Improving Inference in Latent Variable Models

Shubham Anand Jain
Department of Computer Science
Stanford University
Stanford, CA 94305
shubhamj@stanford.edu

Pranay Reddy Samala
Department of Computer Science
Stanford University
Stanford, CA 94305
pranayr@stanford.edu

Jian Vora
Department of Computer Science
Stanford University
Stanford, CA 94305
jianv@stanford.edu

1 Introduction

In various forms of data, we often want to separate certain features in them; for example, in face image data, we want to separate gender, hair, eyes, pose, etc. When we perform discriminative modelling, we are unable to capture or characterize the different features in a meaningful manner. However, generative modelling allows us to capture these features directly by introducing latent variable models, which are a proxy for the actual features. The variational autoencoder is such a generative tool that is extensively used for unsupervised representation learning [1].

However, it turns out that these variational autoencoders are much harder to optimize parameters over by maximizing likelihood than autoregressive models. Thus, we instead resolve to maximize a lower bound to the likelihood, which we term as ELBO (Evidence Lower Bound). To maximize this ELBO, we have an encoder structure that maps every input to a distribution over latent variables $q_\phi(z|x)$, and a decoder that parameterizes the distribution $p(x|z)$. In general, a latent variable model can be used to model the input distribution with $\int p(z)p(x|z)dz$ where $p(z)$ is the prior over the latent variables which is chosen beforehand.

Inference in latent variables models means to infer the posterior $p(z|x)$ if we are given the joint $p(x, z)$. This is hard problem as it involves solving high-dimensional integrals which are intractable and hence people resort to approximation methods. In our project, we discuss and investigate a method of maximizing ELBO and trying to perform better inference in VAEs. This has applications in tasks such as image inpainting and de-noising.

2 Problem Statement & Related Work

For VAEs, in practice, we often map the input to a distribution over latent variables using a neural network, which tells us the parameters of the posterior distribution. This makes it easy to train the VAE, and is called amortized inference. Thus, in inference models, we usually have 3 types of gaps created in the ELBO [2, 3, 4, 5]:

- Approximation gap: This is the gap between the best-fit distribution from a variational family to the actual distribution $p(x)$. Thus, this is the gap created due to the choice of variational family in a model. A more expressive variational family will lead to a smaller Approximation gap.

- Amortization gap: Amortization gap is created due to our approximation of using a common parameterized network (across all x; for example, any encoder) to map the input x to parameters ϕ, which makes inference easy to perform, rather than having to optimize the parameters at each x. Due to this approximation, we do not get the best-fit model from the family; Amortization gap represents this difference.
- Inference gap: Inference gap is the total gap between a model's ELBO estimate and the actual $\log(p(x))$. It is the sum of Approximation and Amortization gaps.

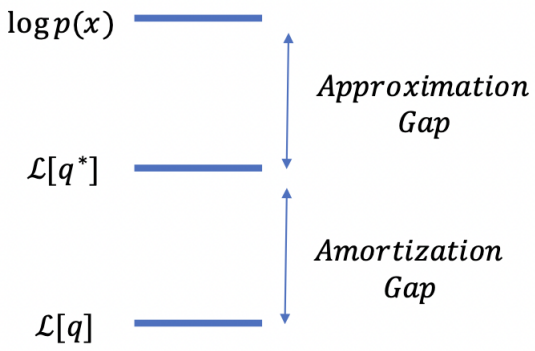


Figure 1. Gaps in Inference

The problem statement is to try and reduce the Inference gap in VAEs. Ideally, we should learn a different set of distribution-characterizing parameters for every input - this is called unamortized inference. It is intuitive that unamortized inference will lead to a tighter ELBO bound. In our project, we propose a model that aims to reduce both Approximation and Amortization gaps in latent variable models. We do this by:

- Using an amortized model by picking an initial ϕ value from an encoder, and then optimizing parameters for a test point x; this should reduce the amortization gap.
- We use hierarchical VAEs with top-down and bottom up merging. The intuition from the merging idea is drawn from the precision-weighted merging trick that was previously applied in Ladder Networks [6, 7].
- We also use a normalizing flow after sampling from the posterior to go the z-space, which expands the variational family and allows the encoder to more closely represent the maximum likelihood of the real distribution wrt the decoder family, which reduces the NELBO. This idea is drawn from the NVAE paper [8].

A secondary problem statement, which is a direct application of our ideas above, is performing better image in-painting and denoising using VAEs. To perform Image inpainting and denoising, we use the iterative Gibbs-sampling approach described in [9, 10], which is expanded upon in the next section.

3 Technical Approach

In a variational auto-encoder, we send the input x to a latent space z, and we try to model $p(x)$ as $\int p(x|z)p(z)$, where $p(x|z)$ and $p(z)$ are functions that are easy to represent. Specifically, in a VAE, we consider the following model:

$$p(z) = \mathcal{N}(z|0, I)$$
$$p_\theta(x|z) = \text{Bern}(x|f_\theta(x))$$

Ideally, we want to choose θ to maximize $p_\theta(x)$ (i.e, the likelihood of the data that we get). However, evaluating $p_\theta(x)$ is complicated. So we take inspiration from the evidence lower bound (ELBO), which is as follows and holds for any function $q_\phi(z|x)$:

$$\log p_\theta(x) \geq \text{ELBO}(x;\theta,\phi) = \mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x|z)\right] - D_{KL}(q_\phi(z|x)||p(z))$$

We will choose a large variational family $q_\phi(z|x)$, and then maximize the ELBO in lieu of the actual likelihood. This acts as an approximation to maximum likelihood estimation. In a VAE, the variational approximation we consider is as follows:

$$q_\phi(z|x) = \mathcal{N}(z|\mu_\phi(x), \text{diag}(\sigma^2_\phi(x)))$$

We note that here we have a encoder $q_\phi(z|x)$, which is a neural network finding the parameters of the posterior for any input point. This is called amortization. Ideally, we should fit the posterior parameters that arise for each input point x separately; this would be called unamortized inference.

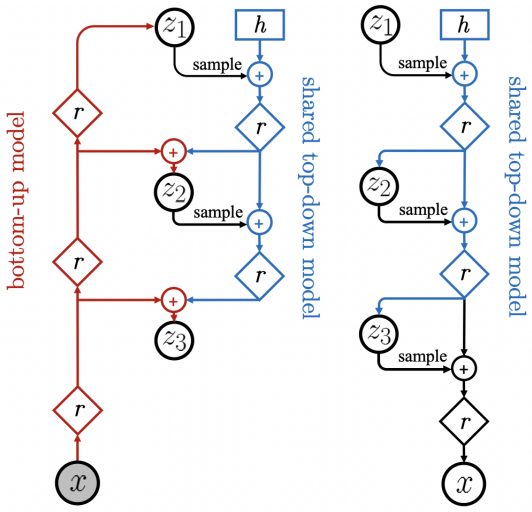
A GMVAE is a Mixture of Gaussians VAE which expands the variational family by choosing a more complicated learnable prior. In other words, our prior is as follows:

$$p_\theta(z) = \sum_{i=1}^{k} \frac{1}{k}\mathcal{N}(z|\mu_i, \text{diag}(\sigma_i^2))$$

NVAE

Nouveau VAE (NVAE) [8] is a hierarchical VAE which has multiple scales of latent variables (z_1 to z_L) unlike traditional VAEs which have to model the latent space components to be independent of each other. In hierarchical VAEs, there exists an autroregressive structure in the latent space prior and the posterior which makes the encoding distribution more expressive.

NVAE is not only an hierarchial VAE but also implements weight sharing between the top-down and the bottom-up path which ensures that a common signal is used for finding the autoregressive structure in the latent space for both the encoder and the decoder. In other words, NVAE has a bidirectional encoder and it's the top-down path of the encoder has the same weights as that of the decoder. The model architecture of NVAE is shown below:



(a) Bidirectional Encoder (b) Generative Model

Figure 1: NVAE model architecture [8]

As we can see in figure 1, the top-down path in the encoder has the same weights as the decoder(generative model), both of which are denoted by the same blue color. During training time, an input sample x is passed through the encoder to first get z_1. The blocks labelled r are just neural networks with residual connections and swish activations. From z_1 and h(which can be a simply initialization vector like all zeros), we keep on finding z_2 to z_L ($L = 3$ in the above diagram). From the top-down path, we get the reconstruction $\hat{x}$ which is used to compute the NELBO for training. The loss function of NVAE is as follows:

$$\mathcal{L}_{\text{VAE}}(\boldsymbol{x}) := \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}\left[\log p(\boldsymbol{x}|\boldsymbol{z})\right] - \text{KL}(q(\boldsymbol{z}_1|\boldsymbol{x})||p(\boldsymbol{z}_1)) - \sum_{l=2}^{L} \mathbb{E}_{q(\boldsymbol{z}_{<l}|\boldsymbol{x})}\left[\text{KL}(q(\boldsymbol{z}_l|\boldsymbol{x}, \boldsymbol{z}_{<l})||p(\boldsymbol{z}_l|\boldsymbol{z}_{<l}))\right].$$

Figure 2: NVAE NELBO Loss Function [8]

Why NVAE?
As mentioned earlier, we want to bridge both the approximation and the amortized gap. For bridging the approximation gap, we have to make the encoding distribution $q_\phi(z|x)$ to be more expressive in order to model the $p(z|x)$. Adding additional latent representations helps in making the encoder more expressive. NVAE also uses a flow after the latent variables in order to make the encoder more expressive. Thus, the idea is that use of hierarchical VAE with weight sharing helps in reducing the approximation gap.

Unamortized Inference

To reduce the second type of gap (amortization gap), we perform un-amortized inference where the encoder parameters are optimised for each datapoint. We perform unamortized inference by first training an encoder and decoder in an amortized manner, and then performing gradient descent on the NELBO (Negative ELBO) over the input parameters to find the posterior parameters for each sample datapoint $x^{(i)}$. To be precise, the encoder gives the parameters $\mu_\phi(x), \sigma^2_\phi(x)$ to the posterior distribution. For our model, we perform gradient descent on $\mu(x), \sigma^2(x)$, and initialize the descent with the encoder parameters $\mu_\phi(x), \sigma^2_\phi(x)$. We used the Adam optimizer with a learning rate of 10^{-3} and study how they number of iterations affects the performance.

Applications of Inference

We study two broad applications of inference in latent variable model:

1. Image Inpainting
2. Image Denoising

For image inpainting, we perform iterative Gibbs sampling [9] which works as follows:

- Let the image at time t be x'_t
- We sample a latent variable $z_t \sim q_\phi\left(z|x'_t\right)$
- We then sample an output $x_t \sim p_\theta(x|z_t)$
- Let our image at time $t+1$ be $x'_{t+1} = m \odot x'_t + (1-m) \odot x_t$

Here, m is the binary mask of the image. We perform 100 iterations of the above approach to perform inpainting. For image denoising, we perform the same iterations as of Gibb's sampling but this time we do not have any mask for performing the update. The idea for both these tasks is that if the model can infer the latent variable well, then even slight perturbations in the input image should not change the latent variable we infer much and hence on passing through the decoder, we should get a reconstructed image which looks like that it is drawn from the true data distribution.

Table 1: NELBO on MNIST, unamortized inference, VAE

GRADIENT DESCENT STEPS	NELBO ON TRAIN DATA	NELBO ON TEST DATA
1	99.86	101.34
10	99.71	101.16
100	99.07	100.23
1000	96.67	98.9

Table 2: NELBO on MNIST, unamortized inference, GMVAE

GRADIENT DESCENT STEPS	NELBO ON TRAIN DATA	NELBO ON TEST DATA
1	98.43	98.43
10	96.72	98.27
100	95.34	97.12
1000	92.2	95.64

4 Results

Metrics

We utilize three major quantitative metrics to define model performance: NELBO, FID scores and SSIM scores. The NELBO is primarily utilized once we fix a model to compare performance between unamortized and amortized inference. We use FID and SSIM metrics to compare quality of generated images across models. The mathematical description of the metrics are:

1. NELBO: The NELBO is a component of the VAE loss, and has already been defined in the previous sections.
2. The Frechet Inception Distance (or FID)[1] is a metric for evaluating the quality of generated images. The metric is computed by first passing generated images into the Inception V3 network to obtain activations. A multivariate gaussian is then fit on these output activations, for both generated and real images. If μ, Σ are the parameters for the generator, and μ_w, Σ_w are the parameters for the real data, then the distance metric is computed as $||\mu - \mu_w||_2^2 + \mathbf{tr}(\Sigma + \Sigma_w - 2(\Sigma\Sigma_w)^{\frac{1}{2}})$. Essentially, this metric is a measure of the similarity between real images and generated images.
3. Structural Similarity Index Measure (SSIM)[2] is another metric used for measuring the similarity between two images. At a high level, the metric computes the patchwise similarity between two images. We use this metric for the tasks of denoising and inpainting to compare the denoised/inpainted image with the original image.

Preliminary Results

Our first attempt was on standard VAEs and GMVAEs before moving to hierarchical architectures such as NVAE. We evaluated how scaling the number of gradient descent steps for unamortized inference affects the NELBO for this model. From tables 1, 2, we can observe that the NELBO of the model decreases as expected. Furthermore, in Figure 4, we observe the qualitative effect of changing the gradient descent steps across model architectures. The images infilled using a GMVAE are better than the simple VAE, which is to be expected due to a lower approximation gap.

[1] `https://github.com/mseitzer/pytorch-fid`

[2] `https://github.com/jorge-pessoa/pytorch-msssim`

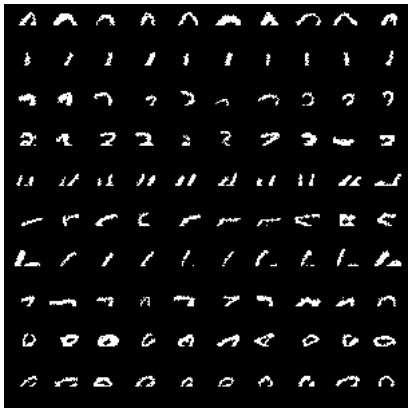
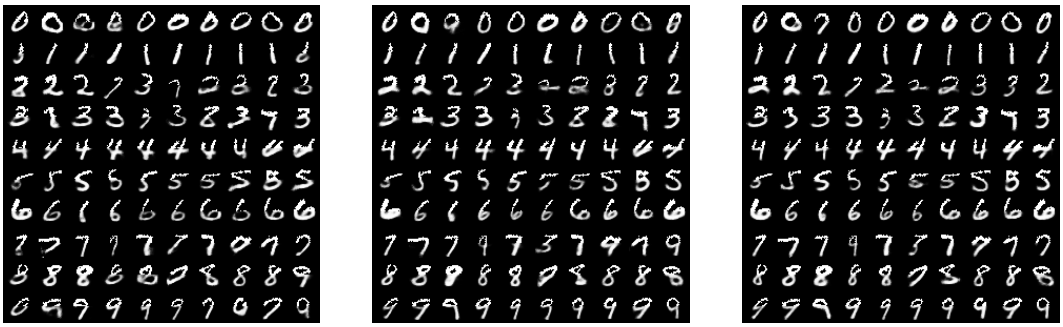


Figure 3: Image to be inpainted



(a) VAE, GD Steps = 1 *(b) VAE, GD Steps = 1000* *(c) GMVAE, GD Steps = 1000*

Figure 4: Image inpainting for VAEs and GMVAEs: The first image is obtained after inpainting by optimizing the inference step using 1 gradient step update. The second & third images are obtained after performing 1000 gradient descent update steps; there is a clear visual contrast between the sharpness on changing the Gradient Descent Iterations. We can also observe that GMVAEs are able to impute some digits better than VAEs (example digit 7).

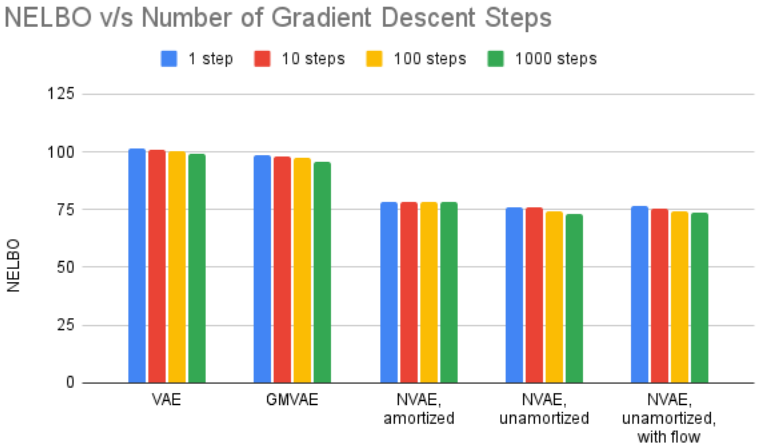
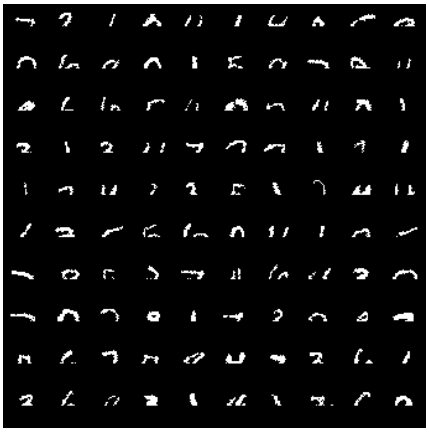


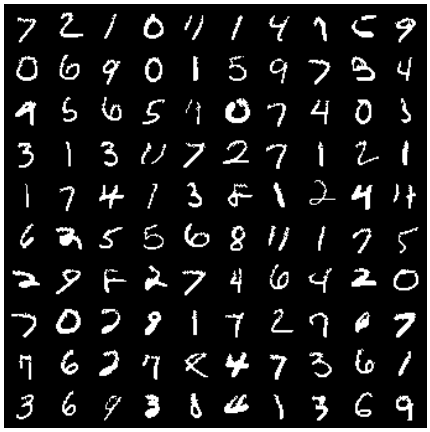
Figure 5: NELBO comparison across models

Technical Variations

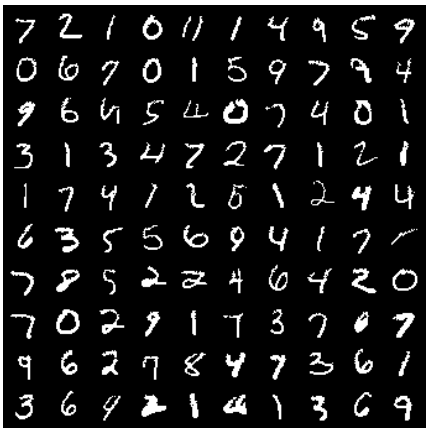
We now move to reducing the approximation gap further by using hierarchical models such as NVAE. We use an NVAE model pre-trained on MNIST for all our experiments. Hence, this means that we use the same decoder and never update the decoder parameters.



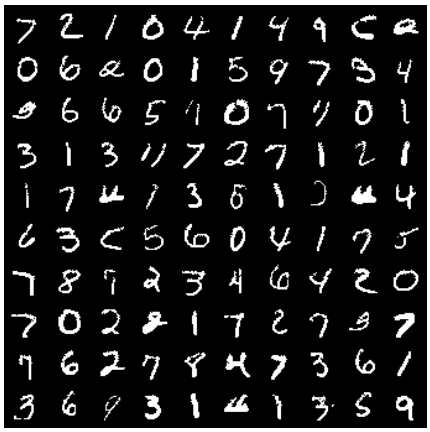
(a) Masked Image with missing pixels



(b) Amortized Inference



(c) Unamortized Inference



(d) Unamortized with Flow re-training

Figure 6: Inpainting, comparison across methods for NVAE

On NVAE, we performed three different methods for inpainting and denoising. We first applied standard amortized inference as the baseline. We then attempted two variations of unamortized inference, one variation involving gradient descent on the parameters μ, σ of the latent variables, and another with gradient descent on both flow parameters and on μ, σ. We compare these different variations both qualitatively and quantitatively for both inpainting and denoising.

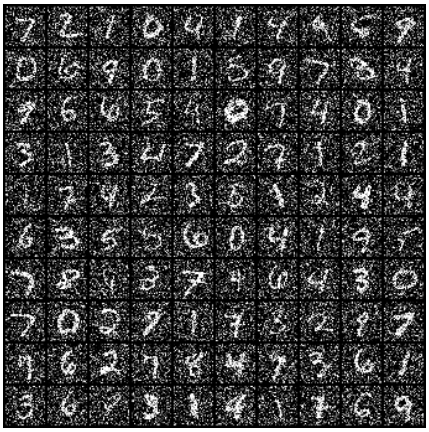
MODEL	FID SCORE (↓)	SSIM SCORE (↑)
NVAE, AMORTIZED	11.52	0.9915
NVAE, UNAMORTIZED	**10.90**	**0.9925**
NVAE, UNAMORTIZED, WITH FLOW	11.23	0.9875

Table 3: SSIM and FID scores

We also perform an experiment to compare the NELBO of these three NVAE methods with the VAE and the GMVAE. Results are given in Figure 5.

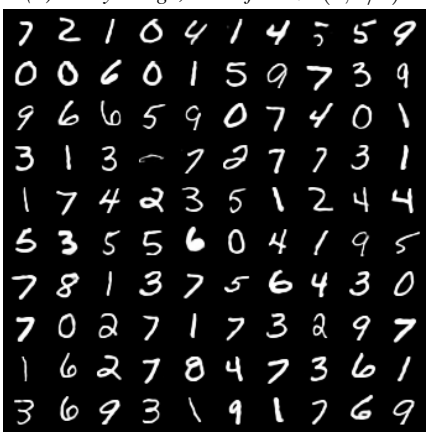
Analysis of results

We note that, in Figure 6 the NVAE with flow retraining tends to give the most clear images and manages to reconstruct each digit in some manner, even with a noise variance of 0.5 (even though they are not the most accurate, as quantitatively validated in Table **??**). It could be that the normalizing flow smoothens out the image but does not correlate to being similar to the data

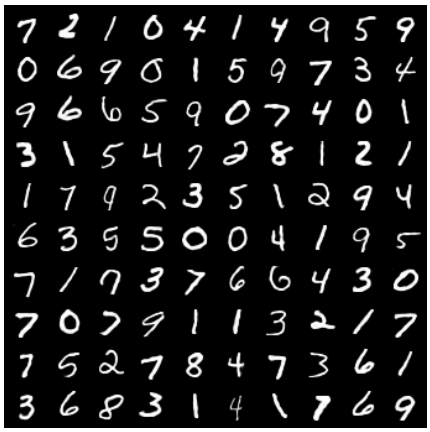


(a) Noisy Image, noise from $\mathcal{N}(0, 1/2)$

(b) Amortized Inference



(c) Unamortized Inference



(d) Unamortized with Flow re-training

Figure 7: De-noising, comparison across methods for NVAE

distribution.
In Figure 6, we note that the image formed by unamortized inference is the clearest, with a higher amount of digits identified. Flow-retraining does not seem to do well here.

The trend in Figure 5 represents a trend of lowering NELBO as training proceeds and gradient descent steps increase, with the winner between unamortized inference and unamortized with flow-retraining being unclear. The NELBO of the NVAE is much lower than the VAE and the GMVAE NELBOs, which demonstrates it's clear superiority as an architecture. The code for the above results can be found here:
1. `https://github.com/pranayreddys/CS236-Project-Unamortized-Inference`
2. `https://github.com/jianvora/VAE_unamortized_CS236`

5 Conclusions & Future Work

We observe from our results in section 4 for the VAE, GMVAE and NVAE that unamortized inference not only lowers the NELBO, but also improves image quality significantly in practice. However, there is a trade-off between time taken to optimize the parameters and the quality we obtain.

The idea that we used in this paper, which we coin "unamortized inference", is known in the literature as Stochastic Variational Inference.

In Stochastic Variational Inference, multiple gradient steps leads to a trade-off between compute, time and high-quality inference. Recent ideas such as Semi-amortized Variational Inference

[11], and Recursive Inference for Variational Autoencoders [12] provide a way to perform a single pass on getting x to update the encoder, rather than multiple gradient steps (or in general faster methods to update the encoder rather than performing multiple descents). This is useful, since it allows us to perform high-quality inference with low amounts of compute and time taken. Thus, it might be an interesting area for future work to combine the two ideas and try to train the NVAE with Recursive Inference for every datapoint x.

References

[1] Diederik P. Kingma and Max Welling. *An Introduction to Variational Autoencoders*. 2019.

[2] Chris Cremer, Xuechen Li, and David Duvenaud. Inference suboptimality in variational autoencoders. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1078–1086. PMLR, 10–15 Jul 2018.

[3] Matthew D. Hoffman, David M. Blei, Chong Wang, and John Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14(4):1303–1347, 2013.

[4] Alex Graves. Practical variational inference for neural networks. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011.

[5] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2014.

[6] Casper Kaae Sø nderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sø nderby, and Ole Winther. Ladder variational autoencoders. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[7] Cheng Zhang, Judith Bütepage, Hedvig Kjellström, and Stephan Mandt. Advances in variational inference. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(8):2008–2026, 2019.

[8] Arash Vahdat and Jan Kautz. Nvae: A deep hierarchical variational autoencoder. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 19667–19679. Curran Associates, Inc., 2020.

[9] Cusuh Ham, Amit Raj, Vincent Cartillier, and Irfan Essa. Variational image inpainting. In *Advances in Neural Information Processing Systems, Bayesian Deep Learning Workshop*, volume 29. Curran Associates, Inc., 2016.

[10] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1278–1286, Bejing, China, 22–24 Jun 2014. PMLR.

[11] Yoon Kim, Sam Wiseman, Andrew Miller, David Sontag, and Alexander Rush. Semi-amortized variational autoencoders. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2678–2687. PMLR, 10–15 Jul 2018.

[12] Minyoung Kim and Vladimir Pavlovic. Recursive inference for variational autoencoders. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 19632–19641. Curran Associates, Inc., 2020.