBP model. (c) Stationary features learned by the DRIFT model.

Table 1: Mean-squared error results of prediction on synthetic datasets using MBP, DRIFT and BP models, with standard deviation of multiple runs in brackets.

	MBP	DRIFT	BP
SYNTH-MBP	**1.03 (0.04)**	2.16 (0.08)	2.84 (0.10)
SYNTH-DRIFT	1.66 (0.05)	**1.37 (0.07)**	2.45 (0.09)

for modeling expressive transitions between features.

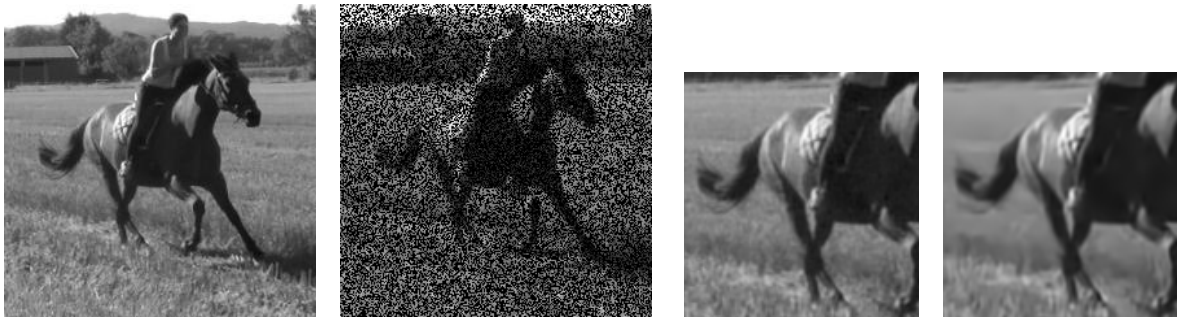
5.2 IMAGE DENOISING AND INPAINTING

Sequences of grayscale images were considered for this task, but our method easily extends to color image sequences. The baseline model we compare our model to, is the beta process model of Zhou et al. [2009], modeling each image within the sequence independently. We implement the Foulds et al. [2011] model to assess the benefit of time-evolving feature usage versus time-evolving features and finally, we consider the dHBP [Zhou et al., 2011], which is designed to exploit intra-image dependencies. Inference in the dIBP [Williamson et al., 2010] unfortunately scales in N^3 which is prohibitive for this task.

Consider a sequence of T images of size $Q_x \times Q_y$. We model overlapping patches of size 8×8 as individual data points, giving a total of $N = (Q_x - 7) \times (Q_y - 7)$ data points, each with dimension $D = 8$. Whilst this breaks the exchangeability assumption of the prior, we benefit from model averaging, as each final pixel estimate is in fact an average of the estimates of 64 patches (except for near edge pixels).

Our first experiment involved synthetically generated data of the form shown in Figure 2a. We formed $N = 500$ sequences of images of size 12×12. In each sequence, we observe shapes traverse left to right in the top half of the images and other shapes traverse right to left. We trained both the MBP and the DRIFT models on this dataset using $K = 75$ features. The MBP was able to learn interesting structure amongst features as can be seen in Figure 2b. The model was able to identify the individual shapes which traversed left and right across the image over time. Since the DRIFT model uses a stationary set of features, it tended to learn features as noisy compositions of various data points 2c, not exhibiting the more elegant structure we found amongst the MBP features.

Notice that the shapes in the synthetic sequences often switch from frame to frame; squares become triangles, crosses become squares, etc. The MBP model is capable of modeling these transitions because whilst feature probabilities are stationary, actual feature usage between time points is conditionally independent given the feature probabilies. (More succinctly, $z_{ik}^t \perp\!\!\!\perp z_{ik}^{t'} | \pi_k$ for $t \neq t'$.) On average, the DP transition function of the MBP model used 16 clusters for the synthetic image data. This was somewhat crucial in being able to learn the time-evolving features we see in Figure 2b.



(a) (b) (c) (d)

Figure 3: Image sequence experiment on horse data. (a) Image 6 of the sequence. (b) Image with 50% of pixels omitted at random with pixels corrupted with Gaussian noise. Zoomed reconstructions of (c) MBP and (d) dHBP models.

We ran an experiment restricting the MBP model to have a single transition function, and noticed that the model learned features more similar to those learned by the DRIFT model.

We test the methods on four black and white image sequences: (i) 8 frames of size 241×241 of a lady riding a horse in a field [Ochs et al., 2014], (ii) 8 frames of size 192×256 with a set of mountains being panned about a fixed axis [Porzi et al., 2014], (iii) 8 frames of size 120×216 of a rabbit leaping across a living room [Ochs et al., 2014], and (iv) 10 frames of size 60×64 of a hand holding a bowl and rotating it [Wang, 1997].

Pixels in these datasets take integer values in $[0, 255]$. We added Gaussian noise with standard deviation 15 independently to each pixel. For each image sequence, we sample a binary matrix, Σ, of size $Q_x \times Q_y$, where each entry is an independent Bernoulli sample. Σ indicates which pixels are used for training by each algorithm for each image sequence. A typical evaluation metric for image reconstruction is the peak signal-to-noise ratio (PSNR), defined as

$$\text{MSE} = \sum_{t=1}^{T} \sum_{q_x=1}^{Q_x} \sum_{q_y=1}^{Q_y} \frac{\left(\mathbf{\Pi}_t(q_x, q_y) - \hat{\mathbf{\Pi}}_t(q_x, q_y)\right)^2}{TQ_xQ_y}$$

$$\text{PSNR} = 10 \log_{10} \left(\frac{255^2}{\text{MSE}} \right), \tag{5}$$

where $\mathbf{\Pi}_t$ is the original $Q_x \times Q_y$ image at time t and $\hat{\mathbf{\Pi}}_t$ is the estimate of this image. We allow the samplers to burn-in for 2000 iterations, and then use 500 Gibbs samples for predictions, using $K = 250$ features.

The experiments on each set of image sequences were repeated using 20%, 30% and 50% of the pixels chosen uniformly at random for training, the results are summarized in Table 2. The MBP approach tends to outperform other methods consistently, in particular the DRIFT model which does encode time evolving feature usage. The dHBP model, designed particularly for the task of image denoising and inpainting does well on most sequence tasks without modeling temporal dependencies, but on the whole does not perform as well as the MBP. The MBP does not model the intra-image dependencies, and we believe that combining the temporal modeling of the MBP with the intra-image dependencies of the dHBP would lead to a highly sophisticated image sequence denoising and inpainting framework. Since our focus in this paper is on the MBP model and not the specific task of image sequence analysis, we leave the combined model idea to future work.

Figure 3 illustrates a situation where the dHBP performs worse than the MBP. The dHBP model uses patches which are close in Euclidean distances to use similar sets of features. A consequence of this property, is that the area of the image under the body of the horse, appears to have a uniform color. This is likely because the bottom section of the horse body also has uniform color. Conversely the MBP model is better able to learn features which are separate for the horse body and the background grass, leading to a clearer reconstruction. The tail in the dHBP reconstruction also exhibits more smoothing and less detail than that of the MBP.

5.3 GENE EXPRESSION DATA

Time-course gene expression data are often measured to study dynamic biological systems and gene-regulatory networks. Vast amounts of biological data are being collected as technology in the field advances. This is a setting where the number of time points may be small, but the dimension of the data is large (potentially of the order of 10s of 1000s). Sparse methods have been successful in modeling genetic data [Carvalho et al., 2008, Knowles and Ghahramani, 2011],

Table 2: Results of gray-scale image sequence denoising and interpolation (PSNR) for BP, dHBP, DRIFT and MBP, using patch size 8×8, varying the ratio of observed pixels. Image pixels have Gaussian white noise (standard deviation 15).

Ratio		Horse	Mountains	Rabbit	Hand
20%	BP	25.32	27.68	24.46	30.22
	dHBP	26.21	28.91	25.43	31.58
	DRIFT	26.04	28.23	24.85	30.74
	MBP	**26.81**	**29.09**	**25.61**	**31.62**
30%	BP	26.58	28.85	25.52	31.43
	dHBP	27.09	29.78	26.03	**32.56**
	DRIFT	27.02	29.34	25.94	31.96
	MBP	**27.47**	**29.94**	**26.54**	32.51
50%	BP	27.45	29.95	26.38	32.64
	dHBP	28.09	30.55	26.84	**33.28**
	DRIFT	28.14	30.50	26.82	33.19
	MBP	**28.38**	**30.66**	**26.92**	33.22

and subsequently, using the MBP to model time-evolving gene data seems an appropriate extension. Comparisons between the MBP, DRIFT, BP and dIBP models are made. We consider 2 datasets, discarding 2000 samples to allow the Markov chains to burn-in and using the following 500 samples for prediction, and using $K = 500$ features.

Yeast cell cycle Spellman et al. [1998] measured the genome-wide mRNA levels for 6108 genes during 2 cell cycles over $T = 17$ time points. We consider $N = 2$ strands, cdc15 and cdc28 and randomly pick $D = 1000$ genes randomly from the ones which have no missing data.

Transcriptome alterations in mice This dataset, collected by Piechota et al. [2010], consists of readings from 46632 proteins collected from mice brains under the infuence of 1 of $N = 8$ types of drugs over $T = 4$ time points. We select a random subset of $D = 1000$ proteins for our experiments.

In each of the experiments we hold out 15% of the data points uniformly at random for prediction. Results of the experiments on the 2 gene data sets are summarized in Table 3. The dIBP performs best on the yeast cell cycle data set, as the Gaussian process draws controlling feature usage over time are best able to pick up the periodic nature of the cell cycle sequence data. However, the MBP is not far off the accuracy of the dIBP here. The MBP outperforms other methods on the mice transcriptome data set, which has fewer time points. The DP in the MBP inference procedure used an average of 82 transition functions after burn-in. Note that the total number of transitions is $K \times (T - 1) = 1500$, and hence 82 clusters is perfectly reasonable. In the yeast cell cycle task, we have $K \times (T - 1) = 8000$ feature transitions, which is very large, and illustrates to some extent the requirement for highly flexible mixture model of transition functions, such as the Dirichlet process mixture we employ.

Table 3: Mean-squared error results of prediction on gene datasets using MBP, DRIFT, BP and dIBP.

	MBP	DRIFT	BP	dIBP
Yeast	0.92	1.06	1.63	**0.88**
Mice	**1.12**	1.42	1.67	1.28

6 CONCLUSIONS

In this work, we construct a Markov chain of beta processes for use as a model for learning time evolving sparse latent representations. Particularly we exploit the property that the beta process is a type of Poisson process. This enables us to invoke well known operations, which when applied to a Poisson process draw keep the object marginally Poisson process distributed. Having marginal beta process objects crucially allows us to develop a simple and fast mixing Gibbs sampler. To illustrate the power of our model and inference scheme, we considered image denoising and inpainting tasks and gene expression data, showing superior performance over other beta process related models. The high flexibility of our model leads to a potential drawback when modelling long time series. Inference may become prohibitively slow as the time series are made longer, and to compromise, it may be necessary to replace the DP based transition function with a small fixed mixture of Gaussians.

Most machine learning problems involve constructing an appropriate model and then developing an inference scheme to train the model and use it for prediction. Whilst Bayesian nonparametrics make it simple to write down a model, the inference often requires a complicated and slow procedure which significantly can reduce its applicability to real problems. We hope is that our exploitation of Poisson process preserving operations and conjugacy encourages further investigation in how one can develop flexible nonparametric Bayesian models which are amenable to elegant inference techniques, by exploiting interesting and well grounded mathematical concepts.

References

D. J. Bartholomew. The Foundations of Factor Analysis. *Biometrika*, 1987.

C. M. Carvalho, J. Chang, J. E. Lucas, J. R. Nevins, Q. Wang, and M. West. High Dimensional Sparse Factor Modeling: Applications in Gene Expression Genomins. *Journal of the American Statistical Association*, 2008.

C. Chen, N. Ding, and W. Buntine. Dependent Hierarchical Normalized Random Measures for Dynamic Topic Modeling. *Proceedings of the 29th International Conference on Machine Learning*, 2012.

F. Doshi-Velez and Z. Ghahramani. Accelerated Sampling for the Indian Buffet Process. *Proceedings of the 26th International Conference on Machine Learning*, 2009.

J. Foulds, C. DuBois, A. U. Asuncion, C. T. Butts, and P. Smyth. A Dynamic Relational Infinite Feature Model for Longitudinal Social Networks. *Proceedings of the 14th Conference on Artificial Intelligence and Statistics*, 2011.

T. Griffiths and Z. Ghahramani. The Indian Buffet Process: An introduction and review. *Journal of Machine Learning Research*, 12:1185–1224, 2011.

J. Kingman. *Poisson Processes*. Oxford University Press, 1993.

D. Knowles and Z. Ghahramani. Nonparametric Bayesian Sparse Factor Models with Application to Gene Expression Modeling. *The Annals of Applied Statistics*, 2011.

D. Lin, E. Grimson, and J. Fisher. Construction of Dependent Dirichlet Processes based on Poisson Processes. *Advances in Neural Information Processing Systems*, 2010.

G. MacLachan and D. Peel. *Finite Mixture Models*. John Wiley & Sons, 2000.

P. Ochs, J. Malik, and T. Brox. Segmentation of moving objects by long term video analysis. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2014.

M. Piechota, M. Korostynski, W. Solecki, A. Gieryk, M. Slezak1, W. Bilecki, B. Ziolkowska, E. Kostrzewa, I. Cymerman, L. Swiech, J. Jaworski, and R. Przewlocki. The dissection of trancriptional modules regulated by various drugs of abuse in the mouse striatum. *Genome Biology, 11:R48*, 2010.

L. Porzi, S. R. Bulo, P. Valigi, O. Lanz, and E. Ricci. Learning Contours for Automatic Annotations of Mountains on a Smartphone. *ACM/IEEE Intl. Conference on Distributed Smart Cameras*, 2014.

L. Ren, Y. Wang, D. Dunson, and L. Carin. The Kernel Beta Process. *Advances in Neural Information Processing Systems*, 2011.

P. Spellman, G. Sherlock, W. Q. Zhang, V. R. Iyer, K. Anders, M. B. Eisen, P. O. Brown, D. Botstein, and B. Futcher. Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Mol. Biol. Cell*, 9:3273–3297, 1998.

Y. W. Teh, D. Görür, and Z. Ghahramani. Stick breaking construction for the Indian buffet process. *Proceedings of the 11th Conference on Artificial Intelligence and Statistics*, 2007.

R. Thibaux and M. I. Jordan. Hierarchical Beta Processes and the Indian Buffet Process. *Proceedings of the 11th Conference on Artificial Intelligence and Statistics*, 2007.

J. Van Gael, Y. W. Teh, and Z. Ghahramani. Infinite Factorial Hidden Markov Model. *Advances in Neural Information Processing Systems*, 2008.

C. C. Wang. Carnegie Mellon Image Database, 1997. URL `vasc.ri.cmu.edu/idb/`.

S. Williamson, P. Orbanz, and Z. Ghahramani. Dependent Indian Buffet Processes. *Proceedings of the 13th Conference on Artificial Intelligence and Statistics*, 2010.

M. Zhou, H. Chen, J. Paisley, L. Ren, G. Sapiro, and L. Carin. Non-Parametric Bayesian Dictionary Learning for Sparse Image Representations. *Advances in Neural Information Processing Systems*, 2009.

M. Zhou, H. Yang, G. Sapiro, D. Dunson, and L. Carin. Dependent Hierarchical Beta Process for Image Interpolation and Denoising. *Proceedings of the 14th Conference on Artificial Intelligence and Statistics*, 2011.