DVAE++: Discrete Variational Autoencoders with Overlapping Transformations

Arash Vahdat [1] **William G. Macready** [1] **Zhengbing Bian** [1] **Amir Khoshaman** [1] **Evgeny Andriyash** [1]

Abstract

Training of discrete latent variable models remains challenging because passing gradient information through discrete units is difficult. We propose a new class of smoothing transformations based on a mixture of two overlapping distributions, and show that the proposed transformation can be used for training binary latent models with either directed or undirected priors. We derive a new variational bound to efficiently train with Boltzmann machine priors. Using this bound, we develop DVAE++, a generative model with a global discrete prior and a hierarchy of convolutional continuous variables. Experiments on several benchmarks show that overlapping transformations outperform other recent continuous relaxations of discrete latent variables including Gumbel-Softmax (Maddison et al., 2016; Jang et al., 2016), and discrete variational autoencoders (Rolfe, 2016).

1. Introduction

Recent years have seen rapid progress in generative modeling made possible by advances in deep learning and stochastic variational inference. The reparameterization trick (Kingma & Welling, 2014; Rezende et al., 2014) has made stochastic variational inference efficient by providing lower-variance gradient estimates. However, reparameterization, as originally proposed, does not easily extend to semi-supervised learning, binary latent attribute models, topic modeling, variational memory addressing, hard attention models, or clustering, which require discrete latent-variables.

Continuous relaxations have been proposed for accommodating discrete variables in variational inference (Maddison et al., 2016; Jang et al., 2016; Rolfe, 2016). The Gumbel-Softmax technique (Maddison et al., 2016; Jang et al., 2016) defines a temperature-based continuous distribution that in the zero-temperature limit converges to a discrete distribution. However, it is limited to categorical distributions and does not scale to multivariate models such as Boltzmann machines (BM). The approach presented in (Rolfe, 2016) can train models with BM priors but requires careful handling of the gradients during training.

We propose a new class of smoothing transformations for relaxing binary latent variables. The method relies on two distributions with overlapping support that in the zero temperature limit converge to a Bernoulli distribution. We present two variants of smoothing transformations using a mixture of exponential and a mixture of logistic distributions.

We demonstrate that overlapping transformations can be used to train discrete directed latent models as in (Maddison et al., 2016; Jang et al., 2016), *and* models with BMs in their prior as in (Rolfe, 2016). In the case of BM priors, we show that the Kullback-Leibler (KL) contribution to the variational bound can be approximated using an analytic expression that can be optimized using automatic differentiation without requiring the special treatment of gradients in (Rolfe, 2016).

Using this analytic bound, we develop a new variational autoencoder (VAE) architecture called DVAE++, which uses a BM prior to model discontinuous latent factors such as object categories or scene configuration in images. DVAE++ is inspired by (Rolfe, 2016) and includes continuous local latent variables to model locally smooth features in the data. DVAE++ achieves comparable results to the state-of-the-art techniques on several datasets and captures semantically meaningful discrete aspects of the data. We show that even when all continuous latent variables are removed, DVAE++ still attains near state-of-the-art generative likelihoods.

1.1. Related Work

Training of models with discrete latent variables $\boldsymbol{z}$ requires low-variance estimates of gradients of the form $\nabla_\phi \mathbb{E}_{q_\phi(\boldsymbol{z})}[f(\boldsymbol{z})]$. Only when $\boldsymbol{z}$ has a modest number of configurations (as in semi-supervised learning (Kingma et al., 2014) or semi-supervised generation (Maaløe et al., 2017))

[1]Quadrant.ai, D-Wave Systems Inc., Burnaby, BC, Canada. Correspondence to: Arash Vahdat <arash@quadrant.ai>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

can the gradient of the expectation be decomposed into a summation over configurations.

The REINFORCE technique (Williams, 1992) is a more scalable method that migrates the gradient inside the expectation: $\nabla_\phi \mathbb{E}_{q_\phi(\boldsymbol{z})} f(\boldsymbol{z}) = \mathbb{E}_{q_\phi(\boldsymbol{z})}[f(\boldsymbol{z}) \nabla_\phi \log q_\phi(\boldsymbol{z})]$. Although the REINFORCE estimate is unbiased, it suffers from high variance and carefully designed "control variates" are required to make it practical. Several works use this technique and differ in their choices of the control variates. NVIL (Mnih & Gregor, 2014) uses a running average of the function, $f(\boldsymbol{z})$, and an input-dependent *baseline*. VIMCO (Mnih & Rezende, 2016) is a multi-sample version of NVIL that has baselines tailored for each sample based on all the other samples. MuProp (Gu et al., 2015) and DARN (Gregor et al., 2013) are two other REINFORCE-based methods (with non-zero biases) that use a Taylor expansion of the function $f(\boldsymbol{z})$ to create control variates.

To address the high variance of REINFORCE, other work strives to make discrete variables compatible with the reparametrization technique. A primitive form arises from estimating the discrete variables by a continuous function during back-propagation. For instance, in the case of Bernoulli distribution, the latent variables can be approximated by their mean value. This approach is called the *straight-through (ST) estimator* (Bengio et al., 2013). Another way to make discrete variables compatible with the reparametrization is to relax them into a continuous distribution. Concrete (Maddison et al., 2016) or Gumbel-Softmax (Jang et al., 2016) adopt this strategy by adding Gumbel noise to the logits of a softmax function with a temperature hyperparameter. A slope-annealed version of the ST estimator is proposed by (Chung et al., 2016) and is equivalent to the Gumbel-Softmax approach for binary variables. REBAR (Tucker et al., 2017) is a recent method that blends REINFORCE with Concrete to synthesize control variates. (Rolfe, 2016) pairs discrete variables with auxiliary continuous variables and marginalizes out the discrete variables.

Both overlapping transformations and Gumbel-based approaches offer smoothing through non-zero temperature; however, overlapping transformations offer additional freedom through the choice of the mixture distributions.

2. Background

Let $\boldsymbol{x}$ represent observed random variables and $\boldsymbol{z}$ latent variables. The joint distribution over these variables is defined by the generative model $p(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{z})p(\boldsymbol{x}|\boldsymbol{z})$, where $p(\boldsymbol{z})$ is a prior distribution and $p(\boldsymbol{x}|\boldsymbol{z})$ is a probabilistic decoder. Given a dataset $\boldsymbol{X} = \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(N)}\}$, the parameters of the model are trained by maximizing the log-likelihood:

$$\log p(\boldsymbol{X}) = \sum_{i=1}^{N} \log p(\boldsymbol{x}^{(i)}).$$

Typically, computing $\log p(\boldsymbol{x})$ requires an intractable marginalization over the latent variables $\boldsymbol{z}$. To address this problem, the VAE (Kingma & Welling, 2014) introduces an inference model or probabilistic encoder $q(\boldsymbol{z}|\boldsymbol{x})$ that infers latent variables for each observation. In the VAE, instead of the maximizing the marginal log-likelihood, a variational lower bound (ELBO) is maximized:

$$\log p(\boldsymbol{x}) \geq \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}\big[\log p(\boldsymbol{x}|\boldsymbol{z})\big] - \mathrm{KL}\big(q(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z})\big). \quad (1)$$

The gradient of this objective is computed for the parameters of both the encoder and decoder using the reparameterization trick. With reparametrization, the expectation with respect to $q(\boldsymbol{z}|\boldsymbol{x})$ in Eq. (1) is replaced with an expectation with respect to a known optimization-parameter-independent base distribution and a differentiable transformation from the base distribution to $q(\boldsymbol{z}|\boldsymbol{x})$. This transformation may be a scale-shift transformation, in the case of Gaussian base distributions, or rely on the inverse cumulative distribution function (CDF) in the general case. Following the law of the unconscious statistician, the gradient is then estimated using samples from the base distribution.

Unfortunately, the reparameterization trick cannot be applied directly to the discrete latent variables because there is no differentiable transformation that maps a base distribution to a discrete distribution. Current remedies address this difficulty using a continuous relaxation of the discrete latent variables (Maddison et al., 2016; Jang et al., 2016). The discrete variational autoencoder (DVAE) (Rolfe, 2016) develops a different approach which applies the reparameterization trick to a marginal distribution constructed by pairing each discrete variable with an auxiliary continuous random variable.

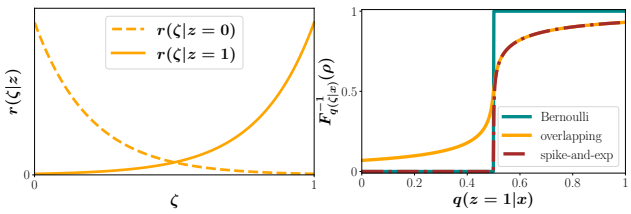
For example, let $z \in \{0, 1\}$ represent a binary random variable with the probability mass function $q(z|x)$. A smoothing transformation is defined using spike-and-exponential transformation $r(\zeta|z)$, where $r(\zeta|z = 0) = \delta(\zeta)$ is a Dirac δ distribution and $r(\zeta|z = 1) \propto \exp(\beta\zeta)$ is an exponential distribution defined for $\zeta \in [0, 1]$ with inverse temperature β that controls the sharpness of the distribution. (Rolfe, 2016) notes that the autoencoding term can be defined as:

$$\sum_z q(z|x) \int d\zeta \; r(\zeta|z) \log p(x|\zeta) = \int d\zeta \; q(\zeta|x) \log p(x|\zeta),$$

where the marginal

$$q(\zeta|x) = \sum_z q(z|x) r(\zeta|z) \quad (2)$$

is a mixture of two continuous distributions. By factoring the inference model so that x depends on ζ rather than z, the discrete variables can be explicitly eliminated from the ELBO and the reparameterization trick applied.



(a) smoothing transformations (b) inverse CDF

Figure 1: a) Smoothing transformations using exponential distributions. b) Inverse CDF as a function of $q(z=1|x)$ for $\rho = 0.5$ in comparison to the spike-and-exp smoothing (Rolfe, 2016). The inverse CDF resulting from the mixture of exponential distributions approximates the step function that samples from the Bernoulli distribution.

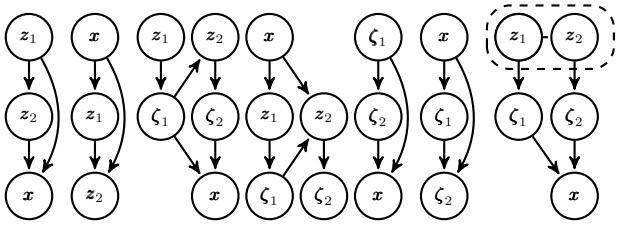


Figure 2: (a) A generative model with binary latent variables $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$, and (b) the corresponding inference model. In (c) and (d), the continuous ζ is introduced and dependencies on $\boldsymbol{z}$ are transferred to dependencies on ζ. In (e) and (f) the binary latent variables $\boldsymbol{z}$ are marginalized out. (g) A generative model with a Boltzmann machine (dashed) prior.

The smoothing transformations in (Rolfe, 2016) are limited to spike-and-X type of transformations (e.g., spike-and-exp and spike-and-Gaussian) where $r(\zeta|z=0)$ is assumed to be a Dirac δ distribution. This property is required for computing the gradient of the KL term in the variational lower bound.

3. Overlapping Transformations

A symmetric smoothing transformation of binary variables can also be defined using two exponential distributions:

$$r(\zeta|z=0) = \frac{e^{-\beta\zeta}}{Z_\beta} \quad \text{and} \quad r(\zeta|z=1) = \frac{e^{\beta(\zeta-1)}}{Z_\beta},$$

for $\zeta \in [0,1]$, where $Z_\beta = (1-e^{-\beta})/\beta$. These conditionals, visualized in Fig. 1(a), define the mixture distribution $q(\zeta|x)$ of Eq. (2). The scalar β acts as an inverse temperature as in the Gumbel softmax relaxation, and as $\beta \to \infty$, $q(\zeta|x)$ approaches $q(z=0|x)\delta(\zeta) + q(z=1|x)\delta(\zeta-1)$.

Application of the reparameterization trick for $q(\zeta|x)$ requires the inverse CDF of $q(\zeta|x)$. In Appendix A of the supplementary material, we show that the inverse CDF is

$$F^{-1}_{q(\zeta|x)}(\rho) = -\frac{1}{\beta}\log\frac{-b+\sqrt{b^2-4c}}{2} \tag{3}$$

where $b = [\rho + e^{-\beta}(q-\rho)]/(1-q) - 1$ and $c = -[qe^{-\beta}]/(1-q)$. Eq. (3) is a differentiable function that converts a sample ρ from the uniform distribution $\mathcal{U}(0,1)$ to a sample from $q(\zeta|x)$. As shown in Fig. 1(b) the inverse CDF approaches a step function as $\beta \to \infty$. However, to benefit from gradient information during training, β is set to a finite value. Appendix C provides further visualizations comparing overlapping transformations to Concrete smoothing (Maddison et al., 2016; Jang et al., 2016).

The overlapping exponential distributions defined here can be generalized to any pair of smooth distributions converging to $\delta(\zeta)$ and $\delta(\zeta-1)$. In Appendix B, we provide analogous results for logistic smoothing distributions.

Next, we apply overlapping transformations to the training of generative models with discrete latent variables. We consider both directed and undirected latent variable priors.

4. Directed Prior

The simplest discrete prior is factorial; however, with conditioning, we can build complex dependencies. To simplify presentation, we illustrate a VAE prior with one or two groups of conditioning variables, but note that the approach straight-forwardly generalizes to many conditioning groups.

Our approach parallels the method developed in (Rolfe, 2016) for undirected graphical models. Consider the generative model in Fig. 2(a) and its corresponding inference model in Fig. 2(b). To train this model using smoothing transformations, we introduce the continuous ζ in Figs. 2(c) and 2(d) in which dependencies on z are transferred to dependencies on ζ. In this way, binary latent variables influence other variables only through their continuous counterparts. In Figs. 2(e) and 2(f) we show the same model but with $\boldsymbol{z}$ marginalized out. The joint $(\boldsymbol{z}, \boldsymbol{\zeta})$ model of Figs. 2(c) and 2(d) gives rise to a looser ELBO than the marginal $\boldsymbol{\zeta}$ model of Figs. 2(e) and 2(f).

4.1. Joint ELBO

Assuming that $p(\boldsymbol{z}_1)$, $p(\boldsymbol{z}_2|\boldsymbol{\zeta}_1)$, $q(\boldsymbol{z}_1|\boldsymbol{x})$, $q(\boldsymbol{z}_2|\boldsymbol{x},\boldsymbol{\zeta}_1)$, $r(\boldsymbol{\zeta}_1|\boldsymbol{z}_1)$, and $r(\boldsymbol{\zeta}_2|\boldsymbol{z}_2)$ are factorial in both the inference and generative models, then $q(\boldsymbol{\zeta}_1|\boldsymbol{x})$ and $q(\boldsymbol{\zeta}_2|\boldsymbol{\zeta}_1,\boldsymbol{x})$ are also factorial with $q(\boldsymbol{\zeta}_1|\boldsymbol{x}) = \prod_i q(\zeta_{1,i}|\boldsymbol{x})$ where $q(\zeta_{1,i}|\boldsymbol{x}) = \sum_{z_{1,i}} r(\zeta_{1,i}|z_{1,i})q(z_{1,i}|\boldsymbol{x})$, and $q(\boldsymbol{\zeta}_2|\boldsymbol{\zeta}_1,\boldsymbol{x}) = \prod_i q(\zeta_{2,i}|\boldsymbol{\zeta}_1,\boldsymbol{x})$ where $q(\zeta_{2,i}|\boldsymbol{\zeta}_1,\boldsymbol{x}) = \sum_{z_{2,i}} r(\zeta_{2,i}|z_{2,i})q(z_{2,i}|\boldsymbol{\zeta}_1,\boldsymbol{x})$. In this case, the ELBO for

the model in Fig. 2(c) and 2(d) is

$$\mathbb{E}_{q(\boldsymbol{\zeta}_1|\boldsymbol{x})}\left[\mathbb{E}_{q(\boldsymbol{\zeta}_2|\boldsymbol{\zeta}_1,\boldsymbol{x})}\left[\log p(\boldsymbol{x}|\boldsymbol{\zeta}_1,\boldsymbol{\zeta}_2)\right]\right] - \mathrm{KL}(q(\boldsymbol{z}_1|\boldsymbol{x})||p(\boldsymbol{z}_1)) - \mathbb{E}_{q(\boldsymbol{\zeta}_1|\boldsymbol{x})}\left[\mathrm{KL}(q(\boldsymbol{z}_2|\boldsymbol{x},\boldsymbol{\zeta}_1)||p(\boldsymbol{z}_2|\boldsymbol{\zeta}_1))\right]. \quad (4)$$

The KL terms corresponding to the divergence between factorial Bernoulli distributions have a closed form. The expectation over $\boldsymbol{\zeta}_1$ and $\boldsymbol{\zeta}_2$ is reparameterized using the technique presented in Sec. 3.

4.2. Marginal ELBO

The ELBO for the marginal graphical model of Fig. 2(e) and Fig. 2(f) is

$$\mathbb{E}_{q(\boldsymbol{\zeta}_1|\boldsymbol{x})}\left[\mathbb{E}_{q(\boldsymbol{\zeta}_2|\boldsymbol{x},\boldsymbol{\zeta}_1)}\left[\log p(\boldsymbol{x}|\boldsymbol{\zeta}_1,\boldsymbol{\zeta}_2)\right]\right] - \mathrm{KL}(q(\boldsymbol{\zeta}_1|\boldsymbol{x})||p(\boldsymbol{\zeta}_1)) - \mathbb{E}_{q(\boldsymbol{\zeta}_1|\boldsymbol{x})}\left[\mathrm{KL}(q(\boldsymbol{\zeta}_2|\boldsymbol{x},\boldsymbol{\zeta}_1)||p(\boldsymbol{\zeta}_2|\boldsymbol{\zeta}_1))\right] \quad (5)$$

with $p(\boldsymbol{\zeta}_1) = \prod_i p(\zeta_{1,i})$ where $p(\zeta_{1,i}) = \sum_{z_i} r(\zeta_{1,i}|z_{1,i})p(z_{1,i})$ and $p(\boldsymbol{\zeta}_2|\boldsymbol{\zeta}_1) = \prod_i p(\zeta_{2,i}|\boldsymbol{\zeta}_1)$ where $p(\zeta_{2,i}|\boldsymbol{\zeta}_1) = \sum_{z_{2,i}} r(\zeta_{2,i}|z_{2,i})p(z_{2,i}|\boldsymbol{\zeta}_1)$. The KL terms no longer have a closed form but can be estimated with the Monte Carlo method. In Appendix D, we show that Eq. (5) provides a tighter bound on $\log p(\boldsymbol{x})$ than does Eq. (4).

5. Boltzmann Machine Prior

(Rolfe, 2016) defined an expressive prior over binary latent variables by using a Boltzmann machine. We build upon that work and present a simpler objective that can still be trained with a low-variance gradient estimate.

To simplify notation, we assume that the prior distribution over the latent binary variables is a restricted Boltzmann machine (RBM), but these results can be extended to general BMs. An RBM defines a probability distribution over binary random variables arranged on a bipartite graph as $p(\boldsymbol{z}_1,\boldsymbol{z}_2) = e^{-E(\boldsymbol{z}_1,\boldsymbol{z}_2)}/Z$ where $E(\boldsymbol{z}_1,\boldsymbol{z}_2) = -\boldsymbol{a}_1^T\boldsymbol{z}_1 - \boldsymbol{a}_2^T\boldsymbol{z}_2 - \boldsymbol{z}_1^T\boldsymbol{W}\boldsymbol{z}_2$ is an energy function with linear biases $\boldsymbol{a}_1$ and $\boldsymbol{a}_2$, and pairwise interactions $\boldsymbol{W}$. Z is the partition function.

Fig. 2(g) visualizes a generative model with a BM prior. As in Figs. 2(c) and 2(d), conditionals are formed on the auxiliary variables $\boldsymbol{\zeta}$ instead of the binary variables $\boldsymbol{z}$. The inference model in this case is identical to the model in Fig. 2(d) and it infers both $\boldsymbol{z}$ and $\boldsymbol{\zeta}$ in a hierarchical structure.

The autoencoding contribution to the ELBO with an RBM prior is again the first term in Eq. (4) since both models share the same inference model structure. However, computing the KL term with the RBM prior is more challenging. Here, a novel formulation for the KL term is introduced. Our derivation can be used for training discrete variational autoencoders with a BM prior without any manual coding of gradients.

We use $\mathbb{E}_{q(\boldsymbol{z},\boldsymbol{\zeta}|\boldsymbol{x})}[f] = \mathbb{E}_{q(\boldsymbol{\zeta}|\boldsymbol{x})}\left[\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{\zeta})}[f]\right]$ to compute the KL contribution to the ELBO:

$$\mathrm{KL}\big(q(\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{\zeta}_1,\boldsymbol{\zeta}_2|\boldsymbol{x})\|p(\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{\zeta}_1,\boldsymbol{\zeta}_2)\big) = \log Z - \mathrm{H}\big(q(\boldsymbol{z}_1|\boldsymbol{x})\big) - \mathbb{E}_{q(\boldsymbol{\zeta}_1|\boldsymbol{x})}\left[\mathrm{H}\big(q(\boldsymbol{z}_2|\boldsymbol{x},\boldsymbol{\zeta}_1)\big)\right] + \quad (6)$$
$$+ \mathbb{E}_{q(\boldsymbol{\zeta}_1|\boldsymbol{x})}\big[\mathbb{E}_{q(\boldsymbol{\zeta}_2|\boldsymbol{x},\boldsymbol{\zeta}_1)}\big[\underbrace{\mathbb{E}_{q(\boldsymbol{z}_1|\boldsymbol{x},\boldsymbol{\zeta}_1)}\big[\mathbb{E}_{q(\boldsymbol{z}_2|\boldsymbol{x},\boldsymbol{\zeta}_1,\boldsymbol{\zeta}_2)}\big[E(\boldsymbol{z}_1,\boldsymbol{z}_2)\big]\big]}_{\text{cross-entropy}}\big]\big].$$

Here, $\mathrm{H}(q)$ is the entropy of the distribution q, which has a closed form when q is factorial Bernoulli. The conditionals $q(\boldsymbol{z}_1|\boldsymbol{x},\boldsymbol{\zeta}_1)$ and $q(\boldsymbol{z}_2|\boldsymbol{x},\boldsymbol{\zeta}_1,\boldsymbol{\zeta}_2)$ are both factorial distributions that have analytic expressions. Denoting

$$\begin{aligned}\mu_{1,i}(\boldsymbol{x}) &\equiv q(z_{1,i}=1|\boldsymbol{x}),\\ \nu_{1,i}(\boldsymbol{x},\boldsymbol{\zeta}_1) &\equiv q(z_{1,i}=1|\boldsymbol{x},\boldsymbol{\zeta}_1),\\ \mu_{2,i}(\boldsymbol{x},\boldsymbol{\zeta}_1) &\equiv q(z_{2,i}=1|\boldsymbol{x},\boldsymbol{\zeta}_1),\\ \nu_{2,i}(\boldsymbol{x},\boldsymbol{\zeta}_1,\boldsymbol{\zeta}_2) &\equiv q(z_{2,i}=1|\boldsymbol{x},\boldsymbol{\zeta}_1,\boldsymbol{\zeta}_2),\end{aligned}$$

it is straightforward to show that

$$\nu_{1,i}(\boldsymbol{x},\boldsymbol{\zeta}_1) = \frac{q(z_{1,i}=1|\boldsymbol{x})r(\zeta_{1,i}|z_{1,i}=1)}{\sum_{z_{1,i}} q(z_{1,i}|\boldsymbol{x})r(\zeta_{1,i}|z_{1,i})} = \sigma\Big(g(\mu_{1,i}(\boldsymbol{x})) + \log\big[\frac{r(\zeta_{1,i}|z=1)}{r(\zeta_{1,i}|z=0)}\big]\Big),$$

where $\sigma(x) = 1/(1+e^{-x})$ is the logistic function, and $g(\mu) \equiv \log\big[\mu/\big(1-\mu\big)\big]$ is the logit function. A similar expression holds for $\boldsymbol{\nu}_2(\boldsymbol{x},\boldsymbol{\zeta}_1,\boldsymbol{\zeta}_2)$. The expectation marked as cross-entropy in Eq. (6) corresponds to the cross-entropy between a factorial distribution and an unnormalized Boltzmann machine which is

$$-\boldsymbol{a}_1^T\boldsymbol{\nu}_1(\boldsymbol{x},\boldsymbol{\zeta}_1) - \boldsymbol{a}_2^T\boldsymbol{\nu}_2(\boldsymbol{x},\boldsymbol{\zeta}_1,\boldsymbol{\zeta}_2) - \boldsymbol{\nu}_1(\boldsymbol{x},\boldsymbol{\zeta}_1)^T\boldsymbol{W}\boldsymbol{\nu}_2(\boldsymbol{x},\boldsymbol{\zeta}_1,\boldsymbol{\zeta}_2).$$

Finally, we use the equalities $\mathbb{E}_{q(\boldsymbol{\zeta}_1|\boldsymbol{x})}[\boldsymbol{\nu}_1(\boldsymbol{x},\boldsymbol{\zeta}_1)] = \boldsymbol{\mu}_1(\boldsymbol{x})$ and $\mathbb{E}_{q(\boldsymbol{\zeta}_2|\boldsymbol{x},\boldsymbol{\zeta}_1)}[\boldsymbol{\nu}_2(\boldsymbol{x},\boldsymbol{\zeta}_1,\boldsymbol{\zeta}_2)] = \boldsymbol{\mu}_2(\boldsymbol{x},\boldsymbol{\zeta}_1)$ to simplify the cross-entropy term which defines the KL as

$$\begin{aligned}&\mathrm{KL}\big(q(\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{\zeta}_1,\boldsymbol{\zeta}_2|\boldsymbol{x})\|p(\boldsymbol{z}_1,\boldsymbol{z}_2,\boldsymbol{\zeta}_1,\boldsymbol{\zeta}_2)\big) = \log Z\\ &- \mathrm{H}\big(q(\boldsymbol{z}_1|\boldsymbol{x})\big) - \mathbb{E}_{q(\boldsymbol{\zeta}_1|\boldsymbol{x})}\left[\mathrm{H}\big(q(\boldsymbol{z}_2|\boldsymbol{x},\boldsymbol{\zeta}_1)\big)\right]\\ &- \boldsymbol{a}_1^T\boldsymbol{\mu}_1(\boldsymbol{x}) - \mathbb{E}_{q(\boldsymbol{\zeta}_1|\boldsymbol{x})}\left[\boldsymbol{a}_2^T\boldsymbol{\mu}_2(\boldsymbol{x},\boldsymbol{\zeta}_1)\right]\\ &- \mathbb{E}_{q(\boldsymbol{\zeta}_1|\boldsymbol{x})}\left[\boldsymbol{\nu}_1(\boldsymbol{x},\boldsymbol{\zeta}_1)^T\boldsymbol{W}\boldsymbol{\mu}_2(\boldsymbol{x},\boldsymbol{\zeta}_1)\right].\end{aligned}$$

All terms contributing to the KL other than $\log Z$ can be computed analytically given samples from the hierarchical encoder. Expectations with respect to $q(\boldsymbol{\zeta}_1|\boldsymbol{x})$ are reparameterized using the inverse CDF function. Any automatic differentiation (AD) library can then back-propagate gradients through the network. Only $\log Z$ requires special treatment. In Appendix E, we show how this term can also be included in the objective function so that its gradient is computed automatically. The ability of AD to calculate gradients stands in contrast to (Rolfe, 2016) where gradients must be manually coded. This pleasing property is a result of $r(\zeta|z)$ having the same support for both $z=0$ and $z=1$, and having a probabilistic $q(z|x,\zeta)$ which is not the case for the spike-and-X transformations of (Rolfe, 2016).

6. DVAE++

In previous sections, we have illustrated with simple examples how overlapping transformations can be used to train discrete latent variable models with either directed or undirected priors. Here, we develop a network architecture (DVAE++) that improves upon convolutional VAEs for generative image modeling.

DVAE++ features both global discrete latent variables (to capture global properties such as scene or object type) and local continuous latent variables (to capture local properties such as object pose, orientation, or style). Both generative and inference networks rely on an autoregressive structure defined over groups of latent and observed variables. As we are modeling images, conditional dependencies between groups of variables are captured with convolutional neural networks. DVAE++ is similar to the convolutional VAEs used in (Kingma et al., 2016; Chen et al., 2016), but does not use normalizing flows.

6.1. Graphical Model

The DVAE++ graphical model is visualized in Fig. 3. Global and local variables are indicated by $\boldsymbol{z}$ and $\boldsymbol{h}$ respectively. Subscripts indicate different groups of random variables. The conditional distribution of each group is factorial – except for $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ in the prior, which is modeled with an RBM. Global latent variables are represented with boxes and local variables are represented with 3D volumes as they are convolutional.

Groups of local continuous variables are factorial (independent). This assumption limits the ability of the model to capture correlations at different spatial locations and different depths. While the autoregressive structure mitigates this defect, we rely mainly on the discrete global latent variables to capture long-range dependencies. The discrete nature of the global RBM prior allows DVAE++ to capture richly-correlated discontinuous hidden factors that influence data generation.

Fig. 3(a) defines the generative model as

$$p(\boldsymbol{z},\boldsymbol{\zeta},\boldsymbol{h},\boldsymbol{x}) = p(\boldsymbol{z}) \prod_i r(\zeta_{1,i}|z_{1,i}) r(\zeta_{2,i}|z_{2,i}) \times \prod_j p(\boldsymbol{h}_j|\boldsymbol{h}_{<j},\boldsymbol{\zeta}) p(\boldsymbol{x}|\boldsymbol{\zeta},\boldsymbol{h})$$

where $p(\boldsymbol{z})$ is an RBM, $\boldsymbol{\zeta} = [\boldsymbol{\zeta}_1, \boldsymbol{\zeta}_2]$, and r is the smoothing transformation that is applied elementwise to $\boldsymbol{z}$. The conditional $p(\boldsymbol{h}_j|\boldsymbol{h}_{<j},\boldsymbol{\zeta})$ is defined over the j^{th} local variable group using a factorial normal distribution. Inspired by (Reed et al., 2017; Denton et al., 2015), the conditional on the data variable $p(\boldsymbol{x}|\boldsymbol{\zeta},\boldsymbol{h})$ is decomposed into several factors defined on different scales of $\boldsymbol{x}$:

$$p(\boldsymbol{x}|\boldsymbol{\zeta},\boldsymbol{h}) = p(\boldsymbol{x}_0|\boldsymbol{\zeta},\boldsymbol{h}) \prod_i p(\boldsymbol{x}_i|\boldsymbol{\zeta},\boldsymbol{h},\boldsymbol{x}_{<i})$$

Here, $\boldsymbol{x}_0$ is of size 4×4 and it represents downsampled $\boldsymbol{x}$ in the lowest scale. Conditioned on $\boldsymbol{x}_0$, we generate $\boldsymbol{x}_1$ in the next scale, which is of the size 8×8. This process is continued until the full-scale image is generated (see Appendix G.1 for more details). Here, each conditional is represented using a factorial distribution. For binary images, a factorial Bernoulli distribution is used; for colored images a factorial mixture of discretized logistic distributions is used (Salimans et al., 2017).

The inference model of Fig. 3(b) conditions over latent variables in a similar order as the generative model:

$$q(\boldsymbol{z},\boldsymbol{\zeta},\boldsymbol{h}|\boldsymbol{x}) = q(\boldsymbol{z}_1|\boldsymbol{x}) \prod_i r(\zeta_{1,i}|z_{1,i}) \times q(\boldsymbol{z}_2|\boldsymbol{x},\boldsymbol{\zeta}_1) \prod_k r(\zeta_{2,k}|z_{2,k}) \prod_j q(\boldsymbol{h}_j|\boldsymbol{\zeta},\boldsymbol{h}_{<j}).$$

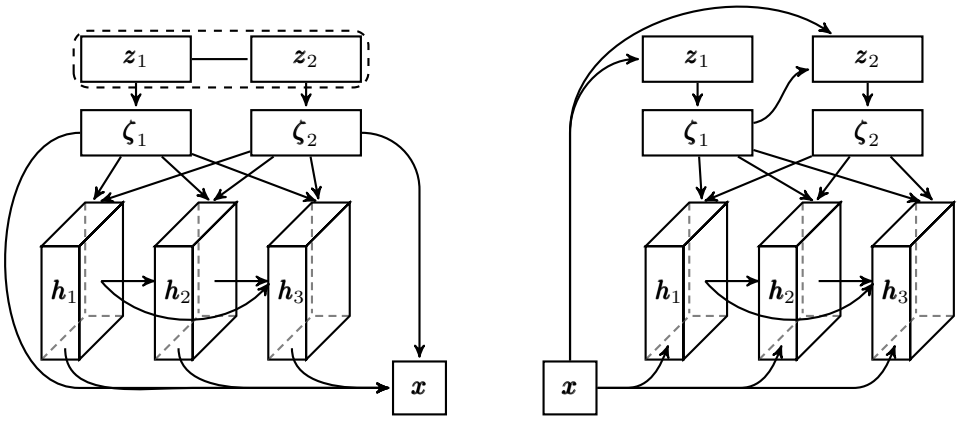
The conditionals $q(\boldsymbol{z}_1|\boldsymbol{x})$ and $q(\boldsymbol{z}_2|\boldsymbol{x},\boldsymbol{\zeta}_1)$ are each modeled with a factorial Bernoulli distribution, and $q(\boldsymbol{h}_j|\boldsymbol{\zeta},\boldsymbol{h}_{<j})$ represents the conditional on the j^{th} group of local variables.

DVAE++ is related to VAEs with mixture priors (Makhzani et al., 2015; Tomczak & Welling, 2017). The discrete variables $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ take exponentially many joint configurations where each configuration corresponds to a mixture component. These components are mixed by $p(\boldsymbol{z}_1, \boldsymbol{z}_2)$ in the generative model. During training, the inference model maps each data point to a small subset of all the possible mixture components. Thus, the discrete prior learns to suppress the probability of configurations that are not used by the inference model. Training results in a multimodal $p(\boldsymbol{z}_1, \boldsymbol{z}_2)$ that assigns similar images to a common discrete mode.

6.2. Neural Network Architecture

We use a novel neural network architecture to realize the conditional probabilities within the graphical model Fig. 3. The network uses residual connections (He et al., 2016) with squeeze-and-excitation (SE) blocks (Hu et al., 2017) that have shown state-of-the-art image classification performance. Our architecture is explained fully in Appendix G, and here we sketch the main components. We refer to a SE-ResNet block as a residual block, and the network is created by combining either residual blocks, fully-connected layers, or convolutional layers.

The encoder uses a series of downsampling residual blocks to extract convolutional features from an input image. This residual network is considered as a pre-processing step that extracts convolutional feature maps at different scales. The output of this network at the highest level is fed to fully-connected networks that define $q(\boldsymbol{z}_i|\boldsymbol{x},\boldsymbol{\zeta}_{<i})$ successively for



(a) generative model (b) inference model

Figure 3: a) In the generative model, binary global latent variables $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ are modeled by an RBM (dashed) and a series of local continuous variables are generated in an autoregressive structure using residual networks. b) After forming distributions over the global variables, the inference model defines the conditional on the local latent variables similarly using residual networks.

all the global latent variables. The feature maps at an intermediate scale are fed to another set of residual networks that define $q(\boldsymbol{h}_j|\boldsymbol{x}, \boldsymbol{\zeta}, \boldsymbol{h}_{<j})$ successively for all the local latent variables.

The decoder uses an upsampling network to scale-up the global latent variables to the intermediate scale. Then, the output of this network is fed to a set of residual networks that define $p(\boldsymbol{h}_j|\boldsymbol{\zeta}, \boldsymbol{h}_{<j})$ one at a time at the same scale. Finally, another set of residual networks progressively scales the samples from the latent variables up to the data space. In the data space, a distribution on the smallest scale $\boldsymbol{x}_0$ is formed using a residual network. Given samples at this scale, the distribution at the next scale is formed using another upsampling residual network. This process is repeated until the image is generated at full scale.

With many layers of latent variables, the VAE objective often turns off many of the latent variables by matching their distribution in the inference model to the prior. The latent units are usually removed differentially across different groups. Appendix H presents a technique that enables efficient use of latent variables across all groups.

7. Experiments

To provide a comprehensive picture of overlapping transformations and DVAE++, we conduct three sets of experiments. In Sec. 7.1 and Sec. 7.2 we train a VAE with several layers of latent variables with a feed-forward encoder and decoder. This allows to compare overlapping transformations with previous work on discrete latent variables. In Sec. 7.3, we then compare DVAE++ to several baselines.

7.1. Comparison with Previous Discrete Latent Variable Models

We compare overlapping transformations to NVIL (Mnih & Gregor, 2014), MuProp (Gu et al., 2015), REBAR (Tucker et al., 2017), and Concrete (Maddison et al., 2016) for training discrete single-layer latent variable models. We follow the structure used by (Tucker et al., 2017) in which the prior distribution and inference model are factorial Bernoulli with 200 stochastic variables. In this setting, the inference and generative models are either linear or nonlinear functions. In the latter case, two layers of deterministic hidden units of the size 200 with tanh activation are used.

We use the settings in (Tucker et al., 2017) to initialize the parameters, define the model, and optimize the parameters for the same number of iterations. However, (Tucker et al., 2017) uses the Adam optimizer with $\beta_2 = 0.99999$ in training. We used Adam with its default parameters except for ϵ which is set to 10^{-3}. The learning rate is selected from the set $\{1 \cdot 10^{-4}, 5 \cdot 10^{-4}\}$. The inverse temperature β for smoothing is annealed linearly during training with initial and final values chosen using cross validation from $\{5, 6, 7, 8\}$ and $\{12, 14, 16, 18\}$ respectively. In Table 1, the performance of our model is compared with several state-of-the-art techniques proposed for training binary latent models on (statically) binarized MNIST (Salakhutdinov & Murray, 2008) and OMNIGLOT (Lake et al., 2015). At test time, all models are evaluated in the binary limit ($\beta = \infty$). Smoothing transformations slightly outperform previous

Table 1: Overlapping transformations are compared against different single-sample based approaches proposed for training binary latent variable models. The performance is measured by 100 importance weighted samples (Burda et al., 2015). Mean ± standard deviation for five runs are reported. Baseline performances are taken from (Tucker et al., 2017).

MNIST (static)	NVIL	MuProp	REBAR	Concrete	Joint ELBO	Marg. ELBO
Linear	-108.35 ± 0.06	-108.03 ± 0.07	-107.65 ± 0.08	**-107.00 ± 0.10**	-107.98 ± 0.10	-108.57 ± 0.10
Nonlinear	-100.00 ± 0.10	-100.66 ± 0.08	-100.69 ± 0.08	-99.54 ± 0.06	**-99.16 ± 0.12**	**-99.10 ± 0.21**
OMNIGLOT						
Linear	**-117.59 ± 0.04**	-117.64 ± 0.04	-117.65 ± 0.04	-117.65 ± 0.05	**-117.38 ± 0.08**	-118.35 ± 0.06
Nonlinear	-116.57 ± 0.08	-117.51 ± 0.09	-118.02 ± 0.05	-116.69 ± 0.08	**-113.83 ± 0.11**	**-113.76 ± 0.18**

Table 2: The performance of the VAE model with an RBM prior trained with the overlapping transformation is compared against (Rolfe, 2016) as well as the directed VAE models (Fig. 2). The performance is measured by 4000 importance weighted samples (Burda et al., 2015). Mean ± standard deviation for five runs are reported.

	MNIST (static)				**OMNIGLOT**			
	RBM (ours)	RBM (Rolfe)	Joint ELBO	Marg. ELBO	RBM (ours)	RBM (Rolfe)	Joint ELBO	Marg. ELBO
1 Linear	**-91.21±0.11**	-91.55±0.08	-106.70±0.08	-106.80±0.19	**-109.66±0.09**	**-109.83±0.17**	-117.62±0.09	-117.78±0.07
2 Linear	-94.15±0.45	**-91.06±0.21**	-98.16±0.11	-98.56±0.10	**-109.01±0.45**	-110.35±0.14	-111.21±0.12	-111.49±0.08
1 Nonlin.	**-85.41±0.04**	-85.57±0.03	-95.04±0.10	-95.06±0.08	**-102.62±0.07**	-103.12±0.06	-108.77±0.24	-108.82±0.20
2 Nonlin.	**-84.27±0.05**	-84.52±0.05	-87.96±0.13	-88.23±0.11	**-100.55±0.05**	-105.60±0.68	-103.57±0.15	-104.05±0.22

techniques in most cases. In the case of the nonlinear model on OMNIGLOT, the difference is about 2.8 nats.

7.2. Comparison with Previous RBM Prior VAE

Techniques such as KL annealing (Sønderby et al., 2016), batch normalization (Ioffe & Szegedy, 2015), autoregressive inference/prior, and learning-rate decay can significantly improve the performance of a VAE beyond the results reported in Sec. 7.1. In this second set of experiments, we evaluate overlapping transformations by comparing the training of a VAE with an RBM prior to the original DVAE (Rolfe, 2016), both of which include these improvements. For a fair comparison, we apply only those techniques that were also used in (Rolfe, 2016). We examine VAEs with one and two latent layers with feed-forward linear or nonlinear inference and generative models. In the one-latent-layer case, the KL term in both our model and (Rolfe, 2016) reduces to the mean-field approximation. The only difference in this case lies in the overlapping transformations used here and the original smoothing method of (Rolfe, 2016). In the two-latent-layer case, our inference and generative model have the forms depicted in Fig. 2(d) and Fig. 2(g). Again, all models are evaluated in the binary limit at the test time.

Comparisons are reported in Table 2. For reference, we also provide the performance of the directed VAE models with the structures visualized in Fig. 2(c) to Fig. 2(f). Implementation details are provided in Appendix F. Two observations can be made from Table 2. First, our smoothing transformation outperforms (Rolfe, 2016) in most cases. In some cases the difference is as large as 5.1 nats. Second, the RBM prior performs better than a directed prior of the same size.

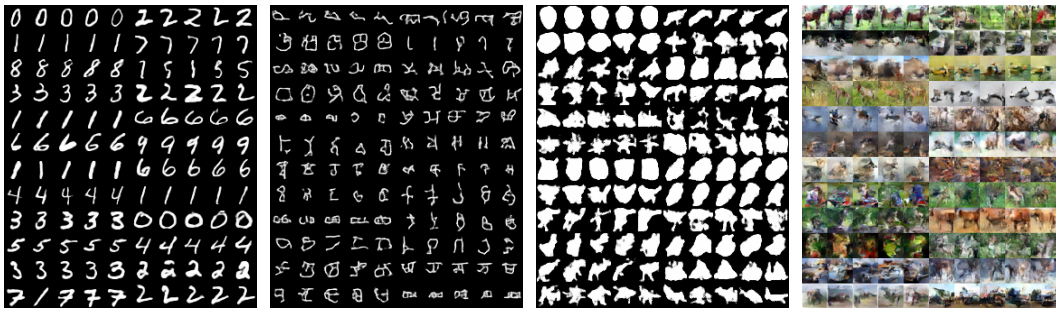
7.3. Experiments on DVAE++

Lastly, we explore the performance of DVAE++ for density estimation on 2D images. In addition to statically binarized MNIST and OMNIGLOT, we test dynamically binarized MNIST (LeCun et al., 1998) and Caltech-101 silhouettes (Marlin et al., 2010). All datasets have 28×28 binary pixel images. We use the same architecture for the MNIST and OMNIGLOT datasets, but because the Caltech-101 silhouettes dataset is smaller, our model easily overfits. Consequently, we use a shallower architecture for Caltech-101. We also evaluate DVAE++ on the CIFAR10 dataset, which consists of 32×32 pixel natural images. Appendix G lists the details of our architecture for different datasets.

Our goal is to determine whether we can use overlapping transformations to train a convolutional VAE with an RBM prior, and whether the RBM prior in DVAE++ captures global discrete hidden factors. In addition to DVAE++ (which uses binary global latent variables and continuous local latent variables), four different baselines are introduced by modifying the global and local distributions. These baselines are listed in Table 3. For RBM (Rolfe), the spike-and-exp smoothing transformation is used and the ELBO is optimized using the derivation supplied in (Rolfe, 2016). For Bernoulli latent variables, we used the marginal distributions proposed in Sec. 4.2. For all the models, we used 16 layers of local latent variables each with 32 random variables at each spatial location. For the RBM global variables, we used 16 binary variables for all the binary datasets and 128 binary variables for CIFAR10. We cross-validated the number of the hierarchical layers in the inference model for the global variables from the set $\{1, 2, 4\}$. We used an unconditional decoder (i.e., factorial $p(\boldsymbol{x}|\boldsymbol{\zeta}, \boldsymbol{h})$) for the MNIST

Table 3: DVAE++ compared against different baselines on several datasets. The performance is reported in terms of the log-likelihood values for all the dataset except for CIFAR10, in which *bits per dimension* is reported. In general, DVAE++ with RBM global prior and normal local variables outperforms the baselines.

Latent type	Global	Local	MNIST (static)	MNIST (dynamic)	OMNIGLOT	Caltech-101	CIFAR10
All cont.	Normal	Normal	-79.40	-78.59	-92.51	-82.24	3.40
Mixed	RBM (Rolfe)	Normal	**-79.04**	-78.65	-92.56	-81.95	3.39
	RBM (ours)	Normal	-79.17	**-78.49**	**-92.38**	**-81.88**	**3.38**
All disc.	RBM (ours)	Bernoulli	-79.72	-79.55	-93.95	-85.40	3.59
	Bernoulli	Bernoulli	-79.90	-79.62	-93.87	-86.57	3.62
Unconditional decoder			Yes	Yes	No	No	No



(a) MNIST (b) OMNIGLOT (c) Caltech-101 (d) CIFAR10

Figure 4: Visualization of samples generated from our model trained on different datasets. In each figure, every five successive samples in each row are generated from a fixed sample drawn from the global RBM prior. Our global latent variables typically capture discontinuous global structures such as digit classes in MNIST or scene configuration in CIFAR10.

datasets. We measure performance by estimating test set log-likelihood (again, according to the binary model) with 4000 importance weighted samples. Appendix I presents additional ablation experiments.

Table 3 groups the baselines into three categories: all continuous latent, discrete global and continuous local (mixed), and all discrete. Within the mixed group, DVAE++ with RBM prior generally outperforms the same model trained with (Rolfe, 2016)'s. Replacing the continuous normal local variables with Bernoulli variables does not dramatically hurt the performance. For example, in the case of statically and dynamically binarized MNIST dataset, we achieve -79.72 and -79.55 respectively with unconditional decoder and 3.59 on CIFAR10 with conditional decoder. To the best of our knowledge these are the best reported results on these datasets with binary latent variables. Samples generated from DVAE++ are visualized in Fig. 4. As shown, the discrete global prior clearly captures discontinuous latent factors such as digit category or scene configuration.

DVAE++ results are comparable to current state-of-the-art convolutional latent variable models such as VampPrior (Tomczak & Welling, 2017) and variational lossy autoencoder (VLAE) (Chen et al., 2016). We note two features of these models that may offer room for further improvement for DVAE++. First, the conditional decoder used here makes independence assumptions in each scale, whereas the state-of-the-art techniques are based on PixelCNN (Van Den Oord et al., 2016), which assumes full autoregressive dependencies. Second, methods such as VLAE use normalizing flows for flexible inference models that reduce the KL cost on the convolutional latent variables. Here, the independence assumption in each local group in DVAE++ can cause a significant KL penalty.

8. Conclusions

We have introduced a new family of smoothing transformations consisting of a mixture of two overlapping distributions and have demonstrated that these transformations can be used for training latent variable models with either directed or undirected priors. Using variational bounds derived for both cases, we developed DVAE++ having a global RBM prior and local convolutional latent variables. All experiments used exponential mixture components, but it would be interesting to explore the efficacy of other choices.

References

Bengio, Y., Léonard, N., and Courville, A. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*,

2013.

Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A., Jozefowicz, R., and Bengio, S. Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pp. 10–21, 2016.

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

Chen, X., Kingma, D. P., Salimans, T., Duan, Y., Dhariwal, P., Schulman, J., Sutskever, I., and Abbeel, P. Variational lossy autoencoder. *arXiv preprint arXiv:1611.02731*, 2016.

Chung, J., Ahn, S., and Bengio, Y. Hierarchical multiscale recurrent neural networks. *arXiv preprint arXiv:1609.01704*, 2016.

Clevert, D.-A., Unterthiner, T., and Hochreiter, S. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

Denton, E. L., Chintala, S., Fergus, R., et al. Deep generative image models using a Laplacian pyramid of adversarial networks. In *Advances in neural information processing systems*, pp. 1486–1494, 2015.

Gregor, K., Danihelka, I., Mnih, A., Blundell, C., and Wierstra, D. Deep autoregressive networks. *arXiv preprint arXiv:1310.8499*, 2013.

Gu, S., Levine, S., Sutskever, I., and Mnih, A. Muprop: Unbiased backpropagation for stochastic neural networks. *arXiv preprint arXiv:1511.05176*, 2015.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition*, 2016.

Hu, J., Shen, L., and Sun, G. Squeeze-and-excitation networks. *arXiv preprint arXiv:1709.01507*, 2017.

Hukushima, K. and Nemoto, K. Exchange Monte Carlo method and application to spin glass simulations. *Journal of the Physical Society of Japan*, 65(6):1604–1608, 1996.

Iba, Y. Extended ensemble Monte Carlo. *International Journal of Modern Physics C*, 12(05):623–656, 2001.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pp. 448–456, 2015.

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

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

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. 2014.

Kingma, D. P., Mohamed, S., Rezende, D. J., and Welling, M. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pp. 3581–3589, 2014.

Kingma, D. P., Salimans, T., Jozefowicz, R., Chen, X., Sutskever, I., and Welling, M. Improved variational inference with inverse autoregressive flow. In *Advances in Neural Information Processing Systems*, pp. 4743–4751, 2016.

Klambauer, G., Unterthiner, T., Mayr, A., and Hochreiter, S. Self-normalizing neural networks. In *Advances in Neural Information Processing Systems*, pp. 972–981, 2017.

Lake, B. M., Salakhutdinov, R., and Tenenbaum, J. B. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

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

Maaløe, L., Fraccaro, M., and Winther, O. Semi-supervised generation with cluster-aware generative models. *arXiv preprint arXiv:1704.00637*, 2017.

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

Makhzani, A., Shlens, J., Jaitly, N., Goodfellow, I., and Frey, B. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

Marlin, B., Swersky, K., Chen, B., and Freitas, N. Inductive principles for restricted Boltzmann machine learning. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 509–516, 2010.

Mnih, A. and Gregor, K. Neural variational inference and learning in belief networks. *arXiv preprint arXiv:1402.0030*, 2014.

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

Reed, S. E., van den Oord, A., Kalchbrenner, N., Gómez, S., Wang, Z., Belov, D., and de Freitas, N. Parallel multiscale autoregressive density estimation. In *Proceedings of The 34th International Conference on Machine Learning*, 2017.

Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, pp. 1278–1286, 2014.

Rolfe, J. T. Discrete variational autoencoders. *arXiv preprint arXiv:1609.02200*, 2016.

Salakhutdinov, R. and Murray, I. On the quantitative analysis of deep belief networks. In *Proceedings of the 25th international conference on Machine learning*, pp. 872–879. ACM, 2008.

Salimans, T., Karpathy, A., Chen, X., and Kingma, D. P. PixelCNN++: Improving the pixelCNN with discretized logistic mixture likelihood and other modifications. *arXiv preprint arXiv:1701.05517*, 2017.

Sønderby, C. K., Raiko, T., Maaløe, L., Sønderby, S. K., and Winther, O. Ladder variational autoencoders. In *Advances in neural information processing systems*, pp. 3738–3746, 2016.

Tieleman, T. Training restricted Boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the 25th international conference on Machine learning*, pp. 1064–1071. ACM, 2008.

Tomczak, J. M. and Welling, M. VAE with a VampPrior. *arXiv preprint arXiv:1705.07120*, 2017.

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

Van Den Oord, A., Kalchbrenner, N., and Kavukcuoglu, K. Pixel recurrent neural networks. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning*, pp. 1747–1756. JMLR. org, 2016.

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

Younes, L. Parametric inference for imperfectly observed Gibbsian fields. *Probability theory and related fields*, 1989.