# REVISITING REWEIGHTED WAKE-SLEEP

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Discrete latent-variable models, while applicable in a variety of settings, can often be difficult to learn. Sampling discrete latent variables can result in high-variance gradient estimators for two primary reasons: 1. branching on the samples within the model, and 2. the lack of a pathwise derivative for the samples. While current state-of-the-art methods employ control-variate schemes for the former and continuous-relaxation methods for the latter, their utility is limited by the complexities of implementing and training effective control-variate schemes and the necessity of evaluating (potentially exponentially) many branch paths in the model. Here, we revisit the reweighted wake-sleep (RWS) (Bornschein & Bengio, 2015) algorithm, and through extensive evaluations, show that it circumvents both these issues, outperforming current state-of-the-art methods in learning discrete latent-variable models. Moreover, we observe that, unlike the importance weighted autoencoder, RWS learns better models *and* inference networks with increasing numbers of particles, and that its benefits extend to continuous latent-variable models as well. Our results suggest that RWS is a competitive, often preferable, alternative for learning deep generative models.

## 1 INTRODUCTION

Learning deep generative models with discrete latent variables opens up an avenue for solving a wide range of tasks including tracking and prediction (Neiswanger et al., 2014), clustering (Rasmussen, 2000), model structure learning (Adams et al., 2010), speech modeling (Juang & Rabiner, 1991), topic modeling (Blei et al., 2003), language modeling (Chater & Manning, 2006), and concept learning (Kemp et al., 2006; Lake et al., 2018). Furthermore, recent deep-learning approaches addressing counting (Eslami et al., 2016), attention (Xu et al., 2015), adaptive computation time (Graves et al., 2014), and differentiable data structures (Graves et al., 2014; 2016; Grefenstette et al., 2015), underscore the importance of models with conditional branching induced by discrete latent variables.

Current state-of-the-art methods optimize the evidence lower bound (ELBO) based on the importance weighted autoencoder (IWAE) (Burda et al., 2016) by using either reparameterization (Kingma & Welling, 2014; Rezende et al., 2014), continuous relaxations of the discrete latents (Maddison et al., 2017; Jang et al., 2017) or the REINFORCE method (Williams, 1992) with control variates (Mnih & Gregor, 2014; Mnih & Rezende, 2016; Gu et al., 2016; Tucker et al., 2017; Grathwohl et al., 2018).

Despite the effective large-scale learning made possible by these methods, several challenges remain. First, with increasing number of particles, the IWAE ELBO estimator adversely impacts inference-network quality, consequently impeding learning of the generative model (Rainforth et al., 2018). Second, using continuous relaxations results in a biased gradient estimator, and in models with stochastic branching, forces evaluation of potentially exponential number of branching paths. For example, a continuous relaxation of the cluster identity in a Gaussian mixture model (GMM) (Section 4.3) forces the evaluation of a weighted average of likelihood parameters over *all* clusters instead of selecting the parameters based on just one. Finally, while control-variate methods may be employed to reduce variance, their practical efficacy can be somewhat limited as in some cases they involve designing and jointly optimizing a separate neural network which can be difficult to tune (Section 4.3).

To address these challenges, we revisit the reweighted wake-sleep (RWS) algorithm (Bornschein & Bengio, 2015), comparing it extensively with state-of-the-art methods for learning discrete latent-variable models, and demonstrate its efficacy in learning better generative models and inference networks, and improving the variance of the gradient estimators, over a range of particle budgets.

Going forward, we review the current state-of-the-art methods for learning deep generative models with discrete latent variables (Section 2), revisit RWS (Section 3), and present an extensive evaluation of these methods (Section 4) on (i) the Attend, Infer, Repeat (AIR) model (Eslami et al., 2016) to perceive and localise multiple MNIST digits, (ii) a continuous latent-variable model on MNIST, and (iii) a pedagogical GMM example, exposing a shortcoming of RWS that we fix using defensive importance sampling (Hesterberg, 1995). Our experiments confirm that RWS is a competitive, often preferable, alternative that unlike IWAE, learns better models and inference networks with increasing particle budgets.

## 2 Background

Consider data $(x^{(n)})_{n=1}^N$ sampled from a true (unknown) generative model $p(x)$, a family of generative models $p_\theta(z, x)$ of latent variable $z$ and observation $x$ parameterized by $\theta$ and a family of inference networks $q_\phi(z|x)$ parameterized by $\phi$. We would like to learn the generative model by maximizing the marginal likelihood over data: $\theta^* = \arg\max_\theta \frac{1}{N}\sum_{n=1}^N \log p_\theta(x^{(n)})$. We would simultaneously also like to learn an inference network $q_\phi(z|x)$ that amortizes inference given observation $x$; i.e., $q_\phi(z|x)$ maps an observation $x$ to an approximation of $p_{\theta^*}(z|x)$. Amortization ensures that evaluation of this function is cheaper than performing approximate inference of $p_{\theta^*}(z|x)$ from scratch. Our focus here is on such joint learning of the generative model and the inference network, here referred to as “learning a deep generative model”, although we note that other approaches exist that learn just the generative model (Goodfellow et al., 2014; Mohamed & Lakshminarayanan, 2016) or the inference network (Paige & Wood, 2016; Le et al., 2017) in isolation.

We begin with a review of IWAEs (Burda et al., 2016) as a general approach for learning deep generative models using stochastic gradient descent (SGD) methods, focusing on generative-model families with discrete latent variables, for which the high variance of the naïve gradient estimator impedes learning. We will also review control-variate and continuous-relaxation methods for gradient-variance reduction. The IWAE used alongside such gradient-variance reduction methods is currently the dominant approach for learning deep generative models with discrete latent variables.

### 2.1 Importance Weighted Autoencoders

Burda et al. (2016) introduced the IWAE which maximizes an average of ELBOs over data, $\frac{1}{N}\sum_{n=1}^N \text{ELBO}_{\text{IS}}^K(\theta, \phi, x^{(n)})$, where, given number of particles $K$,

$$\text{ELBO}_{\text{IS}}^K(\theta, \phi, x) = \mathbb{E}_{Q_\phi(z_{1:K}|x)}\left[\log\left(\frac{1}{K}\sum_{k=1}^K w_k\right)\right],\ Q_\phi(z_{1:K}|x) = \prod_{k=1}^K q_\phi(z_k|x),\ w_k = \frac{p_\theta(z_k, x)}{q_\phi(z_k|x)}. \quad (1)$$

When $K = 1$, this reduces to the variational autoencoder (VAE) (Kingma & Welling, 2014; Rezende et al., 2014). Burda et al. (2016) show that $\text{ELBO}_{\text{IS}}^K(\theta, \phi, x)$ is a lower bound on $\log p_\theta(x)$ and that increasing $K$ leads to a tighter lower bound. Further, tighter lower bounds arising from increasing $K$ improve learning of the generative model, but worsen learning of the inference network (Rainforth et al., 2018), as the signal-to-noise ratio of $\theta$'s gradient estimator is $O(\sqrt{K})$ whereas $\phi$'s is $O(1/\sqrt{K})$. Moreover, poor learning of the inference network, beyond a certain point (large $K$), can actually worsen learning of the generative model as well; a finding we explore in Section 4.3.

Optimizing the IWAE objective using SGD methods requires unbiased gradient estimators of $\text{ELBO}_{\text{IS}}^K(\theta, \phi, x)$ with respect to $\theta$ and $\phi$ (Robbins & Monro, 1951). The $\theta$ gradient $\nabla_\theta \text{ELBO}_{\text{IS}}^K(\theta, \phi, x)$ is estimated by sampling $z_{1:K} \sim Q_\phi(\cdot|x)$ and evaluating $\nabla_\theta \log\left(\frac{1}{K}\sum_{k=1}^K w_k\right)$. While $\nabla_\phi \text{ELBO}_{\text{IS}}^K(\theta, \phi, x)$ is estimated similarly for models with reparameterizable latents, discrete (and other non-reparameterizable) latents require the REINFORCE gradient estimator (Williams, 1992)

$$g_{\text{REINFORCE}} = \underbrace{\log\left(\frac{1}{K}\sum_{k=1}^{K} w_k\right)\nabla_\phi \log Q_\phi(z_{1:K}|x)}_{\textcircled{1}} + \underbrace{\nabla_\phi \log\left(\frac{1}{K}\sum_{k=1}^{K} w_k\right)}_{\textcircled{2}}. \tag{2}$$

### 2.2 Continuous Relaxation and Control Variate Methods

Since the gradient estimator in Eq. (2) can often suffer from high variance, mainly due to the effect of $\textcircled{1}$, a number of approaches have been developed to ameliorate the issue. When employing continuous relaxation methods, all discrete latent variables in the model can be replaced by the Concrete (Maddison et al., 2017) or Gumbel-Softmax (Jang et al., 2017) distribution, whose gradients can be approximated using the reparameterization trick (Kingma & Welling, 2014; Rezende et al., 2014). The main practical difficulty here is tuning the temperature parameter: low temperatures reduce the gradient estimator bias, but rapidly increase its variance. Moreover, since such relaxations are defined on a simplex, applying them to models that exhibit branching requires the computationally expensive evaluation of *all* (potentially exponentially many) paths in the generative model.

Variational inference for Monte Carlo objectives (VIMCO) (Mnih & Rezende, 2016) is a method that doesn't require designing an explicit control variate, as it exploits the particle set obtained in IWAE. It replaces $\textcircled{1}$ with

$$g^{\textcircled{1}}_{\text{VIMCO}} = \sum_{\ell=1}^{K}\left(\underbrace{\log\left(\frac{1}{K}\sum_{k=1}^{K} w_k\right)}_{A} - \underbrace{\log\left(\frac{1}{K}\left(e^{\frac{1}{K-1}\sum_{k\neq\ell}\log w_k} + \sum_{k\neq\ell} w_k\right)\right)}_{B}\right)\nabla_\phi \log q_\phi(z_\ell|x),$$

where term $B$ is independent of $z_\ell$ and highly correlated with term $A$.

Finally, assuming $z_k$ is a discrete random variable with $C$ categories[1] , REBAR (Tucker et al., 2017) and RELAX (Grathwohl et al., 2018) improve on the methods of Mnih & Gregor (2014) and Gu et al. (2016), and replaces $\textcircled{1}$ with

$$g^{\textcircled{1}}_{\text{RELAX}} = \left(\log\left(\frac{1}{K}\sum_{k=1}^{K} w_k\right) - c_\rho(\tilde{g}_{1:K})\right)\nabla_\phi \log Q_\phi(z_{1:K}|x) + \nabla_\phi c_\rho(g_{1:K}) - \nabla_\phi c_\rho(\tilde{g}_{1:K}),$$

where $g_k$ is a $C$-dimensional vector of reparameterized Gumbel random variates, $z_k$ is a one-hot argmax function of $g_k$, and $\tilde{g}_k$ is a vector of reparameterized conditional Gumbel random variates conditioned on $z_k$. The conditional Gumbel random variates are a form of Rao-Blackwellization used to reduce variance. The control variate $c_\rho$, parameterized by $\rho$, is optimized to minimize the gradient variance estimates concurrently with the main ELBO optimization, leading to state-of-the-art performance on, for example, sigmoid belief networks (Neal, 1992). The main practical difficulty in using this method is choosing a suitable family of $c_\rho$, as some choices lead to higher variance despite the concurrent gradient variance minimization. Moreover, the objective for the concurrent optimization requires evaluating a Jacobian-vector product that induces an overhead of $O(D_\phi D_\rho)$ where $D_\phi, D_\rho$ are number of inference network and control-variate parameters respectively.

## 3 Revisiting Reweighted Wake-Sleep

Reweighted wake-sleep (RWS) (Bornschein & Bengio, 2015) comes from another family of algorithms (Hinton et al., 1995; Dayan et al., 1995) for learning deep generative models, eschewing a single objective over parameters $\theta$ and $\phi$ in favour of individual objectives for each. We review the RWS algorithm and discuss its advantages and disadvantages.

[1] The assumption is needed only for notational convenience. However, using more structured latents leads to difficulties in picking the control-variate architecture.

### 3.1 REWEIGHTED WAKE-SLEEP

Reweighted wake-sleep (RWS) (Bornschein & Bengio, 2015) is an extension of the wake-sleep algorithm (Hinton et al., 1995; Dayan et al., 1995) both of which, like IWAE, jointly learn a generative model and an inference network given data. While IWAE targets a single objective, RWS alternates between objectives, updating the generative model parameters $\theta$ using a *wake-phase $\theta$ update* and the inference network parameters $\phi$ using either a *sleep-* or a *wake-phase $\phi$ update* (or both).

**Wake-phase $\theta$ update.** Given a current value of $\phi$, $\theta$ is updated using an unbiased estimate of $\nabla_\theta \left( \frac{1}{N} \sum_{n=1}^{N} \text{ELBO}_{\text{IS}}^{K}(\theta, \phi, x^{(n)}) \right)$, which can be obtained directly without needing reparameterization or control variates as the sampling distribution $Q_\phi(\cdot|x)$ is independent of $\theta$.[2]

**Sleep-phase $\phi$ update.** Here, $\phi$ is updated to maximize the negative Kullback-Leibler (KL) divergence between the posteriors under the generative model and the inference network, averaged over the data distribution of the current generative model

$$\mathbb{E}_{p_\theta(x)}[-D_{\text{KL}}(p_\theta(z|x), q_\phi(z|x))] = \mathbb{E}_{p_\theta(z,x)}[\log q_\phi(z|x) - \log p_\theta(z|x)]. \tag{3}$$

The gradient of this objective is $\mathbb{E}_{p_\theta(z,x)}[\nabla_\phi \log q_\phi(z|x)]$ and can be estimated by sampling $z, x$ from the generative model $p_\theta(z, x)$ and evaluating $\nabla_\phi \log q_\phi(z|x)$. The variance of such an estimator can be reduced at a standard Monte Carlo rate by increasing the number of samples of $z, x$.

**Wake-phase $\phi$ update.** Here, $\phi$ is updated to maximize the negative KL divergence between the posteriors under the generative model and the inference network, averaged over the true data distribution

$$\mathbb{E}_{p(x)}[-D_{\text{KL}}(p_\theta(z|x), q_\phi(z|x))] = \mathbb{E}_{p(x)}[\mathbb{E}_{p_\theta(z|x)}[\log q_\phi(z|x) - \log p_\theta(z|x)]]. \tag{4}$$

The outer expectation of the gradient $\mathbb{E}_{p(x)}[\mathbb{E}_{p_\theta(z|x)}[\nabla_\phi \log q_\phi(z|x)]]$ can be estimated using a single sample $x$ from the true data distribution $p(x)$, given which, the inner expectation can be estimated using a self-normalized importance sampler with $K$ particles using $q_\phi(z|x)$ as the proposal distribution. This results in the following estimator

$$\sum_{k=1}^{K} \left( w_k / \sum_{\ell=1}^{K} w_\ell \right) \nabla_\phi \log q_\phi(z_k|x), \tag{5}$$

where $x \sim p(x)$, $z_k \sim q_\phi(z_k|x)$, and $w_k = p_\theta(z_k, x)/q_\phi(z_k|x)$, in a similar fashion to Eq. (1). Note that equation 5 is negative of the second term of the REINFORCE estimator of the IWAE ELBO in equation 2. The crucial difference of the *wake-phase $\phi$ update* to the *sleep-phase $\phi$ update* is that the expectation in Eq. (4) is over the true data distribution $p(x)$ unlike the expectation in Eq. (3) which is under the current model distribution $p_\theta(x)$. The former is desirable from the perspective of amortizing inference over data from $p(x)$. Although the estimator in Eq. (5) is biased, this bias decreases as $K$ increases.

While Bornschein & Bengio (2015) refer to the use of the *sleep-phase $\phi$ update* followed by an equally-weighted mixture of the *wake-phase* and *sleep-phase* updates of $\phi$, we here use RWS to refer to the variant that employs just the *wake-phase $\phi$ update*, and not the mixture. The rationale for our preference will be made clear from the empirical evaluations (Section 4).

### 3.2 ADVANTAGES OF REWEIGHTED WAKE-SLEEP

While the gradient update of $\theta$ targets the same objective as IWAE, the gradient update of $\phi$ targets the objective in Eq. (3) in the sleep case and Eq. (4) in the wake case. This leads to two advantages for RWS over IWAE. First, since we don't need to use REINFORCE, using RWS leads to much lower variance of gradient estimators of $\phi$. Second, the $\phi$ updates in RWS directly target minimization of the expected KL divergences from true to approximate posteriors. Increasing the computational budget (using more Monte Carlo samples in the *sleep-phase $\phi$ update* case and higher number of

[2] We assume that the deterministic mappings induced by the parameters $\theta, \phi$ are themselves differentiable, such that they are amenable to gradient-based learning.

particles $K$ in the *wake-phase $\phi$ update*) results in a better estimator of these expected KL divergences. This is different to IWAE, where optimizing $\text{ELBO}_{\text{IS}}^{K}$ targets a KL divergence on an extended sampling space (Le et al., 2018) which for $K > 1$ doesn't correspond to a KL divergence between true and approximate posteriors (in any order). Consequently, increasing number of particles in IWAE leads to worse learning of inference networks (Rainforth et al., 2018).

### 3.3 Disadvantages of Reweighted Wake-Sleep

The objective of the *sleep-phase $\phi$ update* in equation 3 is an expectation of KL under the current model distribution $p_\theta(x)$ rather than the true one $p(x)$. This makes the *sleep-phase $\phi$ update* suboptimal since the inference network must always follow a "doubly moving" target (both $p_\theta(x)$ and $p_\theta(z|x)$ change during the optimization).

## 4 Experiments

The IWAE and RWS algorithms have primarily been applied to problems with continuous latent variables and/or discrete latent variables that do not actually induce branching such as sigmoid belief networks (Neal, 1992). The purpose of the following experiments is to compare RWS to IWAE used alongside the control variate and continuous relaxation methods described in Section 3 on models with conditional branching, where, as we will show, the various control-variate schemes underperform in relation to RWS. In several ways, including ELBOs achieved and average distance between true and amortized posteriors, we empirically demonstrate that increasing the number of particles $K$ hurts learning in IWAE but improves learning in RWS.

The first experiment, using the deep generative model from Attend, Infer, Repeat (AIR) (Eslami et al., 2016), demonstrates better learning of the generative model in a model containing both discrete latent variables used for branching as well as continuous latent variables in a complex visual data domain (Section 4.1). The next experiment on MNIST (Section 4.2) does so in a model with continuous latent variables. Finally, a GMM experiment (Section 4.3) serves as a pedagogical example to understand sources of advantage for RWS in more detail.

Notationally, the different variants of RWS will be referred to as wake-sleep (WS) and wake-wake (WW). The *wake-phase $\theta$ update* is always used. We refer to using it in conjunction with the *sleep-phase $\phi$ update* as WS and using it in conjunction with the *wake-phase $\phi$ update* as WW. We tried using both *wake-* and *sleep-phase $\phi$ updates* however, in addition to doubling the amount of sampling, found that doing so only improves performance on the continuous latent variable model. The number of particles $K$ used for the *wake-phase $\theta$* and *$\phi$ updates* will always be specified. The *sleep phase $\phi$ update* will also use $K$ samples from $p_\theta(z, x)$.

### 4.1 Attend, Infer, Repeat

First, we evaluate WW and VIMCO on AIR (Eslami et al., 2016), a structured deep generative model with both discrete and continuous latent variables. AIR uses the discrete variable to decide how many continuous variables are necessary to explain an image (see supplementary material for details). The sequential inference procedure of AIR poses a difficult problem, since it implies a sequential decision process with possible branching. See (Eslami et al., 2016) for the details of the model (see supplementary material for the model in our notation).

We set the maximum number of inference steps in AIR to three and we train it on images of size $50 \times 50$ with zero, one or two MNIST digits. The training and testing data sets consist of 60000 and 10000 images, respectively, which are generated from the respective MNIST train/test datasets. Unlike AIR, which used Gaussian likelihood with fixed standard deviation and continuous inputs (i.e., input $\mathbf{x} \in [0, 1]^{50\times50}$), we use a Bernoulli likelihood and binarized data; the stochastic binarization is the same as in Burda et al. (2016). This choice is motivated by initial experiments, which have shown that the original setup is detrimental to the sleep phase of WS – samples from the generative model did not look similar to true data even after the training has converged. Training is performed over two million iterations by RmsProp (Tieleman & Hinton, 2012) with the learning rate of $10^{-5}$, which is divided by three after 400k and 1000k training iterations. We set the glimpse size to $20 \times 20$.

We first evaluate the generative model via the average test log marginal where each log marginal is estimated by a one-sample, 5000-particle IWAE estimate. The inference network is then evaluated via the average test KL from the inference network to the posterior under the current model where each $D_{\text{KL}}(q_\phi(z|x), p_\theta(z|x))$ is estimated as a difference between the log marginal estimate above and a 5000-sample, one-particle IWAE estimate. Note that this KL estimate is merely a proxy to the desired KL from the inference network to the posterior under the *true* model.

This experiment confirms (Fig. 1) that increasing number of particles hurts learning in VIMCO but improves learning in WW. Increasing $K$ improves WW monotonically but VIMCO only up to a point. WW also results in significantly lower variance and better inference networks than VIMCO.

### 4.2 Continuous Latent Variable Model

RWS has typically only been considered for learning models with discrete latent variables. In order to gauge its applicability to a wider class of problems, we evaluate it on a variational autoencoder with normally distributed latent variables for MNIST. To do this, we use the training procedure and the model with a single stochastic layer of Burda et al. (2016) and their stochastic binarization of data.

To train the model, we use the Adam optimizer (Kingma & Ba, 2015) with the learning rate of $10^{-3}$ annealed over the course of about 3000 epochs (9840000 iterations) to the value of $10^{-4}$ with the batch size of 32. We evaluate performance in the same way as we evaluate the AIR model. Additionally, we evaluate the number of active latent units (Burda et al., 2016).

Table 1: Number of active latent units evaluated on the test set.

| $K$ | IWAE | WW |
|---|---|---|
| 8 | $23.00 \pm 0.00$ | $23.75 \pm 0.83$ |
| 16 | $23.75 \pm 0.43$ | $25.00 \pm 0.00$ |
| 32 | $24.50 \pm 0.50$ | $25.00 \pm 0.00$ |
| 64 | $25.00 \pm 0.00$ | $26.50 \pm 0.50$ |

This experiment confirms (Fig. 2 and Table 1) that increasing number of particles hurts learning in IWAE but improves learning in WW. Increasing $K$ does improve the marginal likelihood attained by WW and IWAE with reparameterization. However, the latter learns a better model only up to a point ($K = 128$) — further increase in the number of particles has diminishing returns. WW also results in better inference networks than IWAE as showed by the KL plot on the right of Fig. 2.

### 4.3 Why?

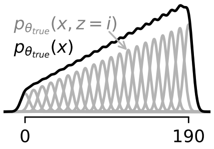

Figure 3: True GMM.

To see what is going on, we study a GMM which branches on a discrete latent variable to select cluster assignments. The generative model and inference network are defined as

$$p_\theta(z) = \text{Categorical}(z|\text{softmax}(\theta)), \;\; p(x|z) = \text{Normal}(x|\mu_z, \sigma_z^2),$$
$$q_\phi(z|x) = \text{Categorical}(z|\text{softmax}(\eta_\phi(x))),$$

where $z \in \{0, \dots, C-1\}$, $C$ is the number of clusters and $\mu_c, \sigma_c^2$ are fixed to $\mu_c = 10c$ and $\sigma_c^2 = 5^2$. The generative model parameters are $\theta \in \mathbb{R}^C$. The inference network consists of a multilayer perceptron $\eta_\phi : \mathbb{R} \to \mathbb{R}^C$, with the 1-16-$C$ architecture and the $\tanh$ nonlinearity, parameterized by $\phi$. The chosen family of inference networks is empirically expressive enough to capture the posterior

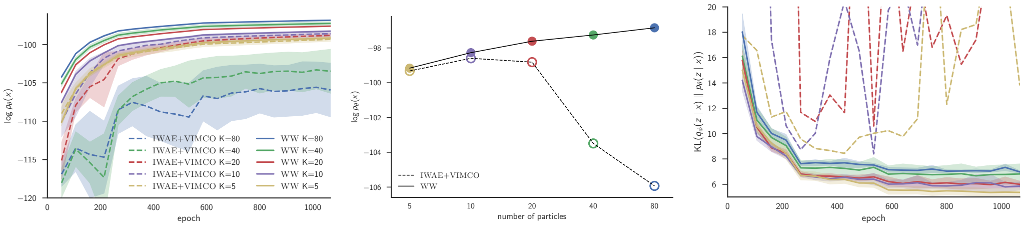

Figure 1: Training of AIR. *(Left)* Training curves: training with VIMCO leads to larger variance in training than WW. *(Middle)* Log evidence values at the end of training: increasing number of particles improves WW monotonically but improves VIMCO only up to a point ($K = 10$ is the best). *(Right)* WW results in significantly lower variance and better inference networks than VIMCO. Note that KL is between the inference network and the *current* generative model.

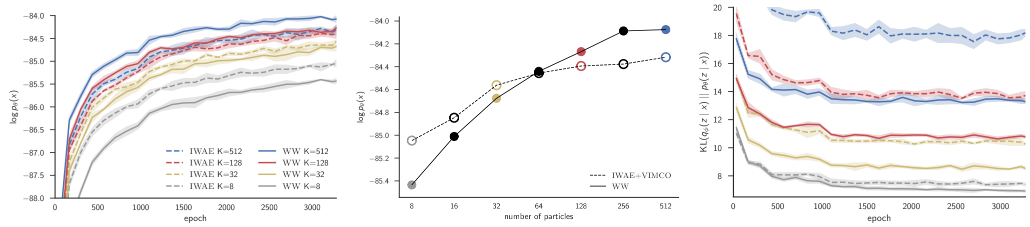

Figure 2: Training of a continuous latent variable model on MNIST. *(Left)* Training curves: the variance in training is comparable between WW and IWAE with reparameterization. *(Middle)* Log evidence values at the end of training: increasing number of particles improves both WW and IWAE with reparameterization; the latter learns a better model only up to a point ($K = 128$) and suffers from diminishing returns afterwards. *(Right)* WW results in better inference networks than IWAE. Note that KL is between the inference network and the *current* generative model.

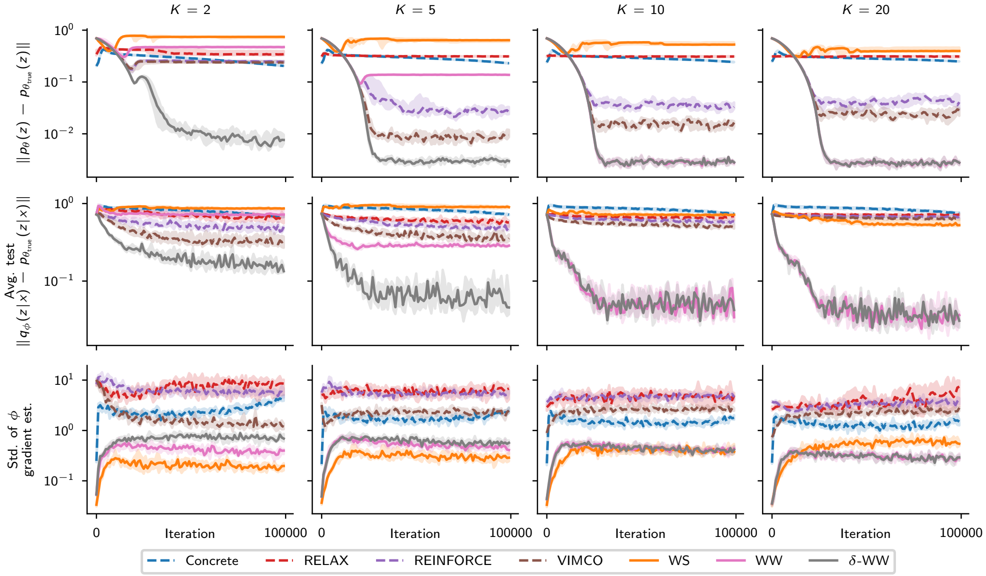

Figure 4: GMM training. Median and interquartile ranges from 10 repeats shown. *(Top)* Quality of the generative model: WS and WW improve with larger particle budget thanks to lower variance and lower bias estimators of the gradient respectively. IWAE methods suffer with a larger particle budget (Rainforth et al., 2018). WS performs the worst as a consequence of computing the expected KL under model distribution $p_\theta(x)$ equation 3 instead of the true data distribution $p(x)$ as with WW equation 4. WW suffers from zero-forcing (described in text) in low-particle regimes, but learns the best model fastest in the many-particle regime; $\delta$-WW additionally learns well in the low-particle regime. *(Middle)* The quality of the inference network develops identically to that of the generative model. *(Bottom)* WW and WS have lower-variance gradient estimators of $\phi$ than IWAE, as they don't include the high-variance term ① in equation 2. This is a necessary, but not sufficient, condition for efficient learning with other important factors being gradient direction and the ability to escape local optima.

under the true model. The true model is set to $p_{\theta_{\text{true}}}(x)$ where $\text{softmax}(\theta_{\text{true}})_c = (c+5)/\sum_{i=1}^{C}(i+5)$ ($c = 0, \dots, C-1$), i.e. the mixture probabilities are linearly increasing with the $z$ (Fig. 3). We fix the mixture parameters in order to study the important features of the problem at hand in isolation.

IWAE was trained with REINFORCE, RELAX, VIMCO and the Concrete distribution. We also train using WS and WW. We fix $C = 20$ and increase number of particles from $K = 2$ to 20. We use the Adam optimizer with the learning rate $10^{-3}$ and default $\beta$ parameters. At each iteration, a batch of 100 data points is generated from the true model to train. Having searched over several temperature schedules for the Concrete distribution, we use the one with the lowest trainable terminal temperature (linearly annealing from 3 to 0.5). We found that using the control variate $c_\rho(g_{1:K}) = \frac{1}{K}\sum_{k=1}^{K} \text{MLP}_\rho([x, g_k])$, where the architecture of the multilayer perceptron (MLP) is

$(1 + C)$-16-16-1 (with $\tanh$ nonlinearity) led to most stable training (see supplementary material for more details).

We evaluate the generative model, inference network and the variance of the gradient estimator of $\phi$. The generative model is evaluated via the L2 distance between the probability mass functions (PMFs) of its prior and true prior as $\|\mathrm{softmax}(\theta) - \mathrm{softmax}(\theta_{\text{true}})\|$. The inference network is evaluated via the L2 distance between PMFs of the current and true posteriors, averaged over a fixed set ($M = 100$) of observations $(x_{\text{test}}^{(m)})_{m=1}^{M}$ from the true model: $\frac{1}{M}\sum_{m=1}^{M}\|q_\phi(z|x_{\text{test}}^{(m)}) - p_{\theta_{\text{true}}}(z|x_{\text{test}}^{(m)})\|$. Finally, $\phi$'s gradient-estimator standard deviation is given by $\frac{1}{D_\phi}\sum_{d=1}^{D_\phi}\mathrm{std}(\hat{g}_d)$ where $\hat{g}_d$ is the $d$th element of one of $\phi$'s gradient estimators (e.g. equation 2 for REINFORCE) and $\mathrm{std}(\cdot)$ is estimated using $10$ samples.

Here, we demonstrate that using WS and WW with larger particle budgets leads to a better inference networks whereas this is not the case for IWAE methods (Fig. 4, Middle). Recall that the former is because using more samples to estimate the gradient of the sleep $\phi$ objective equation 3 for WS reduces variance at a standard Monte Carlo rate and that using more particles in equation 5 to estimate the gradient of the wake $\phi$ objective results in a lower bias. The latter is because using more particles results in the signal-to-noise of IWAE's $\phi$ gradient estimator to drop at the rate $O(1/\sqrt{K})$ (Rainforth et al., 2018).

Learning of the generative model, as a consequence of inference-network learning, is also better for WS and WW, but worse for IWAE methods when the particle budget is increased. This is because the the $\theta$ gradient estimator (common to all methods), $\nabla_\theta\, \mathrm{ELBO}_{\text{IS}}^{K}(\theta, \phi, x)$ can be seen as an importance sampling estimator whose quality is tied to the proposal distribution (inference network).

WW and WS have lower variance gradient estimators than IWAE, even if used with control-variate and continuous-relaxation methods. This is because $\phi$'s gradient estimators for WW and WS don't include the high-variance term ① in equation 2. This is a necessary but not sufficient condition for efficient learning with other important factors being gradient direction and the ability to escape local optima (explored below). Employing the Concrete distribution gives low-variance gradients for $\phi$ to begin with, but the model learns poorly due to the high gradients bias (due to high temperature hyperparameter).

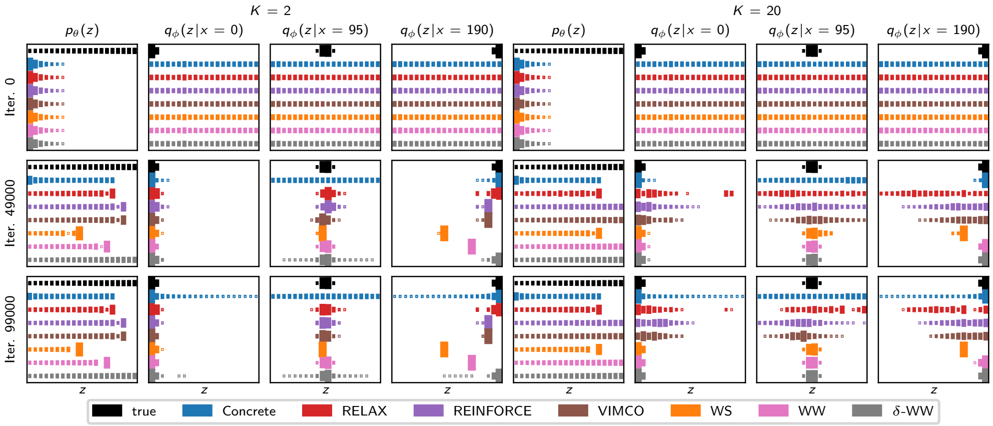

Figure 5: Generative model and inference network during GMM training shown as Hinton diagrams where areas are proportional to probability. $z$ goes from 0 to 19, left to right. Rows correspond to start, middle and end of optimization. *(Left half)* Learning with few particles leads to the zero-forcing (described in text) of the inference network (shown as conditional PMF given different $x$) and the generative model (first column of each half) for all methods except $\delta$-WW. Concrete distribution fails. *(Right half)* Learning with many particles leads to zero-forcing only for WS; WW and $\delta$-WW succeed where IWAE fails, learning a suboptimal final generative model.

We now describe a failure mode, that affects WS, WW, VIMCO and REINFORCE, which we will refer to as *zero-forcing*. It is best illustrated by inspecting the generative model and the inference network as the training progresses, focusing on the low-particle ($K = 2$) regime (Fig. 5). For WS, the generative model $p_\theta(z)$ peaks at $z = 9$ and puts zero mass for $z > 9$; the inference network $q_\phi(z|x)$ becomes the posterior for this model which, in this model, also has support at most $\{0, \dots, 9\}$ for all $x$. This is a local optimum for WS because (i) the inference network already approximates the posterior of the model $p_\theta(z, x)$ well, and (ii) the generative model $p_\theta(z)$, being trained using samples from $q_\phi(z|x)$, has no samples outside of its current support ($\{0, \dots, 9\}$). Similar failure mode occurs for WW and VIMCO/REINFORCE although the support of the locally optimal $p_\theta(z)$ is larger ($\{0, \dots, 14\}$ and $\{0, \dots, 17\}$ respectively).

While this failure mode is a particular feature of the GMM, we hypothesize that WS and WW suffer from it more, as they alternate between two different objectives for optimizing $\theta$ and $\phi$. WS attempts to amortize inference for the current model distribution $p_\theta(x)$ which reinforces the coupling between the generative model and the inference network, making it easier to get stuck in a local optimum. WW with few particles (say $K = 1$) on the other hand, results in a highly-biased gradient estimator equation 5 that samples $z$ from $q_\phi(\cdot|x)$ and evaluates $\nabla_\phi \log q_\phi(z|x)$; this encourages the inference network to concentrate mass. This behavior is not seen in WW with many particles where it is the best algorithm at learning both a good generative model and inference network (Fig. 4; Fig. 5, right).

We propose a simple extension of WW, denoted $\delta$-WW, that mitigates this shortcoming by changing the proposal of the self-normalized importance sampling estimator in Eq. (5) to $q_{\phi,\delta}(z|x) = (1-\delta)q_\phi(z|x) + \delta\mathrm{Uniform}(z)$. We use $\delta = 0.2$, noting that the method is robust to a range of values. Using a different proposal than the inference network $q_\phi(z|x)$ means that using the low-particle estimator in Eq. (5) no longer leads to zero-forcing. This is known as defensive importance sampling (Hesterberg, 1995), and is used to better estimate integrands that have long tails using short-tailed proposals. Using $\delta$-WW outperforms all other algorithms in learning both the generative model and the inference network.

## 5 Conclusion

Our experiments suggest that RWS learns both better generative models and inference networks in models that involve discrete latent variables, while performing just as well as state-of-the-art on continuous-variable models as well. The AIR experiment (Section 4.1) shows that the trained inference networks are unusable when trained with high number of particles. Moreover, the MNIST experiment (Section 4.2) suggests that RWS is competitive even on models with continuous latent variables, especially for high number of particles where IWAE ELBO starts suffering from worse inference networks. The GMM experiment (Section 4.3) illustrates that this is at least at least in part due to a lower variance gradient estimator for the inference network and the fact that for RWS—unlike the case of optimizing IWAE ELBO (Rainforth et al., 2018)—increasing number of particles actually improves the inference network. In the low-particle regime, the GMM suffers from zero-forcing of the generative model and the inference network, which is ameliorated using defensive RWS. Finally, all experiments show that, beyond a certain point, increasing the particle budget starts to affect the quality of the generative model for IWAE ELBO whereas this is not the case for RWS. As a consequence of our findings, we recommend reconsidering using RWS for learning deep generative models, especially those containing discrete latent variables that induce branching.

## References

Ryan Adams, Hanna Wallach, and Zoubin Ghahramani. Learning the structure of deep sparse graphical models. In *International Conference on Artificial Intelligence and Statistics*, 2010.

David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

Jörg Bornschein and Yoshua Bengio. Reweighted wake-sleep. In *International Conference on Learning Representations*, 2015.

Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. In *International Conference on Learning Representations*, 2016.

Nick Chater and Christopher D Manning. Probabilistic models of language processing and acquisition. *Trends in cognitive sciences*, 10(7):335–344, 2006.

Peter Dayan, Geoffrey E Hinton, Radford M Neal, and Richard S Zemel. The Helmholtz machine. *Neural computation*, 7(5):889–904, 1995.

S. M. Ali Eslami, Nicolas Heess, Theophane Weber, Yuval Tassa, David Szepesvari, Koray Kavukcuoglu, and Geoffrey E. Hinton. Attend, infer, repeat: Fast scene understanding with generative models. In *Advances in Neural Information Processing Systems*, 2016.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Will Grathwohl, Dami Choi, Yuhuai Wu, Geoff Roeder, and David Duvenaud. Backpropagation through the void: Optimizing control variates for black-box gradient estimation. In *International Conference on Learning Representations*, 2018.

Alex Graves, Greg Wayne, and Ivo Danihelka. Neural Turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio Gómez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, et al. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538 (7626):471, 2016.

Edward Grefenstette, Karl Moritz Hermann, Mustafa Suleyman, and Phil Blunsom. Learning to transduce with unbounded memory. In *Advances in Neural Information Processing Systems*, pp. 1828–1836, 2015.

Shixiang Gu, Sergey Levine, Ilya Sutskever, and Andriy Mnih. Muprop: Unbiased backpropagation for stochastic neural networks. In *International Conference on Learning Representations*, 2016.

Tim Hesterberg. Weighted average importance sampling and defensive mixture distributions. *Technometrics*, 37(2):185–194, 1995.

Geoffrey E Hinton, Peter Dayan, Brendan J Frey, and Radford M Neal. The "wake-sleep" algorithm for unsupervised neural networks. *Science*, 268(5214):1158–1161, 1995.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with Gumbel-softmax. In *International Conference on Learning Representations*, 2017.

Biing Hwang Juang and Laurence R Rabiner. Hidden markov models for speech recognition. *Technometrics*, 33(3):251–272, 1991.

Charles Kemp, Joshua B Tenenbaum, Thomas L Griffiths, Takeshi Yamada, and Naonori Ueda. Learning systems of concepts with an infinite relational model. 2006.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014.

Brenden M Lake, Neil D Lawrence, and Joshua B Tenenbaum. The emergence of organizing structure in conceptual representation. *Cognitive science*, 2018.

Tuan Anh Le, Atilim Gunes Baydin, and Frank Wood. Inference compilation and universal probabilistic programming. In *International Conference on Artificial Intelligence and Statistics*, 2017.

Tuan Anh Le, Maximilian Igl, Tom Rainforth, Tom Jin, and Frank Wood. Auto-encoding sequential Monte Carlo. In *International Conference on Learning Representations*, 2018.

Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, 2017.

Andriy Mnih and Karol Gregor. Neural variational inference and learning in belief networks. In *International Conference on Machine Learning*, pp. 1791–1799, 2014.

Andriy Mnih and Danilo Rezende. Variational inference for Monte Carlo objectives. In *International Conference on Machine Learning*, pp. 2188–2196, 2016.

Shakir Mohamed and Balaji Lakshminarayanan. Learning in implicit generative models. *arXiv preprint arXiv:1610.03483*, 2016.

Radford M Neal. Connectionist learning of belief networks. *Artificial intelligence*, 56(1):71–113, 1992.

Willie Neiswanger, Frank Wood, and Eric Xing. The dependent Dirichlet process mixture of objects for detection-free tracking and object modeling. In *Artificial Intelligence and Statistics*, pp. 660–668, 2014.

Brooks Paige and Frank Wood. Inference networks for sequential monte carlo in graphical models. In *International Conference on Machine Learning*, pp. 3040–3049, 2016.

Tom Rainforth, Adam R Kosiorek, Tuan Anh Le, Chris J Maddison, Maximilian Igl, Frank Wood, and Yee Whye Teh. Tighter variational bounds are not necessarily better. In *International Conference on Machine Learning*, 2018.

Carl Edward Rasmussen. The infinite Gaussian mixture model. In *Advances in neural information processing systems*, pp. 554–560, 2000.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, 2014.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pp. 400–407, 1951.

T. Tieleman and G. Hinton. Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 2012.

George Tucker, Andriy Mnih, Chris J Maddison, John Lawson, and Jascha Sohl-Dickstein. Rebar: Low-variance, unbiased gradient estimates for discrete latent variable models. In *Advances in Neural Information Processing Systems*, pp. 2624–2633, 2017.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, pp. 2048–2057, 2015.