Non-conjugate Posterior using Stochastic Gradient Ascent with Adaptive Stepsize

Kart-Leong Lim
Institute of Microelectronics
*A*Star*
Singapore
lkartl@yahoo.com.sg

***Abstract*—Large scale Bayesian nonparametrics (BNP) learner such as Stochastic Variational Inference (SVI) can handle datasets with large class number and large training size at fractional cost. Like its predecessor, SVI rely on the assumption of conjugate variational posterior to approximate the true posterior. A more challenging problem is to consider large scale learning on non-conjugate posterior. Recent works in this direction are mostly associated with using Monte Carlo methods for approximating the learner. However, these works are usually demonstrated on non-BNP related task and less complex models such as logistic regression, due to higher computational complexity. In order to overcome the issue faced by SVI, we develop a novel approach based on the recently proposed constant stepsize stochastic gradient ascent to allow large scale learning on non-conjugate posterior. Unlike SVI, our new learner does not require closed-form expression for the variational posterior expectatations. Our only requirement is that the variational posterior is differentiable. In order to ensure convergence in stochastic settings, SVI rely on decaying step-sizes to slow its learning. Inspired by SVI and Adam, we propose the novel use of adaptive stepsizes in our method to significantly improve its learning. We show that our proposed methods is compatible with ResNet features when applied to large class number datasets such as MIT67 and SUN397. Finally, we compare our proposed learner with several recent works such as deep clustering algorithms and showed we were able to produce on-par or outperform the state-of-the-art methods in terms of clustering measures.**

***Index Terms*—Variational Inference, Stochastic Gradient Ascent, Non-Conjugate Posterior**

I. Introduction

Bayesian nonparametrics (BNP) is widely used in image processing, video processing and natural language processing. A common task in BNP also known as model selection is to automatically estimate the number of classes to represent an unlabelled dataset while clustering samples (or label) accordingly. A widely used BNP is the Variational Bayes Dirichlet process mixture [1], [2].

In the past, approximate learning for BNPs is mainly based on Variational Inference (VI) where it iteratively repeats its computational task (or algorithm) on the entire dataset, also known as batch learning [3], [4]. Today, most large scale BNP learners such as Stochastic Variational Inference (SVI) [2], [5], [6]. The latter repeats its computational task on a smaller set of randomly drawn samples (or minibatch) each iteration. This allows the algorithm to "see" the entire dataset especially large datasets when sufficient iterations has passed. However, both SVI and VI rely on closed-form solution to work. Thus, they are limited to to conjugate posteriors. To remove this constraint, several recent works turn to Monte Carlo gradient estimator (MC) to approximate the expectation (or gradient) of non-conjugate posterior. However, MC algorithms come at an expensive cost since it require generating samples from the approximated posteriors. Moreover, such works are usually confined to binary classifier such as logistic regression [7]–[10] or Gaussian assumptions [11], [12] and mainly demonstrated on datasets with smaller class numbers such as MNIST or UCI repository. Thus, the MC approach described above are more suitable to relatively simpler parameter inference problems.

Due to the recent paradigm shift towards deep ConvNet (CNN) [13], [14] and generative networks [12], [15]–[19], it is very rare to find newer works following the pipeline of SVI or MC since CNN and generative networks do not specifically deal with model selection or unsupervised class prediction.

The main problems faced by SVI and MC are:

1) SVI - The approximate posterior must come from the conjugate exponential family e.g. Gaussian-Gamma.
2) MC - Not scalable since method requires generating samples from the approximated posteriors which is expensive.

The contributions in this work are:

1) VI without closed form - We use stochastic gradient ascent instead of closed form coordinate ascent for VI as similarly in [8].
2) Adaptive stepsize - Inspired by Adam, we use decaying stepsize on both 1st and 2nd order moment of gradient for optimizating stochastic gradient ascent.
3) Non-conjugate posterior - We introduce the generalized Gaussian density as our mixture model. There is no closed form solution for the VI of this model.

This work was done at the Rapid-Rich Object Search lab, School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

We test the performance of our proposed learner on large class number datasets such as MIT67 and SUN397. Due to using deep ConvNet features (ResNet18), we also reported better results than most recent literature baselines.

This paper is organized as follows: Firstly, we recall VMM [20] for conjugate posteriors and discuss why it cannot work on nonconjugate posterior. Next, we propose using SGA for learning non-conjugate posterior. We further improve this learning with an Adam like stochastic optimization. We then present an algorithm that iteratively learns all the hidden variables of PYPM in a typical VI fashion. Lastly, we perform a study on several datasets including the more challenging MIT67 and SUN397 to evaluate the performance of our proposed method and enhancements. Finally, we include comparison with latest published works citing the datasets we use.

A. Related Works

SVI do not involve actual computation of SGA. Instead, SVI parameters are initially computed by closed-form solution [21] and then corrected via a weights biasing step. The weights follow a decay that gradually bias towards earlier computed values. SVI is recently demonstrated on BNP models with conjugate posterior such as the hierarchical Dirichlet Process topic model [2] and on large datasets as large as 3.8M samples and 300 classes.

On the other hand, MC methods use SGA to perform learning. Thus MC methods works on non-conjugate posterior. SGA was also recently discussed in [8] for learning approximate posterior. However, the authors mainly use SGA with constant stepsize for learning. Some notable works in this area include the black box VI [10], VI with stochastic search [9], the stochastic gradient variational Bayes [12] and the stochastic gradient Langevin dynamics [7].

II. Problem Statement

We present the problem of learning a non-conjugate posterior for model selection in Bayesian nonparametrics. We introduce a variant of Gaussian mixture model (GMM) that exist outside the exponential family distribution. Our model of choice is the Pitman-Yor process mixture (PYPM) with a generalized Gaussian mixture model (GGD). The GGD is a versatile 3 parameters model with mean, shape and scale parameters $\{B, s, \rho\}$. It can model the non-Gaussianity assumption for datasets. For simplicity, we only focus on the following assumption for PYPM, which has the simplest form for non-conjugate posterior i.e. by treating $\{s, \rho\}$ as constant variables for GGD

$$\begin{gathered} x \mid B, z \sim \mathcal{GGD}\left(B_k\right)^{z_{nk}} \\ B \sim \mathcal{N}\left(m_0, \lambda_0\right) \\ z_{nk} \mid v_k \sim Mult\left(\pi_k\right) \\ \nu_k \sim Beta(a_k, b_k) \end{gathered} \tag{1}$$

Only conjugate posterior can be learnt the traditional way e.g. MAP estimate followed by re-arranging a closed form solution. However, this strategy is not available for non-conjugate case:

We consider a case of non-conjugate posterior, $\ln q(B_k)$ where the likelihood is generalized Gaussian distributed and prior Gaussian distributed. When dealing with conjugate posterior, traditional VI technique such as the VMM [20] take the MAP estimate to obtain a closed form solution.

1) Taking the MAP estimate of $\ln q(B_k)$

$$\begin{gathered} E\left[B_k\right] = \arg\max_{B_k} \ln q(B_k) \\ = \arg\max_{B_k} \underset{z_{nk}}{E}\left[\ln p(x_n \mid B_k, z_{nk}) + \ln p(B_k)\right] \end{gathered} \tag{2}$$

2) Because the likelihood is not from the exponential family, re-arranging the gradient in terms of B_k for $\nabla_{B_k} \ln q(B_k) = 0$ is difficult

$$\begin{gathered} \nabla_{B_k} \ln p(x_n \mid B_k, E\left[z_{nk}\right]) \\ = \frac{\rho}{s}\left|\frac{x_n - B_k}{s}\right|^{\rho-1} sgn(\frac{x_n - B_k}{s}) E\left[z_{nk}\right] \end{gathered} \tag{3}$$

The above requires i) a numerical approach and ii) a converging learner for large sample size and large class number. Both problems are the main highlights of this work and shall be discussed in detail in the next section.

III. Proposed learning: Adaptive Stepsize for Variational Inference

Previously, the goal in (2) and (3) is to learn $\ln q(\theta_j)$ by deriving a closed-form expression for $E\left[\theta_j\right]$. Unfortunately, this is impossible unless $\ln q(\theta_j)$ is a conjugate posterior. In this section, we propose to estimate non-conjugate posterior using the stochastic gradient ascent (SGA) approach. We also seek stochastic learning, faster convergence and returning better local maxima. For the sake of brevity, we refer to θ as θ_j in this section.

A. Constant Stepsize SGA for Variational Inference

To overcome the lack of a closed-form solution for $E\left[\theta\right]$ in (2), some recent works [7], [9]–[12], [22], propose the learning of non-conjugate posterior using Monte Carlo gradient estimate, $\nabla_\theta E\left[f(\theta)\right] \approx \frac{1}{S}\sum_{s=1}^{S} f(\theta)\nabla_\theta \ln q(\theta_s)$ for approximation. However, this approximation is associated with large gradient variance and requiring generating posterior samples, θ_s. A more recent work [8] proposed using constant stepsize SGA for VI. Similarly, we can re-express the expectation of $\ln q(\theta)$ using constant stepsize SGA below (since approaching the local maximum has the same goal as maximizing the VLP globally)

$$\begin{gathered} E\left[\theta\right]_t = \int \theta_j \, q(\theta_j) \, d\theta_j \\ = E\left[\theta\right]_{t-1} + \eta \nabla_\theta \ln q(\theta) \end{gathered} \tag{4}$$

For SGA, we refer to the gradient of $\ln q(\theta)$ at iteration t using a minibatch with sample size M as

$$g_t = \frac{1}{M}\sum_{m=1}^{M} \nabla_\theta \ln q(\theta_m) \tag{5}$$

B. Adaptive Stepsize SGA for Variational Inference

In stochastic learning, we draw a small subset of samples (e.g. ¿1K samples) per iteration to update each posterior. This is more effective than taking the entire dataset (eg. ¿100K samples) for learning. In stochastic optimization [23], there is a requirement for a decaying step-size p_t to ensure convergence in SGA as given by $\sum p_t = \infty$ and $\sum p_t^2 < \infty$. This is to avoid SGA bouncing around the optimum of the objective function.

In SVI [2], the main goal is to obtain the "global parameter" update of conjugate posterior from its "immediate global parameter" as $(\phi_{global})_t = (1-p_t)(\phi_{global})_{t-1} + p_t \cdot \phi_{immed}$. The "immediate global parameter" is defined as a noisy estimate and is cheaper to run since it is computed from a data point sampled each iteration, rather than from the whole data set. The decaying step-size is defined as $p_t = (\tau + t)^{-\kappa}$ and both τ and κ are treated as constants. Our view of the SVI update equation above is much simpler and has little to do with SVI. Instead, we simply treat it as a weighted average between current and previous computed gradient of the posterior to ensure convergence in learning. In fact, in Table I we observe that SVI has a similar moment form to the common technique called "SGA with momentum". The main difference being p_t is a decaying term rather than fixed constant e.g. β_1. Thus, we define the **first moment of the gradient** (of the posterior) for a given minibatch of size M samples at t iteration as

$$W_t = (1 - p_t) W_{t-1} + p_t \cdot g_t \tag{6}$$

The non-conjugate learner in (4) is based on the SGA approach. Since we are dealing with an approximate posterior or posterior which is assumed convex, a more superior gradient learning is the natural gradient learning. Natural gradient learning is superior to plain vanilla gradient learning because the shortest path between two point is not a straight-line but instead falls along the curvature of the posterior objective [24]. Natural stochastic gradient ascent of posterior [2], [25] is defined as $E[\theta]_t = E[\theta]_{t-1} + \eta G^{-1} \nabla_\theta \ln q(\theta)$, where Fisher information matrix $G = E\left[\nabla_\theta \ln q(\theta) \left(\nabla_\theta \ln q(\theta)\right)^T\right]$. The motivation for the steepest ascent direction search of posterior optimum is best explained by Riemannian geometry in [2], [24], [25]. When we assumed each dimension is independent (spherical or diagonal), we end up with the squared gradient of posterior, $G = E\left[\left(\nabla_\theta \ln q(\theta)\right)^2\right]$. For a minibatch of size M samples, we introduce the **second moment of the gradient** for the squared gradient of posterior, using the identity $E\left[X^2\right] \geq \{E[X]\}^2$ as follows

$$F_t = (1 - p_t) F_{t-1} + p_t \cdot g_t^2 \tag{7}$$

We can take the product of the **first moment of the gradient** in (6) and the **second moment of the gradient** in (7) together to obtain an **adaptive stepsize** update

$$E[\theta]_t = E[\theta]_{t-1} + \eta \frac{W_t}{\sqrt{F_t + \epsilon}} \tag{8}$$

We defined $\epsilon = 10^{-8}$. In the next section, we will make comparison on (8) with other SGAs.

C. Motivation and Comparison with SGAs

Our adaptive stepsize learner is motivated by recent SGA methods and SVI as summarized in Table I. We briefly discuss their similarity below using the case of $\ln q(\theta)$.

SVI: In Table I, we compare (6) to the 1st moment in SVI. We can view the closed-form estimate $\hat{\theta}$ as g_t while $E[\theta]_t$ is seen as W_t in (6).

Momentum SGA: Similarly, when we fix p_t with a constant value (e.g. at iteration 45 in Fig 1.) over the decaying value in (6), both W_t in (6) and S_t will have very similar 1st moment in Table I.

Adam: At a glance, Adam appears to be similar to Momentum SGA for both their 1st moment. The only difference is that Adam normalize it with a decaying curve e.g. β_1^t. Thus, when we take an instaneous value in Fig 1, the value of M_t is proportional to S_t and vice versa for W_t. Our definition of W_t and F_t look very similar to M_t and V_t in Adam. The main difference lies in the way we define the stepsizes p_t. We adopt the decreasing stepsize defined by SVI. We also use an identical expression to Adam for the adaptive stepsize update in (8).

D. Brief analysis on convergence

We plot the curves for $(1 - p_t)$ and p_t to exhibit the behavior of using these stepsizes for W_t or F_t. We set the values $\tau = 1$ and $\kappa = 0.5$ for $t = 50$ iterations in Fig 1. As the number of iterations increases, for W_t and F_t, we see that the curves gradually shift responsibilities from the gradual diminishing value of p_t to the increasing value of $(1 - p_t)$.

Recall that in the SVI update $E[\theta]_t = (1 - p_t) E[\theta]_{t-1} + p_t \hat{\theta}$, the term $\hat{\theta}$ is defined as the closed form coordinate ascent estimate in [2]. Alternatively, $\hat{\theta}$ is computed identical to the conjugate posterior using VMM. Thus, when we let $\hat{\theta} = \nabla_\theta \ln q(\theta)$ at $\nabla_\theta \ln q(\theta) = 0$ we have the following for SVI

$$E[\theta]_t = (1 - p_t) E[\theta]_{t-1} + p_t \nabla_\theta \ln q(\theta) \tag{9}$$

For the proposed adaptive stepsize in (8), we only discuss the case of $E[\theta]_t = E[\theta]_{t-1} + \eta W_t$. Expanding the terms inside, we have the following

$$E[\theta]_t = E[\theta]_{t-1} + \eta (1 - p_t) W_{t-1} + \eta p_t \nabla_\theta \ln q(\theta) \tag{10}$$

Given that $\lim_{t\to\infty} (1 - p_t) = 1$ and $\lim_{t\to\infty} p_t = 0$ in Fig 1, we can see that SVI becomes

$$\lim_{t\to\infty} E[\theta]_t = E[\theta]_{t-1} \tag{11}$$

while (8) becomes

$$\lim_{t\to\infty} E[\theta]_t = E[\theta]_{t-1} + \eta W_{t-1} \tag{12}$$

(11) shows that SVI will reach convergence if $E[\theta]_{t-1}$ is a convex function. (12) consists of an additional term apart from $E[\theta]_{t-1}$. Specifically, W_{t-1} consists of a weighted sum between $\nabla_\theta \ln q(\theta)$ and the previous W_{t-1}. Thus, as long as $\nabla_\theta \ln q(\theta)$ is a convex function we can sufficiently ensure that the proposed stepsize in (8) will also converge.

TABLE I
COMPARISON OF LEARNERS FOR VARIATIONAL INFERENCE (SVI) AND NEURAL NETWORK (SGA)

	Methods	1st moment of Gradient	2nd moment of Gradient	Stepsize
VI	SVI	$E[\theta]_t = (1-p_t)\cdot E[\theta]_{t-1} + p_t \cdot \hat{\theta}$	-	-
	Proposed	$W_t = (1-p_t)\cdot W_{t-1} + p_t \cdot g_t$	$F_t = (1-p_t)\cdot F_{t-1} + p_t \cdot g_t^2$	$E[\theta]_t = E[\theta]_{t-1} + \eta\frac{W_t}{\sqrt{F_t+\epsilon}}$
Non-VI	SGA	-	-	$E[\theta]_t = E[\theta]_{t-1} + \eta\frac{g_t}{1}$
	Momentum SGA	$S_t = \beta_1 \cdot S_{t-1} + (1-\beta_1)\cdot g_t$	-	$E[\theta]_t = E[\theta]_{t-1} + \eta\frac{S_t}{1}$
	Adam	$M_t = \frac{\beta_1 \cdot M_{t-1} + (1-\beta_1)\cdot g_t}{1-\beta_1^t}$	$V_t = \frac{\beta_2 \cdot V_{t-1} + (1-\beta_2)\cdot g_t^2}{1-\beta_2^t}$	$E[\theta]_t = E[\theta]_{t-1} + \eta\frac{M_t}{\sqrt{V_t+\epsilon}}$

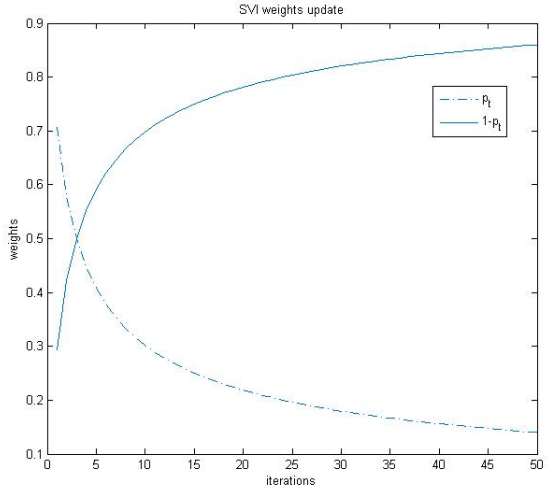

Fig. 1. Behavior of stepsizes using $p_t = (1+t)^{-0.5}$

IV. PROPOSED INFERENCE OF PYPM

We are ready to perform PYPM inference on a dataset given the expectation of all three posterior types (non-conjugate, discrete, conjugate) can be solved. Essentially, we repeat the estimation of all expectations using minibatch each iteration till convergence or sufficient iterations has passed. First, we turn to some formalities on PYPM and GGD. Second, we discuss our proposed inference of PYPM.

A. Pitman-Yor Process

For the last decade, Dirichlet process Gaussian mixture (DPM) has mainly found application in model selection of classification datasets such as UCI, MNIST, text classification, object recognition, scene recognition and etc. The model selection aspect of DPM actually comes from Dirichlet process while the distribution of each component of the mixture comes from a Gaussian. Both Dirichlet process and Gaussian mixture in DPM are assumed disjointed in VI. Another view of Dirichlet process is to consider it as a specific case of the Pitman-Yor process [26]. The latter can model additional tail behavior of dataset over Dirichlet process. The Pitman-Yor process is controlled by a two parameter Beta distribution where the parameters are $a_k = 1-d$ and $b_k = \alpha_0 + kd$ for $0 \leq d < 1$

$$Beta(v_k; a_k, b_k) \propto v_k^{(a_k-1)}(1-v_k)^{(b_k-1)} \tag{13}$$

If we set $d = 0$ in the above expression then Pitman-Yor process reduces back to the Dirichlet process.

B. Generalized Gaussian Density

In GGD, cluster mean is denoted $B = \{B_k\}_{k=1}^K \in \mathbb{R}^D$ and we have two new hidden variables, shape and scale. They are $s = \{s_k\}_{k=1}^K \in \mathbb{R}^D$ and $\rho = \{\rho_k\}_{k=1}^K \in \mathbb{R}^D$ respectively. Specific cases of GGD are the Gaussian PDF $\left(s = \sqrt{2}, \rho = 2\right)$ and Laplacian PDF $\left(s = \sqrt{2}, \rho = 1\right)$. Although, the GGD can be solved by the method of moments for s and ρ, there is no closed-form parameter estimation for GGD when B is non zero centered. In this work, we are only interested in exploring a new non-conjugate form to replace GMM. Hence for functionality, we limit our learning to B, while fixing the parameters s, ρ. The GGD pdf is defined as follows

$$\mathcal{GGD}(x|B,s,\rho) \propto \exp\left(-\left|\frac{x-B}{s}\right|^\rho\right) \tag{14}$$

C. PYPM

The joint probability of PYPM can be depicted as $p(x,B,z,v) = p(x \mid B,z)p(B)p(z \mid v)p(v)$. The observation is denoted $x = \{x_n\}_{n=1}^N \in \mathbb{R}^D$. The cluster assignment is denoted $z = \{z_n\}_{n=1}^N$ where z_n is a $1-of-K$ binary vector, subjected to $\sum_{k=1}^K z_{nk} = 1$ and $z_{nk} \in \{0,1\}$. We have earlier summarized the distribution of each term in PYPM in (1).

Non-Conjugate Posterior: The stochastic learning of PYPM is obtained by the proposed sVMM procedure for updating the generalized Gaussian-Gaussian posterior, $E[B_k]_t = E[B_k]_{t-1} + \eta\frac{W_t}{\sqrt{F_t}}$, whereby $g_t = \frac{1}{M}\sum_{m=1}^M \nabla_{B_k} \ln q(B_k)$.

Algorithm 1 Proposed Inference of PYPM

a) Input: $x \leftarrow \{minibatch\}$
b) Output: $E[z_{nk}]$
c) Initialization: $E[z_{nk}], m_0, \alpha_0, \lambda_0, a_k, b_k, K$
d) Repeat update until convergence,

1) non-conjugate posterior:

$$E[B_k]_t = E[B_k]_{t-1} + \eta \frac{W_t}{\sqrt{F_t}}$$

2) discrete posterior:

$$E[z_n] = \arg\max_{z_{nk}} \ln q(z_n)$$

3) conjugate posterior:

$$E[v_k] \approx \hat{v_k}$$

Due to requiring an initial or previous estimate, the non-conjugate posterior's gradient is computed as follows

$$\nabla_{B_k} \ln q(B_k) = \frac{\rho}{s} \left| \frac{x_n - E[B_k]_{t-1}}{s} \right|^{\rho-1} sgn(\frac{x_n - E[B_k]_{t-1}}{s}) E[z_{nk}] - \lambda_0 (E[B_k]_{t-1} - B_0) \tag{15}$$

Discrete Posterior: In VMM, we update the two conditional density by running through all possible K states of z_n that maximizes the posterior as below

$$\begin{aligned} E[z_{nk}] &= \arg\max_{z_{nk}} \underset{B_k, v_k}{E} [\ln p(x_n \mid B_k, z_{nk}) + \ln p(z_{nk} \mid v_k)] \\ &= \arg\max_{z_{nk}} -\left\{ \left| \frac{x_n - E[B_k]}{s} \right|^{\rho} \right. \\ &\quad \left. + \ln E[v_k] + \textstyle\sum_{l=1}^{k-1} \ln(1 - E[v_l]) \right\} z_{nk} \end{aligned} \tag{16}$$

Conjugate Posterior: Using VMM, we apply the MAP estiamte and re-arrange it to obtain a closed form for updating the Multinomial-Beta posterior below

$$E[v_k] = \frac{\sum_{n=1}^{N} E[z_{nk}] + (a_k - 1)}{\sum_{n=1}^{N} \sum_{j=k+1}^{K} E[z_{nj}] + (a_k - 1) + (b_k - 1)} \tag{17}$$

We summarized our inference of PYPM in Algo. 1.

V. Experiments

Proposed Variants: We consider three variants of proposed method in Table IV-VIII as shown below.

1) (Gau: SGA) SGA using (4) for solving $E[B_k]_t$, with Gaussian case where $s = \sqrt{2}, \rho = 2$ in $\mathcal{GGD}(B, s, \rho)$
2) (Gau: AdaSGA) Using our adaptive stepsize in (8) i.e. $E[B_k]_t = E[B_k]_{t-1} + \eta \frac{W_t}{\sqrt{F_t}}$ with Gaussian case as in variant 1.
3) (Lapl: AdaSGA) similar to variant 2 but now repeated with Laplacian case where $s = \sqrt{2}, \rho = 1$ in $\mathcal{GGD}(B, s, \rho)$

Strong Baseline: We implemented a strong baseline "SVI: DPM" to compare with our best proposed method. This baseline is the Dirichlet process Gaussian mixture and is also classified under BNP. It is implemented using the SVI update in Table I, after obtaining the closed-form expectation of posterior as found in [20]. The remainder of the DPM algorithm is identical to the proposed DPM algorithm in [20], but without the precision posterior. We ran at least 10 reruns and took their average (the values inside the bracket is their standard deviation).

Feature: DDPM-L and OnHGD are using the 128 dimensional SIFT features. For LDPO, the authors use 4096 dimensional AlexNet pretrained on ImageNet. DAEC, DC-Kmeans, DC-GMM and DEC are end-to-end models that rely on the pre-trained and fine-tuned encoder to perform feature extraction. In comparison, we use the 512 dimensional ResNet18 pretrained on ImageNet.

Truncation: For LDPO, DAEC, DC-Kmeans, DC-GMM, DEC, DBC, it is fixed to the ground truth. For SVI: DPM the truncation setting are identical to this work. Ground truth refers to the number of classes per dataset. It ranges from 15 to 397 classes (or clusters in our case). For unsupervised learning we do not require class labels for learning our models. However, we require setting a truncation level (upper limit) for each dataset as our model cannot start with an infinite number of clusters in practice. We typically use a very large truncation value (e.g. $K = 1000$ for SUN397) away from the ground truth to demonstrate that our model is not dependent on ground truth information.

Datasets: The datasets used in our experiments are detailed in Table II. There are 3 scene and 2 digit classification datasets in total. The largest dataset has about over 100K images, smallest dataset is at over 4K. We split the datasets into train and test partition.

Minibatch: For calculating our minibatch size, we approximate it by $M = sampleperclass * (gnd.truth)$, where $sampleperclass$ is typically 20 or 30 (for the datasets in this work) for sufficient statistics. In order to make the training dataset unbiased, we further assume each set of minibatch has sufficient sample draw from each class. This is necessary as some dataset have classes with 8000 samples while other classes have only 100 samples.

Evaluation Metric: We compare three criteria: i) Normalized Mutual Information, iii) Accuracy and iv) Model Selection. We use Normalized Mutual Information (NMI) and Accuracy (ACC) to evaluate the learning performance of our model. Model refers to the model selection estimated by each approach. The definition for ACC and NMI are $ACC = \frac{\sum_{n=1}^{N} \delta(gt_n, map(mo_n))}{N}$ and $NMI = \frac{MU_{info}(gt,mo)}{\max(H(gt),H(mo))}$ where $gt, mo, map, \delta(\cdot), MU_{info}, H$ refers to ground truth label, model's predicted label, permutation mapping function, delta function, mutual information and entropy respectively. Delta function is defined as $\delta(gt, mo) = 1$ if $gt = mo$ and equal 0 otherwise.

A. Comparison with Bayesian Nonparametrics

Bayesian nonparametrics (BNP): BNPs can perform clustering and estimate the cluster number jointly. The work here

#	Dataset	Classes	Train	Test	Trunc. Level	Minibatch Size
1	Scene15	15	750	3735	50	300
2	MIT67	37	3350	12,270	100	1340
3	SUN397	397	39,700	69,054	1000	11,910
4	MNIST	10	60,000	10,000	50	200
5	USPS	10	7,291	2,007	50	200

TABLE II
DATASETS (SCENE) FOR BAYESIAN NONPARAMETRICS

#	Methods	Year	Feature	Minibatch
1	Kmeans [27]	2017	AlexNet	no
2	LDPO-A-FC [27]	2017	AlexNet	no
3	OnHGD [28]	2016	SIFT	yes
4	SVI: DPM [2]	2013	ResNet	yes
5	DAEC [18]	2013	End-to-end	yes
6	DC-Kmeans [15]	2017	End-to-end	yes
7	DC-GMM [15]	2017	End-to-end	yes
8	DEC [16]	2016	End-to-end	yes
9	DBC [17]	2018	End-to-end	yes
10	ClusterGAN [29]	2019	End-to-end	yes
11	DASC [30]	2018	End-to-end	yes

TABLE III
RECENTLY PUBLISHED METHODS USED IN THIS COMPARISON

solely consider the pursuit of advancing statistical model for large scale datasets. The method are OnHGD (based on SVI) [28] and our baseline method SVI: DPM. Our work is also categorized under this area.

We compare our work with recent works citing the datasets we use. First, we group the published methods using the dataset in Table III. Next, we compare some of these published methods (non end-to-end) with our proposed variants in Table IV-VI. Also, we use 10 reruns for our proposed method and took their average. We can achieve convergence on our proposed variants with around 100 iterations.

1) Scene15: In Table IV, Gau: SGA is able to outperform LDPO-A-FC and SVI: DPM. and Kmeans. When adaptive stepsize is applied in Gau: AdaSGA, it further improves the clustering and model selection result. Lapl: AdaSGA is unable to significantly outperform Gau: AdaSGA for this dataset as the sample size is quite small.

2) MIT67: To the best of our knowledge, it is very rare to find recent deep clustering works (e.g. [15], [31]) addressing datasets beyond 10 classes for image datasets. The main reason we suspect is that most recent related works rely on end-to-end learning (i.e. the encoder of the autoencoder) rather than use an ImageNet pretrained CNN for feature extraction. It is likely more difficult to train or finetune the encoder to be as discriminative as ResNet especially when there is only about 200 samples per class for MIT67 in Table III.

In Table V, LDPO-A-FC is almost on par with its baseline comparison using Kmeans on the larger MIT67 dataset at ACC of 37.9% vs 35.6% respectively. Our baseline method "SVI: DPM" using ResNet18 feature also perform better than LDPO-A-FC at ACC of 61.21%. We outperformed the best published method by almost double in performance using "Lapl: AdaSGA" at ACC of 64.47%. We also notice that SVI: DPM outperforms Gau: SGA and Gau: AdaSGA. We believe SVI: DPM works better on larger dataset and the benefit of using a closed form solution is definitely more robust than a numerical approach such as SGA with all things being equal. Fortunately, Lapl: AdaSGA turns the verdict around by offering a more discriminative model that surpasses Gaussian for this dataset. The stronger model and the adaptive stepsize both attribute to the best performance of Lapl: AdaSGA on MIT67.

3) SUN397: In Table VI, OnHGD applies SVI [2] ("On" for online) to their BNP model HGD. They use OnHGD to learn a Bag-of-Words representation for SUN397. It appears they then use a supervised learner such as Bayes's decision rule for classification. No model selection was mentioned for SUN397 either. For SUN397, the ACC reported in OnHGD was 26.52% on SUN397. Although this is not a direct comparison, the same authors also reported an ACC of 67.34% for SUN16. In comparison, we obtained 83.37% on Scene15.

Our baseline "SVI: DPM" was able to get 39.07% on ACC compared to OnHGD of 26.52%. Both Gau: SGA and Gau: AdaSGA are performing worse that SVI: DPM. This is another evidence that SGA is inferior to SVI. Adaptive stepsize can help reduce the gap. Our best result is "Lapl: AdaSGA" which was able to slightly improve the results to 40.39% on the same dataset. The saving grace most likely being the discriminative power of the Laplacian mixture model.

Although not shown, the convergence of "Lapl: AdaSGA" is much slower for this particular dataset. Due to computational budget, we did not further check if better ACC can be obtained beyond 200 iterations using Algo 1. Also, our implementation for "SVI: DPM" faced some cluster singularity issue (cluster disappearing) when given too many iterations for SUN397. We had to stop iterations after around 15 or 20 as the cluster count may fall below 397.

B. Comparison with Deep Clustering

Deep clustering: A hybrid between neural network and statistical clustering, these works perform clustering in the feature space of the neural network, most of the works using autoencoder or GAN. These methods are DAEC [18], DC-Kmeans [15], DC-GMM [15], DEC [16], DBC [17] LDPO [27], DASC [30] and ClusterGAN [29]. Furthermore in these works, the clustering information further optimize the weights update in the hidden layers. However, the statistical clustering employed here are typically the fundamentals ones such as Kmeans or GMM.

1) MNIST: Most recent end-to-end clustering algorithms focus on digit recognition (i.e. MNIST and USPS) for experiments. Compared to MIT67 and SUN397, MNIST is a much easier dataset since the number of classes is mediocre (10 classes) and there is a large number of training images at 60k.

In Table VII, all the end-to-end methods (DAEC, DC-Kmeans/GMM, DEC, DBC) train a deep encoder (e.g. x-500-500-2000-10) as feature extractor. In comparison, we use ResNet feature directly as input to "SVI: DPM" and "Lapl: AdaSGA". Table VII shows the comparison between

	NMI	ACC	Model
Kmeans [27]	0.659	0.65	-
LDPO-A-FC [27]	0.705	0.731	-
SVI: DPM (baseline)	0.7877	0.7659	-
Gau: SGA (ours)	0.80333	0.81901	22
Gau: AdaSGA (ours)	0.81201	**0.83614**	21
Lapl: AdaSGA (ours)	**0.8165**	0.8337	**21**

TABLE IV
PERFORMANCE ON SCENE15

	NMI	ACC	Model
Kmeans [27]	0.386	0.356	-
LDPO-A-FC [27]	0.389	0.379	-
SVI: DPM (baseline)	0.6858	0.6121	-
Gau: SGA (ours)	0.66106	0.56496	78
Gau: AdaSGA (ours)	0.68546	0.60244	78
Lapl: AdaSGA (ours)	**0.7081**	**0.6447**	**78**

TABLE V
PERFORMANCE ON MIT67

	NMI	ACC	Model
OnHGD [28]	-	0.2652	-
SVI: DPM (baseline)	0.596	0.3907	-
Gau: SGA (ours)	0.5281	0.2612	489
Gau: AdaSGA (ours)	0.58226	0.34798	513
Lapl: AdaSGA (ours)	**0.6022**	**0.4039**	**487**

TABLE VI
PERFORMANCE ON SUN397

Methods	NMI	ACC
DAEC [18]	0.6615	0.734
DC-Kmeans [15]	0.7448	0.7448
DC-GMM [15]	0.8318	0.8555
DEC [16]	0.8273	0.8496
DBC [17]	0.917	0.964
ClusterGAN [29]	0.890	0.950
DASC [30]	0.780	0.804
SVI: DPM (baseline)	0.9233	0.9348
Lapl: AdaSGA (ours)	**0.9517**	**0.9580**

TABLE VII
COMPARISON ON MNIST

Methods	NMI	ACC
K-means [15]	0.4503	0.4585
DAEC [18]	0.5449	0.6111
DC-Kmeans [15]	0.5737	0.6442
DEC [16]	0.651	0.6246
DBC [17]	0.724	0.743
SVI: DPM (baseline)	0.6223	0.7763
Gau: AdaSGA (ours)	**0.6507**	**0.8010**

TABLE VIII
COMPARISON ON USPS

the published methods and ours on MNIST. For our best approach, "Lapl: AdaSGA", we are able to outperform our strong baseline "SVI: DPM" as well as obtain comparable ACC and NMI to the best published result by DBC or ClusterGAN.

2) USPS: In Table VIII, all the end-to-end methods (DAEC, DC-Kmeans, DEC, DBC) similarly trains a deep encoder as feature extractor. In comparison, K-means [15] using raw image pixel obtains 45.85% on ACC. For this particular dataset, we only use raw image pixel as direct input to both to "SVI: DPM" and "Gau: AdaSGA". Our best result using "Gau: AdaSGA" consistently outperformed all published result and strong baseline again on USPS at 80.10% on ACC. Our baseline is close behind at 77.63% on ACC. The best published method DBC obtained 74.3% but it outperforms our NMI measure. We believe the reason why most end-to-end methods cannot perform better than our methods on USPS even though they are using deep encoder features while we use pixel intensity is partly due to the comparatively small training size at 7K compared to say 60K on MNIST.

VI. CONCLUSION

The stochastic optimization of VI can be broadly categorized under two types. The first approach formulates the learning of the posterior using SGA while the second approach rely on traditional closed-form learning. In literature, the first approach require generating Monte Carlo sample from the variational posteriors, which is not practical for large datasets such as SUN397. The second approach suffers from the constraint of requiring analytical solution for the variational posterior expectation but has reported the capability to scale up to 3.8M samples and 200 classes. In this paper, we target up to about 100K samples and 400 classes using ResNet feature pretrained on ImageNet. We try to improve on the problems faced in both approaches. We first began with the constant stepsize SGA approach and in order to make it computationally efficient, we further stochastic optimization for VI. Stochastic optimization rely on decreasing step-size for guaranteed convergence. Inspired by Adam, we explored using first and second order moments of the gradient so as to achieve a faster convergence. We test our new stochastic learner on the Pitman-Yor process generalized Gaussian mixture which does not have closed-form learning for the posterior for specific case of Laplacian and Gaussian. We showed the significant performance gained in terms of NMI, ACC, model selection on large class number datasets such as the MIT67 and SUN397 and on MNIST and USPS with recent end-to-end deep learning related works.

REFERENCES

[1] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American Statistical Association*, vol. 112, no. 518, pp. 859–877, 2017.

[2] M. D. Hoffman, D. M. Blei, C. Wang, and J. W. Paisley, "Stochastic variational inference." *Journal of Machine Learning Research*, vol. 14, no. 1, pp. 1303–1347, 2013.

[3] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *the Journal of machine Learning research*, vol. 3, pp. 993–1022, 2003.

[4] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[5] J. Paisley, C. Wang, D. M. Blei, and M. I. Jordan, "Nested hierarchical dirichlet processes," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 2, pp. 256–270, 2015.

[6] A. Kucukelbir, D. Tran, R. Ranganath, A. Gelman, and D. M. Blei, "Automatic differentiation variational inference," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 430–474, 2017.

[7] M. Welling and Y. W. Teh, "Bayesian learning via stochastic gradient langevin dynamics," in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 2011, pp. 681–688.

[8] S. Mandt, M. D. Hoffman, and D. M. Blei, "Stochastic gradient descent as approximate bayesian inference," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 4873–4907, 2017.

[9] J. Paisley, D. M. Blei, and M. I. Jordan, "Variational bayesian inference with stochastic search," in *Proceedings of the 29th International Coference on International Conference on Machine Learning*. Omnipress, 2012, pp. 1363–1370.

[10] R. Ranganath, S. Gerrish, and D. Blei, "Black box variational inference," in *Artificial Intelligence and Statistics*, 2014, pp. 814–822.

[11] D. J. Rezende and S. Mohamed, "Variational inference with normalizing flows," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning-Volume 37*. JMLR. org, 2015, pp. 1530–1538.

[12] D. P. Kingma and M. Welling, "Stochastic gradient vb and the variational auto-encoder," in *Second International Conference on Learning Representations, ICLR*, 2014.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[15] K. Tian, S. Zhou, and J. Guan, "Deepcluster: A general clustering framework based on deep learning," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2017, pp. 809–825.

[16] J. Xie, R. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *International conference on machine learning*, 2016, pp. 478–487.

[17] F. Li, H. Qiao, and B. Zhang, "Discriminatively boosted image clustering with fully convolutional auto-encoders," *Pattern Recognition*, vol. 83, pp. 161–173, 2018.

[18] C. Song, F. Liu, Y. Huang, L. Wang, and T. Tan, "Auto-encoder based data clustering," in *Iberoamerican Congress on Pattern Recognition*. Springer, 2013, pp. 117–124.

[19] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[20] K.-L. Lim and H. Wang, "Fast approximation of variational bayes dirichlet process mixture using the maximization–maximization algorithm," *International Journal of Approximate Reasoning*, vol. 93, pp. 153–177, 2018.

[21] D. M. Blei, M. I. Jordan *et al.*, "Variational inference for dirichlet process mixtures," *Bayesian analysis*, vol. 1, no. 1, pp. 121–144, 2006.

[22] D. A. Knowles, "Stochastic gradient variational bayes for gamma approximating distributions," *arXiv preprint arXiv:1509.01631*, 2015.

[23] H. Robbins and S. Monro, "A stochastic approximation method," in *Herbert Robbins Selected Papers*. Springer, 1985, pp. 102–109.

[24] S.-I. Amari, "Natural gradient works efficiently in learning," *Neural computation*, vol. 10, no. 2, pp. 251–276, 1998.

[25] A. Honkela, M. Tornio, T. Raiko, and J. Karhunen, "Natural conjugate gradient in variational inference," in *International Conference on Neural Information Processing*. Springer, 2007, pp. 305–314.

[26] Y. W. Teh and M. I. Jordan, "Hierarchical bayesian nonparametric models with applications," *Bayesian nonparametrics*, vol. 1, pp. 158–207, 2010.

[27] X. Wang, L. Lu, H.-C. Shin, L. Kim, M. Bagheri, I. Nogues, J. Yao, and R. M. Summers, "Unsupervised joint mining of deep features and image labels for large-scale radiology image categorization and scene recognition," in *Applications of Computer Vision (WACV), 2017 IEEE Winter Conference on*. IEEE, 2017, pp. 998–1007.

[28] W. Fan, H. Sallay, and N. Bouguila, "Online learning of hierarchical pitman–yor process mixture of generalized dirichlet distributions with feature selection," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 9, pp. 2048–2061, 2016.

[29] K. Ghasedi, X. Wang, C. Deng, and H. Huang, "Balanced self-paced learning for generative adversarial clustering network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4391–4400.

[30] P. Zhou, Y. Hou, and J. Feng, "Deep adversarial subspace clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1596–1604.

[31] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 132–149.