Variational Russian Roulette for Deep Bayesian Nonparametrics

Kai Xu [1] **Akash Srivastava** [1 2] **Charles Sutton** [1 3 4]

Abstract

Bayesian nonparametric models provide a principled way to automatically adapt the complexity of a model to the amount of the data available, but computation in such models is difficult. Amortized variational approximations are appealing because of their computational efficiency, but current methods rely on a fixed finite truncation of the infinite model. This truncation level can be difficult to set, and also interacts poorly with amortized methods due to the over-pruning problem. Instead, we propose a new variational approximation, based on a method from statistical physics called Russian roulette sampling. This allows the variational distribution to adapt its complexity during inference, without relying on a fixed truncation level, and while still obtaining an unbiased estimate of the gradient of the original variational objective. We demonstrate this method on infinite sized variational auto-encoders using a Beta-Bernoulli (Indian buffet process) prior.

1. Introduction

A major challenge in unsupervised learning is to infer the complexity of the latent structure, such as the number of clusters or the size of a continuous representation, that is necessary to describe a data set. A principled method from statistics to choose the complexity of a model is provided by *Bayesian nonparametric* methods (Walker et al., 1999; Müller & Quintana, 2004; Orbanz & Teh, 2010; Gershman & Blei, 2012). Nonparametric methods allow for the size of the inferred model to automatically adapt to the amount of data, so that simpler models are preferred for smaller data sets, and more complex models are preferred for larger data sets. For example, a latent feature model with an Indian buffet process (IBP) prior is a representation learning method that infers a latent binary vector for each data point, where the number of binary features is chosen adaptively based on the data. Within machine learning, Bayesian nonparametric methods have been applied within models as diverse as clustering (Antoniak, 1974; Görür & Rasmussen, 2010; Teh et al., 2005), topic modeling (Teh et al., 2006), and infinite deep neural networks (Adams et al., 2010).

However, inference in Bayesian nonparametric models can be computationally challenging. Amortized variational methods (Kingma & Welling, 2013; Rezende et al., 2014; Ranganath et al., 2014a; Mnih & Gregor, 2014) are an appealing option, because they exploit the smoothing properties of deep neural networks to accelerate inference. For Bayesian nonparametric models, however, amortized inference is challenging because the dimensionality of the latent space is not fixed. Previous methods for variational inference in such models rely on a truncated approximation, which places an upper bound on the size of the latent space under the approximate posterior (Blei & Jordan, 2004; Doshi-Velez et al., 2009). Similarly, recent work on amortized inference in Bayesian non-parametrics relies on truncation, albeit sometimes within an outer loop that searches over the truncation size (Miao et al., 2017; Nalisnick & Smyth, 2017; Chatzis, 2014; Singh et al., 2017).

However, the truncated approximation has several drawbacks. If the truncation level is chosen too small, the accuracy of the approximation degrades, whereas if the truncation level is chosen too large, then inference will be slow, removing one of the main advantages of a variational approximation. Perhaps more fundamentally, the truncation level can interact poorly with amortized inference, because of a well-known issue called component collapsing (Dinh & Dumoulin, 2016; van den Oord et al., 2017), which is also called over-pruning (Burda et al., 2015; Yeung et al., 2017). This refers to the problem when the inferred latent representation includes components whose conditional distribution given a test data point tends to remains very similar to the prior. These components are useless as they do not help explain the data.

In this work, we overcome these limitations using a new dynamic variational approximation, which we call *Roulette-based Amortized Variational Expectations (*RAVE*)*. The goal

[1]School of Informatics, University of Edinburgh, Edinburgh, United Kingdom [2]MIT-IBM Watson AI Lab, Cambridge, MA, United States [3]Google AI, Mountain View, CA, United States [4]Alan Turing Institute, London, United Kingdom. Correspondence to: Kai Xu <kai.xu@ed.ac.uk>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

of RAVE is to allow the approximate variational posterior to adapt its size over the course of the optimization. But this causes the problem that expectations for the evidence lower-bound (ELBO) then require computing an infinite summation which cannot be tackled using the reparameterization trick (Williams, 1992; Kingma & Welling, 2013; Rezende et al., 2014). To surmount this problem, we use a different Monte Carlo approximation, namely, the Russian roulette sampling method from statistical physics (Lux & Koblinger, 1991; Carter & Cashwell, 1975; Lyne et al., 2015), which allows us to approximate this sum by a sample from a Markov chain. This leads to an unbiased estimate of the gradient of the ELBO which can be maximized using stochastic gradient ascent.

We demonstrate RAVE on an *infinite variational autoencoder* (Chatzis, 2014), which assigns each data point to a continuous representation whose size is automatically inferred from the data. The prior on the number of components is given by an Indian buffet process model. We show empirically that previous amortized variational methods suffer from the component collapsing problem and tend to infer useless components, whereas RAVE infers a model with many fewer components, while inferring an overall model of similar explanatory power.

2. Background

We review material from nonparametric Bayesian statistics and variational inference. We also introduce the basic form of the Russian roulette estimate that we will use.

2.1. Indian buffet process

An important problem in representation learning is to learn to represent each data item by a binary vector whose elements indicate latent *features* underlying the data. If the necessary number of features is unknown, we can take a Bayesian approach, and place a prior distribution over all possible latent feature matrices. One such prior distribution is the Indian buffet process (IBP), denoted $\mathbf{Z} \sim \text{IBP}(\alpha)$, which is a probability distribution over sparse binary matrices with a finite number of rows and an unbounded number of columns (Griffiths & Ghahramani, 2011). We define the IBP using the stick-breaking construction (SBC) of Teh et al. (2007), which defines a distribution over $\mathbf{Z}$ as

$$\nu_k \sim \text{Beta}(\alpha, 1),\ \pi_k = \prod_{j=1}^{k} \nu_j,\ z_{nk} \sim \text{Bern}(\pi_k), \quad (1)$$

for $n \in 1 \dots N$ and $k \in 1, 2, \dots, \infty$. Intuitively, we start with a stick of length 1 and break it at random to obtain a new stick of length ν_1. We then break this new stick at another random proportion ν_2 to obtain a new stick of length $\nu_1\nu_2$. We write $\mathbf{Z} \sim \text{SBC}(\alpha, N, K)$ to indicate the distribution of the binary matrix $\mathbf{Z}$ if this process is stopped after K columns, and N is the number of data points which should be clear in context. We denote by $\text{IBP}(\alpha)$ the stochastic process that results from $\text{SBC}(\alpha, N, K)$ as $K \to \infty$.

2.2. Latent Feature Models

The IBP can be used as a prior over sparse latent representation $\mathbf{Z} = [\mathbf{z}_1 \dots \mathbf{z}_N]$ of data $\mathbf{X} = [\mathbf{x}_1 \dots \mathbf{x}_N] \in \mathbb{R}^{N \times D}$. Using this prior, we model the data as

$$\mathbf{Z} \sim \text{IBP}(\alpha),\ \mathbf{A} \sim \mathcal{N}(0, \sigma_A^2 I),\ \mathbf{X} \sim p_\theta(\mathbf{X} \mid \mathbf{Z}, \mathbf{A}), \quad (2)$$

for $n \in 1 \dots N$. A popular model arises when $\mathbf{A}$ is a matrix with D columns and infinitely many rows and when $p_\theta(\mathbf{X} \mid \mathbf{Z}, \mathbf{A}) = \mathcal{N}(\mathbf{X} \mid \mathbf{Z}\mathbf{A}, \sigma_{\mathbf{X}}^2 I)$. This is the well-studied linear Gaussian model.[1]

Alternatively, we can use a deep network to parameterize $p_\theta(\mathbf{X} \mid \mathbf{Z}, \mathbf{A})$. Specifically, choose $p_\theta(\mathbf{X} \mid \mathbf{Z}, \mathbf{A}) = \mathcal{N}(\mathbf{X} \mid \mu_\theta(\mathbf{Z} \odot \mathbf{A}), \sigma_\theta(\mathbf{Z} \odot \mathbf{A}))$ or $p_\theta(\mathbf{X} \mid \mathbf{Z}, \mathbf{A}) = \text{Bern}(\mathbf{X} \mid p_\theta(\mathbf{Z} \odot \mathbf{A}))$ where $\odot$ is the Hadamard product (Chatzis, 2014; Singh et al., 2017), μ_θ, σ_θ, and p_θ are multi-layer neural networks with parameter θ. We refer to these three neural networks together as a deep decoder. This leads to the infinite variational autoencoder, which we describe next.

2.3. Infinite Variational Autoencoders

The *infinite variational autoencoder* (infinite VAE) arises when apply variational inference to the deep latent feature model of the previous section. In that model, the posterior distribution over the latent variables $P(\mathbf{Z}, \mathbf{A}, \boldsymbol{\nu} \mid \mathbf{X})$ is intractable, so one popular approximation is variational inference. Singh et al. (2017) present a structured variational inference method for this model, based on the method of Hoffman & Blei (2015), which performs better than the more common mean-field approximation, as it introduces dependencies in the approximate posterior distribution, between $\mathbf{Z}$ and $\boldsymbol{\nu}$. The variational posterior from Singh et al. (2017) has the form

$$q(\mathbf{Z}, \mathbf{A}, \boldsymbol{\nu}_{(1:K)}) = q(\mathbf{A}) q(\boldsymbol{\nu}_{(1:K)}) \prod_{n=1}^{N} \prod_{k=1}^{K} q(z_{nk} \mid \boldsymbol{\nu}_{(1:K)}),$$

where K is the truncation level. Each component of q has parameters, called *variational parameters*, which are optimized to make $q(\mathbf{Z}, \mathbf{A}, \boldsymbol{\nu}_{(1:K)})$ as close as possible to the true posterior $p(\mathbf{Z}, \mathbf{A}, \boldsymbol{\nu}_{(1:K)} | \mathbf{X})$ as measured by KL-divergence. This is accomplished by optimizing a lower bound called the *evidence lower-bound (ELBO)*. Optimizing the ELBO requires sampling from a Monte Carlo estimate, which is designed to be differentiable with respect to

[1] Following Chatzis (2014); Singh et al. (2017), we omit the prior distribution $p(\mathbf{A})$ for this linear model when doing amortized inference, and instead optimize over $\mathbf{A}$.

the model parameters and the variational parameters. This can be made possible with the reparameterization trick.[2] Singh et al. (2017) employs reparameterizations of the Beta distribution (Nalisnick & Smyth, 2017) and the Bernoulli distribution (Jang et al., 2016; Maddison et al., 2016).

2.4. Russian roulette sampling

Russian roulette sampling (Lux & Koblinger, 1991; Carter & Cashwell, 1975; Lyne et al., 2015; Georgoulas et al., 2017) is a Monte Carlo technique for estimating very large sums. Suppose we want to compute an infinite sum

$$S = \sum_{k=1}^{\infty} T_k, \tag{3}$$

where we assume $S < \infty$. In Russian roulette sampling, we estimate S by truncating the sum after τ terms, where τ is random. As a simple example, let $S = T_1 + T_2$. We can approximate S by the estimate $\hat{S} = T_1$ with probability 0.5, and $\hat{S} = T_1 + 2T_2$ with probability 0.5. It is easy to see that $\mathbb{E}[\hat{S}] = S$. Applying this trick recursively yields the general Russian roulette estimate.

More generally, let $P(\tau) = (1 - \rho_{\tau+1}) \prod_{s=1}^{\tau} \rho_s$, be a distribution over the number of terms τ in the estimate. The probabilities $\rho_1, \rho_2, \ldots$ are parameters of the approximation. Define the truncated sum

$$\hat{S}_K = \sum_{k=1}^{K} \frac{T_k}{p_k}, \tag{4}$$

where $p_k = \prod_{j=1}^{k} \rho_j$ is the probability that $\tau > k$; dividing by this quantity has the effect of correcting for the fact that later terms are less likely to be included in the estimate. Then we define the Russian roulette estimate as $\hat{S} = S_\tau$ with probability $P(\tau)$. It can be shown that this estimate is unbiased for S, e.g. see the Appendix of Lyne et al. (2015).

3. Method

Now we introduce RAVE, an amortized variational inference method based on dynamic truncation. For concreteness, we describe RAVE in the context of a deep latent factor model with an IBP prior, but the method can be used more generally. First, we introduce a variational family in which the number of latent dimensions is random, governed by its own variational parameters (Section 3.1); this is essentially an infinite mixture of truncated variational distributions. Then, we present the ELBO over all the variational parameters, showing that it can be written as an infinite sum (Section 3.2). Then we show how Russian roulette sampling can be used to obtain an unbiased Monte Carlo estimate the gradient of this sum (Section 3.3). Finally, we put all of these ideas together into a stochastic gradient optimization algorithm that works on a finite representation of the infinite number of parameters (Section 3.4).

3.1. Infinite-sized Variational Family

We start by describing the variational family of approximate posterior distributions that we consider in RAVE. Unlike previous amortized variational methods, the dimensionality is not bounded a priori, but is controlled by continuous variational parameters. We define the variational family using the stick-breaking construction as

$$\begin{aligned} &\nu_k \sim \mathrm{Beta}(\alpha_k, \beta_k), \quad \pi_k = \prod_{j=1}^{k} \nu_j, \\ &K^* = k \quad \text{with probability } m_k = (1 - \rho_{k+1}) \prod_{i=1}^{k} \rho_i, \\ &z_{nk} \sim \mathrm{Bern}(f_\phi(\pi_k, \mathbf{x}_n) \cdot \delta\{k \le K^*\}) \end{aligned} \tag{5}$$

for $n \in 1 \ldots N$ and $k \in 1, 2, \ldots, \infty$. We denote a single variational distribution in this family as $q(\boldsymbol{\nu}, K^*, \mathbf{Z} | \alpha, \beta, \phi, \rho)$. In this equation, α_k, β_k, ρ_k, and ϕ are the variational parameters, and the neural network f_ϕ is an inference network that amortizes the approximation of the posterior distribution. One can view this variational distribution as a mixture of infinitely many truncated ones.

The only parameters we amortize are those for the variational distribution of $\mathbf{Z}$. The parameters of the inference network are an infinite sequence of vectors $\phi = (\phi_0, \phi_1 \ldots)$ with $\phi_k \in \mathbb{R}^{D+1}$. Then our inference network is

$$f_\phi(\pi_k, \mathbf{x}_n) = \sigma(\mathrm{logit}(\pi_k) + \phi_k^\top [\mathbf{x}_n, 1]), \tag{6}$$

where σ is the sigmoid function.[3] Note that f_ϕ outputs a scalar, essentially the approximate posterior distribution for a single hidden unit.

For the Beta and Bernoulli distributions, we use the Monte Carlo "reparameterization trick" during training for gradient. We use the Kumaraswamy reparameterization for the Beta distribution (Nalisnick & Smyth, 2017) and the Concrete reparameterization for the Bernoulli distribution (Jang et al., 2016; Maddison et al., 2016), following Singh et al. (2017).

We have defined this variational family to have an infinite number of parameters: all of the variational parameters α, β, ρ, and ϕ are infinite sequences. In order to optimize these parameters practically, observe for any integer k, the conditional distribution $q(\boldsymbol{\nu}, \mathbf{Z} | K^* = k, \alpha, \beta, \phi, \rho)$ depends

[2] Alternatives include REINFORCE (Williams, 1992), generalized reparameterization gradient (Ruiz et al., 2016), and automatic differentiation variational inference (Kucukelbir et al., 2017).

[3] The notation $[\mathbf{x}_n, 1]$ represents vector concatenation, so that the dot product implicitly incorporates the bias term.

only on the first k variational parameters. It is this property that we use to approximate the ELBO in the next section.

This completes the description of the approximate posterior distribution $q(\boldsymbol{\nu}, \mathbf{Z}, K^*)$ for the parameters of the IBP prior. We also need a variational distribution $q(\mathbf{A})$ over the parameters $\mathbf{A}$ of the observation model, which we choose to be Gaussian. It is only amortised when using deep decoders.

3.2. Approximating the ELBO Gradient

To find the best approximate posterior distribution, we maximize the ELBO

$$\mathcal{L} = -\operatorname{KL}\left[q(\boldsymbol{\nu}) \,\|\, p(\boldsymbol{\nu})\right] - \operatorname{KL}\left[q(\mathbf{A}) \,\|\, p(\mathbf{A})\right] + \mathbb{E}_{q_{\boldsymbol{\nu}}}\left[\mathbb{E}_{q_{\mathbf{Z}}}\left[\log \frac{p(\mathbf{X} \mid \mathbf{Z}, \mathbf{A}) p(\mathbf{Z} \mid \boldsymbol{\nu})}{q(\mathbf{Z} \mid \boldsymbol{\nu})}\right]\right], \tag{7}$$

which can be derived from the KL-divergence between the marginal distribution $q(\boldsymbol{\nu}, \mathbf{Z}, \mathbf{A})$[4] and the true posterior $p(\boldsymbol{\nu}, \mathbf{Z}, \mathbf{A}|\mathbf{X})$. Optimizing this function is challenging for several reasons. First, there are an infinite number of variational parameters, so we need to obtain a finite representation. Second, the distribution $q(\mathbf{Z} \mid \boldsymbol{\nu})$ in the third term is not easy to compute, because it is a marginal distribution $q(\mathbf{Z} \mid \boldsymbol{\nu}) = \sum_{k=0}^{\infty} q(\mathbf{Z}, K^* = k \mid \boldsymbol{\nu})$. Finally, optimizing with respect to ρ is particularly challenging, intuitively because ρ determines the stochastic control flow of q; see Algorithm 1.

Computing the first two terms of Equation 7 is straightforward (see Appendix A). For the third term, we re-write this expectation using the tower property

$$\mathbb{E}_{q_{\mathbf{Z}}}\left[\log \frac{p(\mathbf{X} \mid \mathbf{Z}, \mathbf{A}) p(\mathbf{Z} \mid \boldsymbol{\nu})}{q(\mathbf{Z} \mid \boldsymbol{\nu})}\right] = \sum_{k=0}^{\infty} m_k \mathbb{E}_{q_{\mathbf{Z}}}\left[\log \frac{p(\mathbf{X} \mid \mathbf{Z}, \mathbf{A}) p(\mathbf{Z} \mid \boldsymbol{\nu})}{q(\mathbf{Z}|\boldsymbol{\nu})} \mid K^* = k\right]. \tag{8}$$

Because the expectation now conditions on $K^* = k$, we know that Z will have at most k nonzero columns, and so the numerator within the expectation is now computable. The denominator $q(\mathbf{Z}|\boldsymbol{\nu})$ is still challenging, because it marginalizes out K^*, and still contains an infinite sum. We can obtain a slightly looser variational lower bound using the inequality

$$q(\mathbf{Z}|\boldsymbol{\nu}) = \sum_{j=1}^{\infty} m_j q(\mathbf{Z}|K^* = j, \boldsymbol{\nu}) \leq q(\mathbf{Z}|K^* = K^\dagger, \boldsymbol{\nu}), \tag{9}$$

where $K^\dagger := \max\{k \mid \exists n, z_{nk} \neq 0\}$, the maximum column index for which that column of $\mathbf{Z}$ is not all 0s. The inequality comes from two facts. First, $q(\mathbf{Z}|K^* = j, \boldsymbol{\nu}) = 0$ for $j < K^\dagger$ because it is impossible to generate more than j features if the process is truncated at j. Second, $q(\mathbf{Z}|K^* = j, \boldsymbol{\nu})$ is a monotonically decreasing function for $j \geq K^\dagger$ because observing more columns of zeros will only decrease the probability under the Bernoulli distribution. A more detailed proof is provided in the Appendix B.

Combining Equations 7–9, we obtain the training objective

$$\tilde{\mathcal{L}} = \sum_{k=0}^{\infty} m_k \tilde{\mathcal{L}}^k, \tag{10}$$

where

$$\tilde{\mathcal{L}}^k = -\operatorname{KL}\left[q(\boldsymbol{\nu}) \,\|\, p(\boldsymbol{\nu})\right] - \operatorname{KL}\left[q(\mathbf{A}) \,\|\, p(\mathbf{A})\right] + \mathbb{E}_{q_{\boldsymbol{\nu}}}\left[\mathbb{E}_{q_{\mathbf{Z}}}\left[\log \frac{p(\mathbf{X} \mid \mathbf{Z}, \mathbf{A}) p(\mathbf{Z} \mid \boldsymbol{\nu})}{q(\mathbf{Z}|K^* = K^\dagger, \boldsymbol{\nu})} \mid K^* = k\right]\right]. \tag{11}$$

Note that we also move the KL terms for $\boldsymbol{\nu}$ and $\mathbf{A}$ inside the infinite summation. This is valid as the infinite summation is an expectation. This expectation cannot be computed exactly, but we will present a method for approximating it in the next section.

When RAVE is used with the IBP, we will refer to the overall method as RRS-IBP, where RR stands for Russian roulette and S stands for either Structured or Sampling, at the reader's option.

Interpretation as random truncation. We give another interpretation of the ELBO in Equation 10. During training, $\tilde{\mathcal{L}}^k$ is exactly the ELBO for the truncated variational method with truncation level k (Singh et al., 2017). Therefore, the lower bound $\tilde{\mathcal{L}}$ that we use can be interpreted as the expectation of the truncated ELBO, where the expectation is taken over our variational distribution $q(K^* = k) = m_k$ over the truncation level.

3.3. Russian roulette estimation of the ELBO gradient

Finally, we describe how we optimize the ELBO $\tilde{\mathcal{L}}$. To simplify the presentation, we introduce the notation $\psi_k = (\alpha_k, \beta_k, \phi_k, \theta_k)$, the vector of the variational and model parameters for component k, except for ρ_k, and we define the matrix $\psi_{1:k} = (\psi_1 \dots \psi_k)$. Then, $\tilde{\mathcal{L}}$ has the form

$$\tilde{\mathcal{L}} = \sum_{k=1}^{\infty} m_k T_k(\psi_{1:k}), \tag{12}$$

where $m_k = (1 - \rho_{k+1}) \prod_{i=1}^{k} \rho_i$ depends only on $\rho_{1:k+1}$, and T_k depends only on the other parameters $\psi_{1:k}$. This can be optimized by stochastic gradient ascent if we can obtain an unbiased estimate of its gradient.

First, the gradient with respect to ψ_k is

$$\partial_{\psi_k} := \frac{\partial \tilde{\mathcal{L}}}{\partial \psi_k} = \sum_{i=k}^{\infty} m_i \frac{\partial T_i}{\partial \psi_k}. \tag{13}$$

[4] We have omitted the dependence of q on the variational parameters for brevity.

We assume each $\frac{\partial T_i}{\partial \psi_k}$ can be computed by standard automatic differentiation techniques. To estimate this, we use a Russian roulette estimate $\hat{\partial}^{\text{RR}}_{\psi}$ with probabilities m_t. More specifically, we sample τ with probability $P(\tau = t) = m_t$, and then return the estimate $\hat{\partial}^{\text{RR}}_{\psi_k} = \hat{\partial}^{\tau}_{\psi_k}$, where

$$\hat{\partial}^{\tau}_{\psi_k} = \sum_{i=k}^{\tau} \frac{m_i}{\left(\prod_{j=1}^{i} \rho_j\right)} \frac{\partial T_i}{\partial \psi_k} = \sum_{i=k}^{\tau} (1 - \rho_{i+1}) \frac{\partial T_i}{\partial \psi_k}. \quad (14)$$

To derive the derivatives for ρ_k, first the chain rule yields

$$\partial_{\rho_k} := \frac{\partial \tilde{\mathcal{L}}}{\partial \rho_k} = \sum_{i=1}^{\infty} \frac{\partial \tilde{\mathcal{L}}}{\partial m_i} \frac{\partial m_i}{\partial \rho_k} = \sum_{i=1}^{\infty} T_i \frac{\partial m_i}{\partial \rho_k}, \quad (15)$$

where

$$\frac{\partial m_i}{\partial \rho_k} = \begin{cases} 0 & i < k-1 \\ -\frac{m_i}{1-\rho_k} & i = k-1 \\ \frac{m_i}{\rho_k} & i > k-1 \end{cases}. \quad (16)$$

This yields

$$\partial_{\rho_k} = \sum_{i=k-1}^{\infty} m_i w_i T_i, \quad w_i = \begin{cases} \frac{1}{\rho_k - 1} & i = k-1 \\ \frac{1}{\rho_k} & i > k-1 \end{cases}. \quad (17)$$

Finally, the Russian roulette estimate $\hat{\partial}^{\text{RR}}_{\rho_k}$ is

$$\hat{\partial}^{\tau}_{\rho_k} = \sum_{i=k-1}^{\tau} \frac{m_i w_i T_i}{\left(\prod_{j=1}^{i} \rho_j\right)} = \sum_{i=k-1}^{\tau} (1 - \rho_{i+1}) w_i T_i \quad (18)$$

with probability $P(\tau = t) = m_t$.

Now, each T_i is still difficult to compute, because it contains the expectation

$$\mathbb{E}_{q_{\boldsymbol{\nu}}} \left[\mathbb{E}_{q_{\mathbf{Z}}} \left[\log \frac{p(\mathbf{X} \mid \mathbf{Z}, \mathbf{A}) p(\mathbf{Z} \mid \boldsymbol{\nu})}{q(\mathbf{Z} | K^* = K^\dagger, \boldsymbol{\nu})} \mid K^* = i \right] \right].$$

We obtain a Monte Carlo approximation of both of these expectations using the standard reparameterization tricks.

In general, the Russian roulette estimation can have high variance, but what makes it so nice for variational inference is that because we are using a stick breaking construction, the earlier terms in the summation are the most important to include in the estimate.[5]

3.4. Stochastic gradient algorithm

Now that we have Monte Carlo estimates of the necessary derivatives, we can define the stochastic gradient algorithm.

[5] It is also possible to estimate (13) and (16) via naive MC estimation as both of them can be written as an expectation under m. However, this estimate is of high variance which cannot be easily overcame by using more samples. We empirically illustrate this in Appendix C with comparison to our Russian roulette estimates.

Algorithm 1 Sampling the truncation level τ during variational optimization, with lazy parameter initialization.

$t \leftarrow 0$
loop
  $t \leftarrow t + 1$
  if $t > L$ **then**
    $\rho_{t+1} \leftarrow 0.5$, $\alpha_t \leftarrow \alpha$, $\beta_t \leftarrow 1.0$, $\phi_t \leftarrow \epsilon_0$, $\theta_t \leftarrow \epsilon_0$
    $L \leftarrow L + 1$
  return $\tau = t$ with probability $1 - \rho_{t+1}$
$\{\alpha$ is the IBP parameter and $\epsilon_0 \sim \mathcal{N}(0, I)\}$

The key point is how we operate with only a finite representation of the variational parameters; essentially, we lazily instantiate only the finite subset of variational parameters that receive stochastic gradient updates. More specifically, at every point in the optimization algorithm, we maintain a finite representation of the variational parameters $\psi = (\psi_0 \dots \psi_L)$ and $\rho = (\rho_0 \dots \rho_L)$. These matrices will lazily grow in size as needed, that is, L will grow over the course of the optimization. At the beginning of the algorithm $L = 0$.

Each iteration of stochastic gradient ascent computes new parameters (ψ', ρ') from the current values (ψ, ρ). To do this, first we sample τ from the distribution $P(\tau)$ defined in the previous section. Importantly, it can happen that $\tau > L$, which means that the current iteration will introduce new parameters, which are initialized to default values. To make this clear, see Algorithm 1.

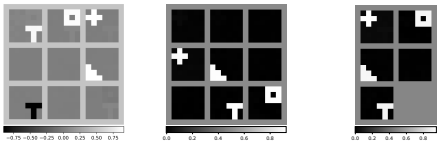
Given a value of τ, we make the gradient updates

$$\psi'_k \leftarrow \psi_k + \epsilon_0 \hat{\partial}^{\tau}_{\psi_k}, \quad k \in 1, 2, \dots \tau$$
$$\rho'_k \leftarrow \rho_k + \epsilon_1 \hat{\partial}^{\tau}_{\rho_k}, \quad k \in 2, 3, \dots \tau + 1$$

where ϵ_0 and ϵ_1 are step sizes. We only need perform the updates for $k \in 1, 2, \dots \tau$ for ψ_k because $\hat{\partial}^{\tau}_{\psi_k} = 0$ if $k > \tau$, and similarly for $\hat{\partial}^{\tau}_{\rho_k}$. We enforce that $\rho_1 = 1$ to ensure $\tau > 0$. Note that the gradient steps for ρ are important to the method, because this determines the inferred number of features. Finally, in practice each gradient is an average over M independent roulette samples; Appendix D describes how to reuse computation over the samples.

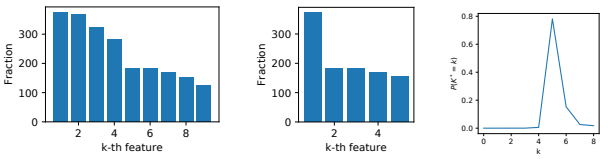
4. Experiments

To evaluate the RRS-IBP method, we compare it with two previous amortized inference approaches for IBP models: MF-IBP (Chatzis, 2014) and S-IBP (Singh et al., 2017). Following Singh et al. (2017), a multiplier greater than 1 is put on the KL term for $\boldsymbol{\nu}$ during training for structured variational methods, to encourage adhering to the IBP prior. See Appendix E for other training details. Source code of the implementation of our method is also available at



(a) MF-IBP (b) S-IBP (c) RRS-IBP

Figure 1. Features learned by VAEs ($\alpha = 4.0$).



(a) S-IBP (b) RRS-IBP (c) $P(K^* = k)$

Figure 2. Number activations per feature and truncation probability

`https://github.com/xukai92/RAVE.jl`.

4.1. Synthetic data

First, in order to check the correctness of the inference, we compare the different inference methods on synthetic dataset, for which the true data distribution and number of components are known. This data set, proposed by Griffiths & Ghahramani (2011), contains 6×6 gray-scale images, each of which are generated as a linear combination of four black and white images with random weights, plus Gaussian noise $\mathcal{N}(0, 0.1^2)$. We sample 2,400 images for training and 400 held-out images for testing.[6]

All inference methods are applied to the linear-Gaussian IBP model (Section 2.1). We set $\alpha = 4$ so that the expected number of components is not the same as the true distribution. For MF-IBP and S-IBP, the truncation level is set to 9, greater than the number of true features in the data.

First, we are interested in whether the inference methods identify the correct number of features for this data. One way to measure this is by the expected number of features per images under the posterior distribution, which we define as $M = N^{-1} \sum_{n=1}^{N} \mathbb{E}_{q\mathbf{z}} \left[\sum_k z_{nk} | \mathbf{x}_n \right]$, where $\mathbf{x}_n$ are the test images. For the true generating process, $M = 2.275$. Because the training data set is large, and the model family contains the generating distribution, the variational approximations should identify a similar number of features. We find that $M = 5.647$ for MF-IBP and $M = 6.021$ for S-IBP, while $M = 3.464$ for RRS-IBP. In other words, both MF-IBP and S-IBP infer many more features than are present in the true distribution, while RRS-IBP infers a value that is closer to the true one.

It may be surprising that the methods are inferring such different values for the number of features, because all three approximate the same posterior distribution. To understand this better, we visualize the learned features in Figure 1, which is simply the matrix $\mathbf{A}$ in the linear decoder. As we can see, S-IBP and RRS-IBP recover four black-and-white image features, which indeed are the images that were used to generate the data. MF-IBP recovers these four features up to a scaling of the image intensities, but also introduces an unnecessary negative feature (lower left in Figure 1(a)). Additionally, both MF-IBP and SS-IBP infer several useless features, which we will call "dummy features".

Such dummy features do no harm if they are never activated, that is, if their corresponding column in $\mathbf{Z}$ is always zero. However, in Figure 2(a), we plot the activation frequencies i.e. the number of features activated from a single sample of $q(\mathbf{Z}|\boldsymbol{\nu})$, inferred by S-IBP, for the testing set. This figure shows that the some top features from S-IBP are almost always activated, and in fact some are the black features in Figure 1(b). In other words, *meaningful* features in S-IBP do not necessarily come in order, as the method can infer "dummy" features.[7] On the other hand, RRS-IBP mostly avoids this issue (Figure 2(b)), as it learns only one dummy feature and this feature is not always being activated. As we can see the four meaningful features for S-IBP and RRS-IBP actually has similar activation probabilities. We hypothesize that this activation of dummy feature phenomena comes from the fact that when training with a high truncation level, there are many local maxima.

The plot of the probability mass function of the stopping level in Figure 2(c) shows that after training, the variational posterior for the truncation level puts most of its mass on the right level of truncation. This plot is not available for the other methods because their truncation level is fixed.

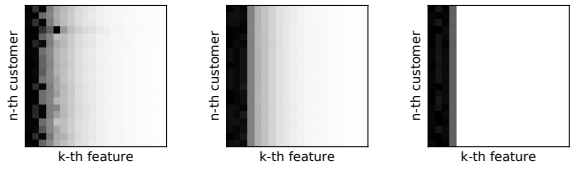
4.2. Image data

Now we compare the inference methods on benchmark image data sets, namely the MNIST dataset of handwritten digits (LeCun et al., 1998) and Fashion-MNIST, a dataset of images of products (Xiao et al., 2017). We use a deep decoder, where the prior is $p(\mathbf{A}) = \mathcal{N}(\mathbf{A}; 0, 1)$, and a Bernoulli observation distribution. Both the encoder and the decoder are two layer neural networks with 500 hidden units and ReLU activation function. We also report the performance of the same model and architecture on the synthetic data from the last section; for that simpler data, we use hidden layer of size 50 and a Gaussian observation distribution.

We measure both the quality of the inferred models and the number of inferred features. To measure model qual-

[6] Different from Griffiths & Ghahramani (2011) in which each feature is activated with probability 0.5, we sample an activation matrix $\mathbf{Z} \sim \text{SBC}(4.0, 2800, 4)$ (Equation 1) and generate the dataset as $\mathbf{X} = \mathbf{ZA}$, where $\mathbf{A}$ is the predefined feature matrix.

[7] This was also observed by Singh et al. (2017) on a different synthetic data set (see their appendix).



(a) MF-IBP (b) S-IBP (c) RRS-IBP

Figure 3. Posterior feature activation probability in grey-scale for VAEs with deep decoders on a subset of SYNTH; darker the higher. The plot for RRS-IBP is padded to the same number of features for comparison.

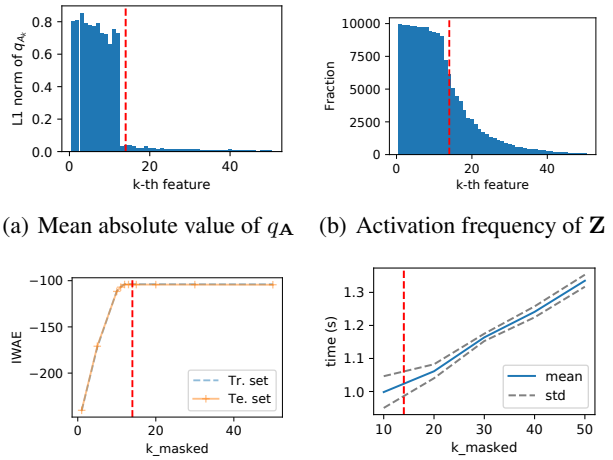
ity, we estimate the marginal probability of the test using the IWAE (Burda et al., 2015), which is a tighter lower bound than the ELBO. For S-IBP, we use the modification of the IWAE from Singh et al. (2017). For RRS-IBP, in order to compare against S-IBP using the same IWAE, we compute the mean of variational distribution of the truncation level, $\bar{m} = \mathbb{E}_{k\sim m_k}[k]$, and use $\lceil \bar{m} \rceil$ as the truncation level when computing the IWAE, where $\lceil \cdot \rceil$ is the ceiling function. To measure the number of inferred features, we report the number of features that have a non-negligible posterior activation probability, defined as $\tilde{K} = \sum_{k=1}^{K_{\max}} \max_{n\in\{1\dots N\}} \delta\{q(z_{nk} = 1) > \epsilon\}$ with $\epsilon = 0.01$. We will call $\tilde{K}$ the *number of activated features*. The results are shown in Table 1. MF-IBP has the worst IWAE over all datasets, consistent with the results of Singh et al. (2017). The model quality of S-IBP and RRS-IBP are mostly similar. However, looking at $\tilde{K}$, we see that RRS-IBP infers many fewer features than the other methods, even though the overall model is of similar or better quality.

We first verify our finding by visualizing the activation posterior probability, $q_{\mathbf{Z}}$, for models on the synthetic dataset. We choose synthetic dataset for visualization because the truncation level for other datasets are too high to populate readable plots. We visualize $q_{\mathbf{Z}}$ on a subset of the synthetic dataset by plotting the probabilities in grey-scale, which is shown in Figure 3. We confirm our previous finding on the visualization: MF-IBP spreads the activation probability after its dark region on the left widely. S-IBP has a more compact dark region and the spread is much less. RRS-IBP has the smallest dark region and spreading effect.

Another way to gain insight into whether the additional features inferred by S-IBP are indeed meaningless dummy features is to visualize the approximate posterior distributions for the S-IBP model. First, to understand $\mathbf{A}$, on the MNIST dataset[8] we plot the absolute value of the posterior mean of each feature k, $|\mathbb{E}_{q_{\mathbf{A}}}[\mathbf{A}_k]|$ ($\mathbf{A}_k$ for the k-th feature) in the order inferred by the model[9], averaged over the testing

[8]We conducted the same analysis on Fashion-MNIST and observed qualitatively similar results (see plots in Appendix F).

[9]Observe that because all the inference procedures are based on the stick breaking construction, this order is meaningful; the inferred features are automatically sorted by π_k.



(a) Mean absolute value of $q_{\mathbf{A}}$ (b) Activation frequency of $\mathbf{Z}$

(c) IWAE with first k features (d) Running time v.s. k

Figure 4. Effects of truncation level for S-IBP with a deep decoder. All plots are computed from the whole testing set of MNIST dataset. The vertical line in red is the corresponding mode of the truncation distribution inferred by RRS-IBP.

set. This is shown in Figure 4(a). We see that the first 13 features have a posterior distribution $q_{\mathbf{A}_k}$ with a mean that is not close to 0; for other features, the posterior mean is very close to 0. This is component collapsing (Dinh & Dumoulin, 2016; van den Oord et al., 2017), a known phenomenon in VAEs, which means that some hidden units collapse to the prior and hence convey no information to the decoder. The vertical line in all the plots in Figure 4 indicates the number of features inferred by RRS-IBP, the mode of $q(K^*)$. It is striking that RRS-IBP learns to truncate the model right after where S-IBP stops producing informative features.

Now, to understand $\mathbf{Z}$, we show the activation frequencies, i.e. the number of times each feature is used, for the testing set (Figure 4(b)). This confirms that many of the uninformative features are activated for a substantial fraction of data points. This is similar to what we have visually seen in Section 4.1 for the synthetic data.

As a final test of whether these uninformative features contribute to the quality of the inferred model, we report the test set IWAE of the S-IBP model if after training is complete, the model is truncated after k features (Figure 4(c)). Figure 4(c) confirms our statement that the dummy features conveys no information to the decoder because the IWAE reaches its maximum right after all non-collapsed features are used. It is clear that the additional features do not help improve the inference results, and is a waste of computation, shown in Figure 4(d). However, RRS-IBP avoids this problem due to its nature of automatic truncation and the inferred truncation level (vertical line in red) is just the optimal level of truncation under our post analysis. Additionally,

Table 1. IWAE for different VAEs under the IBP prior on various datasets.

METHOD	SYNTH ($\alpha = 4$, $K_{\text{MAX}} = 20$)			MNIST ($\alpha = 10$, $K_{\text{MAX}} = 50$)			FASHION-MNIST ($\alpha = 20$, $K_{\text{MAX}} = 50$)		
	TRAINING	TESTING	$\tilde{K}$	TRAINING	TESTING	$\tilde{K}$	TRAINING	TESTING	$\tilde{K}$
MF-IBP	43.91	43.28	20	-111.17	-111.45	50	-241.99	-244.15	50
S-IBP	42.99	42.47	20	-103.83	-104.28	50	-239.24	-241.52	50
RRS-IBP	45.73	44.93	4	-103.54	-104.54	14	-237.84	-240.34	9

compared with a fixed truncation chosen at 50, the inferred network adapted with size 14 has ~ 1.35 times speed-up.

We can understand why S-IBP activates dummy features by considering the effect of dummy features on the ELBO. The additional KL penalty for the dummy features is very small, as $q_{\mathbf{A}_k}$ for the dummy features collapses to the prior. Furthermore, the reconstruction term of the ELBO does not discourage dummy features too much because the dummy features mostly model noise and are ignored by the decoder. As a result, the ELBO does not provide a large penalty for including these features. As an aside, we found that optimization becomes difficult for MF-IBP and S-IBP with a large truncation. E.g. on SYNTH, we were unable to train these methods with a fixed truncation of 50. We hypothesize this is due to an increased number of local maxima introduced by the larger inference and generation networks.

5. Related Work

Variational inference for IBP latent feature models is proposed by Doshi-Velez et al. (2009) and relies on the stick-breaking construction from (Teh et al., 2007). Based on the construction, Teh et al. (2007) develops a slice sampler for IBP latent feature models. The sampler maintains a finite representation of SBC that can be used to compute the conditional probabilities required during the sampling. Our method is also based on the stick-breaking construction and in fact shares a similar spirit with the slice sampling: we maintain a finite representation that can be used to compute the unbiased gradient of ELBO during optimization.

We are aware of only a few papers that apply amortized inference to IBP models. Chatzis (2014) uses black box variational inference (BBVI, Ranganath et al. (2014b)) to train a VAE with an IBP prior using the mean-field variational approximation. Later Singh et al. (2017) propose a more accurate approximation by using the Kumaraswamy and Gumbel reparameterizations instead of BBVI, and by introducing a structured variational approximation instead of mean-field. We follow all of those innovations in this work. However, both of these papers rely on finite truncation in the variational distribution, which we avoid in this work. Another type of nonparametric variational autoencoder is proposed by Abbasnejad et al. (2017), who introduce an infinite mixture of finite-sized variational autoencoders.

In other Bayesian nonparametric models, some authors have reduced the impact of truncation by combining truncated variational inference with other optimization methods; however, these methods perform discrete search over the truncation level outside the VI optimization loop[10]. For example, Bryant & Sudderth (2012); Hughes et al. (2015) use split and merge steps to change the size of the variational model. Hu et al. (2015); Hughes & Sudderth (2013) also propose methods that adapt the truncation levels outside the VI optimization loop in the context of memoized variational inference for Dirichlet process models. Nalisnick & Smyth (2017) mention some initial experiments on making the level of stick-breaking adaptive by putting a threshold on the percentage of the remaining stick.

Within Bayesian statistics, Russian roulette sampling has been previously used within MCMC algorithms for doubly-intractable distributions (Lyne et al., 2015; Wei & Murray, 2016; Georgoulas et al., 2017). We are unaware of previous work applying Russian roulette sampling within variational methods. Indeed, to our knowledge, ours is the first variational inference method for Bayesian nonparametrics (whether amortized or the more traditional mean-field approach) that avoids a truncated variational approximation.

6. Conclusions and Discussions

In this work we propose RAVE, a novel inference method for BNP models that provides a probabilistic way to perform amortized (neural) variational inference without the need of explicitly truncating the maximum number of inferred posterior features as is common in previous work on variational inference for BNP. We clearly demonstrate that our inference method outperforms both structured and mean-filed amortized variational inference methods that resort to explicit truncation of the posterior features. In future work we aim to extend this method to other non-parametric models.

Acknowledgements

This work was funded by Edinburgh Huawei Research Lab, which is funded by Huawei Technologies Co. Ltd.

[10] In fact, discrete search for truncation levels based on various heuristics can be viewed as using a roulette probability that has mass on truncation levels around the mode in our method.

References

Abbasnejad, M. E., Dick, A., and van den Hengel, A. Infinite variational autoencoder for semi-supervised learning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 781–790. IEEE, 2017.

Adams, R., Wallach, H., and Ghahramani, Z. Learning the structure of deep sparse graphical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 1–8, 2010.

Antoniak, C. E. Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. *The annals of statistics*, pp. 1152–1174, 1974.

Blei, D. M. and Jordan, M. I. Variational methods for the Dirichlet process. In *International Conference in Machine Learning (ICML)*, 2004. URL `http://doi.acm.org/10.1145/1015330.1015439`.

Bryant, M. and Sudderth, E. B. Truly nonparametric online variational inference for hierarchical Dirichlet processes. In Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems 25*, pp. 2699–2707, 2012.

Burda, Y., Grosse, R., and Salakhutdinov, R. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

Carter, L. L. and Cashwell, E. D. Particle-transport simulation with the Monte Carlo method. Technical report, Los Alamos Scientific Lab., 1975.

Chatzis, S. P. Indian buffet process deep generative models. *arXiv preprint arXiv:1402.3427*, 2014.

Dinh, L. and Dumoulin, V. Training neural Bayesian nets, 2016.

Doshi-Velez, F., Miller, K., Van Gael, J., and Teh, Y. W. Variational inference for the Indian buffet process. In *Artificial Intelligence and Statistics*, pp. 137–144, 2009.

Georgoulas, A., Hillston, J., and Sanguinetti, G. Unbiased Bayesian inference for population markov jump processes via random truncations. *Statistics and computing*, 27(4): 991–1002, 2017.

Gershman, S. and Blei, D. A tutorial on Bayesian nonparametric models. *Journal of Mathematical Psychology*, 56: 1–12, 2012.

Görür, D. and Rasmussen, C. E. Dirichlet process Gaussian mixture models: Choice of the base distribution. *Journal of Computer Science and Technology*, 25(4):653–664, 2010.

Griffiths, T. L. and Ghahramani, Z. The Indian buffet process: An introduction and review. 12(Apr):1185–1224, 2011.

Hoffman, M. and Blei, D. Stochastic structured variational inference. In *Artificial Intelligence and Statistics*, pp. 361–369, 2015.

Hu, Z., Qirong, H., Dubey, A., and Xing, E. Large-scale distributed dependent nonparametric trees. In *International Conference on Machine Learning*, pp. 1651–1659, 2015.

Hughes, M., Kim, D. I., and Sudderth, E. Reliable and scalable variational inference for the hierarchical Dirichlet process. In *Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 38, pp. 370–378, 2015.

Hughes, M. C. and Sudderth, E. Memoized online variational inference for Dirichlet process mixture models. In *Advances in Neural Information Processing Systems*, pp. 1133–1141, 2013.

Jang, E., Gu, S., and Poole, B. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kingma, D. P. and Welling, M. Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kucukelbir, A., Tran, D., Ranganath, R., Gelman, A., and Blei, D. M. Automatic differentiation variational inference. *The Journal of Machine Learning Research*, 18(1): 430–474, 2017.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Lux, I. and Koblinger, L. *Monte Carlo Particle Transport Methods: Neutron and Photon Calculations*. CRC press, 1991.

Lyne, A.-M., Girolami, M., Atchadé, Y., Strathmann, H., Simpson, D., et al. On Russian roulette estimates for Bayesian inference with doubly-intractable likelihoods. *Statistical science*, 30(4):443–467, 2015.

Maddison, C. J., Mnih, A., and Teh, Y. W. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016.

Miao, Y., Grefenstette, E., and Blunsom, P. Discovering discrete latent topics with neural variational inference. In *International Conference in Machine Learning (ICML)*, 2017.

Mnih, A. and Gregor, K. Neural variational inference and learning in belief networks. In *International Conference on Machine Learning*, Proceedings of Machine Learning Research, pp. 1791–1799. PMLR, 2014.

Müller, P. and Quintana, F. A. Nonparametric Bayesian data analysis. *Statistical science*, pp. 95–110, 2004.

Nalisnick, E. and Smyth, P. Stick-breaking variational autoencoders. In *International Conference on Learning Representations (ICLR)*, 2017.

Orbanz, P. and Teh, Y. W. Bayesian nonparametric models. In *Encyclopedia of Machine Learning*. Springer, 2010.

Pharr, M., Jakob, W., and Humphreys, G. *Physically based rendering: From theory to implementation*. Morgan Kaufmann, 2016.

Ranganath, R., Gerrish, S., and Blei, D. Black box variational inference. In *Artificial Intelligence and Statistics*, 2014a.

Ranganath, R., Gerrish, S., and Blei, D. Black box variational inference. In *Artificial Intelligence and Statistics*, pp. 814–822, 2014b.

Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, 2014.

Robbins, H. and Monro, S. A stochastic approximation method. In *Herbert Robbins Selected Papers*, pp. 102–109. Springer, 1985.

Ruiz, F. R., AUEB, M. T. R., and Blei, D. The generalized reparameterization gradient. In *Advances in neural information processing systems*, pp. 460–468, 2016.

Singh, R., Ling, J., and Doshi-Velez, F. Structured variational autoencoders for the Beta-Bernoulli process. 2017.

Teh, Y. W., Jordan, M. I., Beal, M. J., and Blei, D. M. Sharing clusters among related groups: Hierarchical Dirichlet processes. In *Advances in neural information processing systems*, pp. 1385–1392, 2005.

Teh, Y. W., Jordan, M. I., Beal, M. J., and Blei, D. M. Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

Teh, Y. W., Grür, D., and Ghahramani, Z. Stick-breaking construction for the Indian buffet process. In *Artificial Intelligence and Statistics*, pp. 556–563, 2007.

van den Oord, A., Vinyals, O., et al. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, pp. 6297–6306, 2017.

Walker, S. G., Damien, P., Laud, P. W., and Smith, A. F. Bayesian nonparametric inference for random distributions and related functions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3): 485–527, 1999.

Wei, C. and Murray, I. Markov chain truncation for doubly-intractable inference. *arXiv preprint arXiv:1610.05672*, 2016.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. In *Reinforcement Learning*, pp. 5–32. Springer, 1992.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms, 2017.

Yeung, S., Kannan, A., Dauphin, Y., and Fei-Fei, L. Tackling over-pruning in variational autoencoders. *arXiv preprint arXiv:1706.03643*, 2017.