Training Variational Autoencoders with Discrete Latent Variables Using Importance Sampling

Alexander Bartler, Felix Wiewel, Lukas Mauch and Bin Yang
Institute of Signal Processing and System Theory
University of Stuttgart
Stuttgart, Germany
Email: {alexander.bartler, felix.wiewel, lukas.mauch, bin.yang}@iss.uni-stuttgart.de

***Abstract*—The Variational Autoencoder (VAE) is a popular generative latent variable model that is often used for representation learning. Standard VAEs assume continuous-valued latent variables and are trained by maximization of the evidence lower bound (ELBO). Conventional methods obtain a differentiable estimate of the ELBO with reparametrized sampling and optimize it with Stochastic Gradient Descend (SGD). However, this is not possible if we want to train VAEs with discrete-valued latent variables, since reparametrized sampling is not possible. In this paper, we propose an easy method to train VAEs with binary or categorically valued latent representations. Therefore, we use a differentiable estimator for the ELBO which is based on importance sampling. In experiments, we verify the approach and train two different VAEs architectures with Bernoulli and categorically distributed latent representations on two different benchmark datasets.**

***Index Terms*—variational autoencoder, discrete latent variables, importance sampling**

I. The Variational Autoencoder

The Variational Autoencoder (VAE) is a generative model which is trained to approximate the true data generating distribution $p(\mathbf{x})$ of an observed random vector $\mathbf{x}$ from a given training set $\mathcal{D} = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$ [1,2]. It is an especially suited model if $\mathbf{x}$ is high dimensional or has highly nonlinear dependent elements. Therefore, the VAE is often used for tasks like density estimation, data generation, data interpolation [3], outlier and anomaly detection [4,5] or clustering [6,7].

The VAE is an latent variable model, where the observations $\mathbf{x} \sim p(\mathbf{x}|\mathbf{z})$ are dependent on latent variables $\mathbf{z} \sim p(\mathbf{z})$. During training, the VAE maximizes the probability $p(\mathbf{x})$ to observe the data $\mathbf{x}$. Therefore, the negative evidence lower bound (ELBO)

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}) &= -\mathcal{L}_L(\boldsymbol{\theta}) + \mathcal{L}_{KL}(\boldsymbol{\theta}) && (1)\\ &= -\mathrm{E}_{q(\mathbf{z}|\mathbf{x})}\left[\ln p(\mathbf{x}|\mathbf{z})\right] + \mathrm{D}_{KL}(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) && (2)\\ &= -\ln p(\mathbf{x}) + \mathrm{D}_{KL}(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z}|\mathbf{x})) && (3)\end{aligned}$$

is minimized, where $p(\mathbf{z}|\mathbf{x}) = p(\mathbf{x}|\mathbf{z})p(\mathbf{z}) / \int p(\mathbf{x}|\mathbf{z})p(\mathbf{z})\mathrm{d}\mathbf{z}$ is the true but intractable posterior distribution the model assigns to $\mathbf{z}$, $q(\mathbf{z}|\mathbf{x})$ is the corresponding tractable variational approximation and $\mathrm{D}_{KL}(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z}|\mathbf{x}))$ is the Kullback-Leibler (KL) divergence between $p(\mathbf{z}|\mathbf{x})$ and $q(\mathbf{z}|\mathbf{x})$. Because $\mathrm{D}_{KL}(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z}|\mathbf{x})) > 0$, minimizing $\mathcal{L}(\boldsymbol{\theta})$ means to maximize the probability $p(\mathbf{x})$ the model assigns to observations $\mathbf{x}$. Therefore, $\mathrm{D}_{KL}(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z}|\mathbf{x}))$ must be as close as possible to 0, meaning that after training $q(\mathbf{z}|\mathbf{x})$ is a very good approximation of the true posterior $p(\mathbf{z}|\mathbf{x})$. [1] proposed to minimize $\mathcal{L}(\boldsymbol{\theta})$ using stochastic gradient descent, which they called Stochastic Gradient Variational Bayes (SGVB).

The VAE uses parametric distributions that are parametrized by an encoder network with parameters $\boldsymbol{\theta}_E$ and a decoder network with parameters $\boldsymbol{\theta}_D$ for both $q(\mathbf{z}|\mathbf{x})$ and $p(\mathbf{x}|\mathbf{z})$, respectively. This leads to the well known encoder-decoder structure in Fig. 1. The data likelihood is a distribution with mean $\hat{\mathbf{x}}$ that is the output of the decoder network. Further, we assume in this paper that the variational posterior $q(\mathbf{z}|\mathbf{x})$ is a distribution from the exponential family

$$q(\mathbf{z}|\mathbf{x}) = \exp(\boldsymbol{\eta}^T(\mathbf{x};\boldsymbol{\theta}_E)T(\mathbf{z}) - A(\boldsymbol{\eta}(\mathbf{x};\boldsymbol{\theta}_E))) \qquad (4)$$

with natural parameters $\boldsymbol{\eta}(\mathbf{x};\boldsymbol{\theta}_E)$, sufficient statistic $T(\mathbf{z})$ and log partition function $A(\boldsymbol{\eta}(\mathbf{x};\boldsymbol{\theta}_E))$. This gives us the flexibility to study training with different $q(\mathbf{z}|\mathbf{x})$ in the same mathematical framework. As shown in Fig. 1, the natural parameters $\boldsymbol{\eta}$ are the output of the encoder network, where we drop the arguments $\mathbf{x}, \boldsymbol{\theta}_E$ for shorter notations in the remainder of the paper.

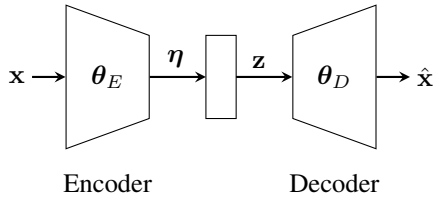


Fig. 1: The encoder-decoder structure of a VAE. The encoder parametrizes $q(\mathbf{z}|\mathbf{x})$ as an exponential family distribution with natural parameters $\boldsymbol{\eta}$ and the decoder parametrizes $p(\mathbf{x}|\mathbf{z})$.

The conventional VAE proposed in [1, 2] learns continuous latent representations $\mathbf{z} \in \mathbb{R}^c$. It uses i.i.d. Gaussian distributed $\mathbf{z}$, meaning that $\boldsymbol{\eta} = [\mu_1/\sigma_1^2, -1/(2\sigma_1^2), ..., \mu_c/\sigma_c^2, -1/(2\sigma_c^2)]^T$, $T(\mathbf{z}) = [z_1, z_1^2, ..., z_c, z_c^2]^T$ and $A(\boldsymbol{\eta})$ are chosen such that $q(\mathbf{z}|\mathbf{x})$ integrates to one. The likelihood is also Gaussian with $p(\mathbf{x}|\mathbf{z}) \sim N(\hat{\mathbf{x}}, \mathbf{1})$. But in many applications learning discrete rather than continuous representations is advantageous. Binary representations $\mathbf{z} \in \{0,1\}^c$ can, for example, be used very efficiently for hashing, which is a

978-9-0827-9703-9/19/$31.00 ©2019 IEEE

powerful method for large-scale visual search [8]. Learning categorical representations $\mathbf{z} \in \{\mathbf{e}_1, ..., \mathbf{e}_c\}$ is interesting, because this naturally leads to clustering of the data $\mathbf{x}$. Further, for both binary and categorical $\mathbf{z}$, it is easy to find entropy based heuristics to choose the size of the latent space, because the entropy is bounded for discrete $\mathbf{z}$.

However, training VAEs with discrete latent representations is problematic, since standard SGVB can not be applied for optimization. Because SGVB is a gradient based method, we need to calculate the derivative of the two cost terms in Eq. 1 with respect to the encoder and decoder parameters

$$\frac{\partial}{\partial \boldsymbol{\theta}} \mathcal{L}_{KL}(\boldsymbol{\theta}) = \frac{\partial}{\partial \boldsymbol{\theta}} \mathrm{D}_{KL}(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) \quad (5)$$

$$\frac{\partial}{\partial \boldsymbol{\theta}} \mathcal{L}_{L}(\boldsymbol{\theta}) = \frac{\partial}{\partial \boldsymbol{\theta}} \mathrm{E}_{q(\mathbf{z}|\mathbf{x})} \left[\ln p(\mathbf{x}|\mathbf{z})\right], \quad (6)$$

where $\mathcal{L}_{KL}(\boldsymbol{\theta})$ only depends on the encoder parameters and the expected log-likelihood term $\mathcal{L}_L(\boldsymbol{\theta})$ depends on both encoder and decoder parameters. For a suited choice of $p(\mathbf{z})$ and $q(\mathbf{z}|\mathbf{x})$, $\mathcal{L}_{KL}(\boldsymbol{\theta})$ can be calculated in closed form. However, $\mathcal{L}_L(\boldsymbol{\theta})$ contains an expectation over $\mathbf{z} \sim q(\mathbf{z}|\mathbf{x})$ that has to be estimated during training. A good estimator $\hat{\mathcal{L}}_L(\boldsymbol{\theta})$ for $\mathcal{L}_L(\boldsymbol{\theta})$ that is unbiased, differentiable with respect to $\boldsymbol{\theta}$ and has low variance is the key to train VAEs. SGVB uses an estimator $\hat{\mathcal{L}}_L^R(\boldsymbol{\theta})$ that is based on reparametrization of $q(\mathbf{z}|\mathbf{x})$ and sampling [1]. However, as described in section II, this method places many restrictions on the form of $q(\mathbf{z}|\mathbf{x})$ and fails if $q(\mathbf{z}|\mathbf{x})$ can not be reparametrized. This is the difficulty if $\mathbf{z}$ is discrete.

In this paper, we propose a simple and differentiable estimator $\hat{\mathcal{L}}_L^I(\boldsymbol{\theta})$ for $\mathcal{L}_L(\boldsymbol{\theta})$ that is based on importance sampling. Because no reparametrization is needed, it can be used to train VAEs with binary or categorical latent representations. Compared to previously proposed methods like the Vector Quantised-Variational Autoencoder (VQ-VAE) [9] based on a straight-through estimator for the gradient of $\mathcal{L}_L(\boldsymbol{\theta})$ [10] or methods based on continuous relaxation of discrete variables [11–13], our proposed estimator has two advantages: It is unbiased and, compared to methods based on continuous relaxation, there is no need to tune additional parameters like the temperature term of the Gumbel-Softmax [12].

II. Estimating the expected log-likelihood with reparametrized sampling

The standard estimator $\hat{\mathcal{L}}_L^R(\boldsymbol{\theta})$ proposed in [1] is based on reparametrized sampling

$$\frac{\partial}{\partial \boldsymbol{\theta}} \mathcal{L}_L(\boldsymbol{\theta}) = \frac{\partial}{\partial \boldsymbol{\theta}} \mathrm{E}_{q(\mathbf{z}|\mathbf{x})} \left[\ln p(\mathbf{x}|\mathbf{z})\right] \quad (7)$$

$$= \frac{\partial}{\partial \boldsymbol{\theta}} \mathrm{E}_{p(\boldsymbol{\epsilon})} \left[\ln p(\mathbf{x}|\mathbf{z} = f(\boldsymbol{\epsilon}, \boldsymbol{\theta}))\right] \quad (8)$$

$$\approx \frac{\partial}{\partial \boldsymbol{\theta}} \frac{1}{M} \sum_{m=1}^{M} \ln p(\mathbf{x}|\mathbf{z} = f(\boldsymbol{\epsilon}_m, \boldsymbol{\theta})) \quad (9)$$

$$= \frac{\partial}{\partial \boldsymbol{\theta}} \hat{\mathcal{L}}_L^R(\boldsymbol{\theta}) \quad (10)$$

where $\boldsymbol{\epsilon}$ is a random variable with the distribution $p(\boldsymbol{\epsilon})$, $\boldsymbol{\epsilon}_m$ are samples from this distribution and $f(\boldsymbol{\epsilon}, \boldsymbol{\theta})$ is a reparametrization function such that $\mathbf{z} = f(\boldsymbol{\epsilon}, \boldsymbol{\theta}) \sim q(\mathbf{z}|\mathbf{x})$. This estimator can be used to train VAEs with SGVB if two conditions are fulfilled:

1) There exists a distribution $p(\boldsymbol{\epsilon})$ and a reparametrization function $f(\boldsymbol{\epsilon}, \boldsymbol{\theta})$ such that $\mathbf{z} = f(\boldsymbol{\epsilon}, \boldsymbol{\theta}) \sim q(\mathbf{z}|\mathbf{x})$.
2) The derivative of Eq. 6 must exist.

With Eq. 9, we obtain

$$\frac{\partial}{\partial \boldsymbol{\theta}} \hat{\mathcal{L}}_L^R(\boldsymbol{\theta}) = \frac{1}{M} \sum_{m=1}^{M} \frac{\partial}{\partial \mathbf{z}} \ln p(\mathbf{x}|\mathbf{z} = f(\boldsymbol{\epsilon}_m, \boldsymbol{\theta})) \frac{\partial}{\partial \boldsymbol{\theta}} \mathbf{z}. \quad (11)$$

This means both the reparametrization function $f(\boldsymbol{\epsilon}, \boldsymbol{\theta})$ and $\ln p(\mathbf{x}|\mathbf{z})$ must be differentiable with respect to $\boldsymbol{\theta}$ and $\mathbf{z}$, respectively, to allow direct backpropagation of the gradient through the reparametrized sampling operator. If these conditions are fulfilled, the gradient can flow directly from the output to the input layer of the VAE as shown in Fig. 2. Distributions over discrete latent representations $\mathbf{z}$ can not be reparametrized in this way. Therefore, this estimator can not be used to train VAEs with such representations.

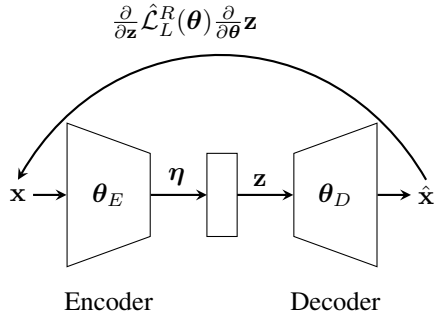


Fig. 2: The gradient flow through the VAE, using $\hat{\mathcal{L}}_L^R(\boldsymbol{\theta})$ based on reparametrized sampling. The gradient is propagated directly through the reparametrized sampling operator.

III. Estimating the expected log-likelihood with importance sampling

We propose an estimator $\hat{\mathcal{L}}_L^I(\boldsymbol{\theta})$ which is based on importance sampling and can also be used to train VAEs with binary or categorical latent representations $\mathbf{z}$. Expanding Eq. 6 leads to

$$\frac{\partial}{\partial \boldsymbol{\theta}} \mathcal{L}_L(\boldsymbol{\theta}) = \frac{\partial}{\partial \boldsymbol{\theta}} \int \ln p(\mathbf{x}|\mathbf{z}) q(\mathbf{z}|\mathbf{x}) \mathrm{d}\mathbf{z} \quad (12)$$

$$= \frac{\partial}{\partial \boldsymbol{\theta}} \int \ln p(\mathbf{x}|\mathbf{z}) \frac{q(\mathbf{z}|\mathbf{x})}{q_I(\mathbf{z})} q_I(\mathbf{z}) \mathrm{d}\mathbf{z} \quad (13)$$

$$\approx \frac{\partial}{\partial \boldsymbol{\theta}} \frac{1}{M} \left(\sum_{m=1}^{M} \ln p(\mathbf{x}|\mathbf{z}_m) \frac{q(\mathbf{z}_m|\mathbf{x})}{q_I(\mathbf{z}_m)} \right) \quad (14)$$

$$= \frac{\partial}{\partial \boldsymbol{\theta}} \hat{\mathcal{L}}_L^I(\boldsymbol{\theta}), \quad (15)$$

where $q_I(\mathbf{z})$ is an arbitrary distribution of the same form as $q(\mathbf{z}|\mathbf{x})$ and is independent from the parameters $\boldsymbol{\theta}$. $\mathbf{z}_m \sim q_I(\mathbf{z})$

are samples from this distribution. The estimator computes a weighted sum of the log-likelihood $\ln p(\mathbf{x}|\mathbf{z}_m)$ with the weighting $q(\mathbf{z}_m|\mathbf{x})/q_I(\mathbf{z}_m)$.

The benefit is that the log-likelihood $\ln p(\mathbf{x}|\mathbf{z}_m)$ depends on the decoder parameters $\boldsymbol{\theta}_D$ only and not on the encoder parameters $\boldsymbol{\theta}_E$ whereas the weighting $q(\mathbf{z}_m|\mathbf{x})/q_I(\mathbf{z}_m)$ depends only on $\boldsymbol{\theta}_E$ and not on $\boldsymbol{\theta}_D$. Therefore, calculation of the gradient of $\hat{\mathcal{L}}_L^I(\boldsymbol{\theta})$ can be separated

$$\frac{\partial}{\partial \boldsymbol{\theta}} \hat{\mathcal{L}}_L^I(\boldsymbol{\theta}) = \left[\frac{\partial}{\partial \boldsymbol{\theta}_E} \hat{\mathcal{L}}_L^I(\boldsymbol{\theta}), \frac{\partial}{\partial \boldsymbol{\theta}_D} \hat{\mathcal{L}}_L^I(\boldsymbol{\theta}) \right], \tag{16}$$

with

$$\frac{\partial}{\partial \boldsymbol{\theta}_E} \hat{\mathcal{L}}_L^I(\boldsymbol{\theta}) = \frac{1}{M} \sum_{m=1}^{M} \ln p(\mathbf{x}|\mathbf{z}_m) \frac{\partial}{\partial \boldsymbol{\theta}_E} \frac{q(\mathbf{z}_m|\mathbf{x})}{q_I(\mathbf{z}_m)} \tag{17}$$

$$\frac{\partial}{\partial \boldsymbol{\theta}_D} \hat{\mathcal{L}}_L^I(\boldsymbol{\theta}) = \frac{1}{M} \sum_{m=1}^{M} \frac{q(\mathbf{z}_m|\mathbf{x})}{q_I(\mathbf{z}_m)} \frac{\partial}{\partial \boldsymbol{\theta}_D} \ln p(\mathbf{x}|\mathbf{z}_m). \tag{18}$$

As shown in Fig. 3, gradient backpropagation is split into two separate parts. $\frac{\partial}{\partial \boldsymbol{\theta}_D} \hat{\mathcal{L}}_L^I(\boldsymbol{\theta})$ backpropagates the gradient of $\ln p(\mathbf{x}|\mathbf{z})$ from the output of the VAE to the sampling operator and $\frac{\partial}{\partial \boldsymbol{\theta}_E} \hat{\mathcal{L}}_L^I(\boldsymbol{\theta})$ backpropagates the gradient $\frac{q(\mathbf{z}|\mathbf{x})}{q_I(\mathbf{z})}$ from the sampling operator to the input layer of the VAE.

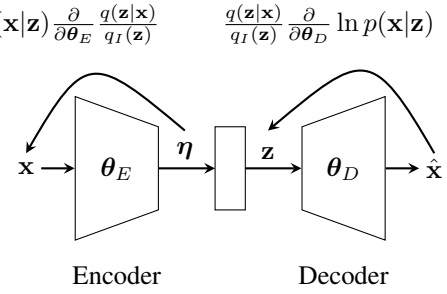


Fig. 3: Gradient flow through the VAE when using $\hat{\mathcal{L}}_L^I(\boldsymbol{\theta})$, based on importance sampling.

Compared to $\hat{\mathcal{L}}_L^R(\boldsymbol{\theta})$, we do not need to find a differentiable reparametrization for $q(\mathbf{z}|\mathbf{x})$, because we do not propagate the gradient through the sampling operator. Therefore, $\hat{\mathcal{L}}_L^I(\boldsymbol{\theta})$ can also be used if no reparametrization function exists for $q(\mathbf{z}|\mathbf{x})$. This is the case for a Bernoulli or a Categorical distribution of latent variables.

IV. VAE with Bernoulli distributed $\mathbf{z}$ (BVAE)

Assume that the latent representation $\mathbf{z}$ has i.i.d. Bernoulli distributed components, i.e. both the variational posterior distribution $q(\mathbf{z}|\mathbf{x})$ and $q_I(\mathbf{z})$ have the form

$$q(\mathbf{z}|\mathbf{x}) = \exp(\boldsymbol{\eta}^T \mathbf{z} - A(\boldsymbol{\eta})) \tag{19}$$

$$q_I(\mathbf{z}) = \exp(\boldsymbol{\xi}^T \mathbf{z} - A(\boldsymbol{\xi})), \tag{20}$$

where $\mathbf{z} \in \{0,1\}^c$, $\boldsymbol{\eta} = [\ln(q_1/(1-q_1)), ..., \ln(q_c/(1-q_c))]$ is the output vector of the encoder containing the logits of the independent Bernoulli distributions and $A(\boldsymbol{\eta}) = \mathbf{1}^T \ln(1+\mathrm{e}^{\boldsymbol{\eta}})$ is the corresponding log-partition function.

Hence, Eq. 17 is

$$\begin{aligned} \frac{\partial}{\partial \boldsymbol{\theta}_E} \hat{\mathcal{L}}_L^I(\boldsymbol{\theta}) &= \frac{1}{M} \sum_{m=1}^{M} \ln p(\mathbf{x}|\mathbf{z}_m) \frac{\partial}{\partial \boldsymbol{\eta}} \exp\left((\boldsymbol{\eta}-\boldsymbol{\xi})^T \mathbf{z}_m \right. \\ &\quad \left. - (A(\boldsymbol{\eta}) - A(\boldsymbol{\xi}))\right) \left(\frac{\partial}{\partial \boldsymbol{\theta}_E} \boldsymbol{\eta} \right) \qquad (21) \\ &= \frac{1}{M} \sum_{m=1}^{M} \ln p(\mathbf{x}|\mathbf{z}_m) \exp\left((\boldsymbol{\eta}-\boldsymbol{\xi})^T \mathbf{z}_m \right. \\ &\quad \left. - (A(\boldsymbol{\eta}) - A(\boldsymbol{\xi}))\right) (\mathbf{z}_m - \mathbf{q})^T \left(\frac{\partial}{\partial \boldsymbol{\theta}_E} \boldsymbol{\eta} \right) \qquad (22) \end{aligned}$$

where $\mathbf{q} = [q_1, ..., q_c]^T = \frac{1}{1+\mathrm{e}^{-\boldsymbol{\eta}}}$ contains the probabilities $q(z_i = 1|\mathbf{x})$. The variance of the estimator $\hat{\mathcal{L}}_L^I(\boldsymbol{\theta})$ heavily depends on the choice of the natural parameters $\boldsymbol{\xi}$ of the distribution $q_I(\mathbf{z})$. We choose $\boldsymbol{\xi} = \boldsymbol{\eta}$, leading to a gradient of the very simple form

$$\frac{\partial}{\partial \boldsymbol{\theta}_E} \hat{\mathcal{L}}_L^I(\boldsymbol{\theta}) = \frac{1}{M} \sum_{m=1}^{M} \ln p(\mathbf{x}|\mathbf{z}_m) (\mathbf{z}_m - \mathbf{q})^T \left(\frac{\partial}{\partial \boldsymbol{\theta}_E} \boldsymbol{\eta} \right). \tag{23}$$

This estimator is also known as the score function estimator of the gradient or as REINFORCE algorithm [14] and has the desirable property for training, which can be easily seen in the one dimensional case with $z \in \{0,1\}$. The mean of the estimator is

$$\begin{aligned} &\mathrm{E}_{q_I(z)} \left[\frac{\partial}{\partial \boldsymbol{\theta}_E} \hat{\mathcal{L}}_L^I(\boldsymbol{\theta}) \right] = \\ &= \mathrm{E}_{q_I(z)} \left[\frac{1}{M} \sum_{m=1}^{M} \ln p(\mathbf{x}|z_m) (z_m - q) \left(\frac{\partial}{\partial \boldsymbol{\theta}_E} \eta \right) \right] \qquad (24) \\ &= q(1-q)(\ln p(\mathbf{x}|z=1) - \ln(p(\mathbf{x}|z=0)) \left(\frac{\partial}{\partial \boldsymbol{\theta}_E} \eta \right) \qquad (25) \\ &= \frac{\partial}{\partial \boldsymbol{\theta}_E} \mathcal{L}_L(\boldsymbol{\theta}). \qquad (26) \end{aligned}$$

The estimator is unbiased.

V. VAE with Categorically distributed $\mathbf{z}$ (CVAE)

For categorically distributed $\mathbf{z}$, both the variational posterior distribution $q(\mathbf{z}|\mathbf{x})$ and $q_I(\mathbf{z})$ again have the form

$$q(\mathbf{z}|\mathbf{x}) = \exp(\boldsymbol{\eta}^T \mathbf{z} - A(\boldsymbol{\eta})) \tag{27}$$

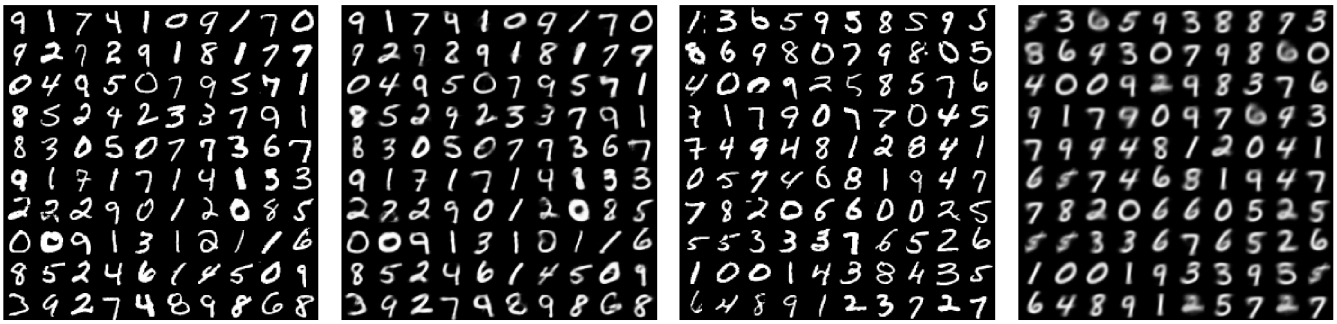
$$q_I(\mathbf{z}) = \exp(\boldsymbol{\xi}^T \mathbf{z} - A(\boldsymbol{\xi})), \tag{28}$$

but now $\mathbf{z} \in \{\mathbf{e}_1, ..., \mathbf{e}_c\}$ can assume only c different values. The vector of natural parameters is $\boldsymbol{\eta} = [\ln(p_1/p_c), ..., \ln(p_{c-1}/p_c), 0)]$ and the log partition function is $A(\boldsymbol{\eta}) = \ln \mathbf{1}^T \mathrm{e}^{\boldsymbol{\eta}}$.

With the formulas above, we arrive at the same form of the expected gradient of the log-likelihood

$$\frac{\partial}{\partial \boldsymbol{\theta}_E} \hat{\mathcal{L}}_L^I(\boldsymbol{\theta}) = \frac{1}{M} \sum_{m=1}^{M} \ln p(\mathbf{x}|\mathbf{z}_m) (\mathbf{z}_m - \mathbf{q})^T \left(\frac{\partial}{\partial \boldsymbol{\theta}_E} \boldsymbol{\eta} \right), \tag{29}$$

but now with $\mathbf{q} = \mathrm{softmax}(\boldsymbol{\eta})$ containing the probabilities $q_i = q(\mathbf{z} = \mathbf{e}_i)$ with $\sum_{i=1}^{c} q_i = 1$.



(a) Input FC_BVAE (b) Generated FC_BVAE (c) Input FC_CVAE (d) Generated FC_CVAE

Fig. 4: Test input images and generated handwritten images of the FC_BVAE and FC_CVAE

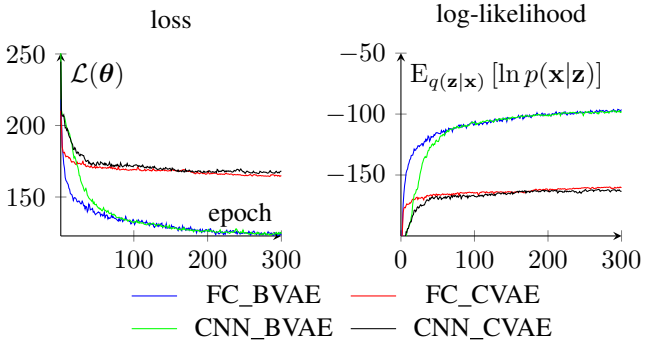


Fig. 5: Convergence of the loss, log-likelihood and gradient variance over 300 epochs of training on the MNIST dataset.

TABLE I: The architectural details of the trained VAEs. FC, Conv and Conv^{-1} are the fully connected, convolutional and deconvolutional layer, respectively. The shape of the convolutional layers is given in the form $\mathrm{DimA} \times \mathrm{DimB} \times \mathrm{Channel/Stride/Activation}$.

Architecture	In/Out	Encoder	Latent	Decoder
FC_BVAE	784	FC 1024/ReLu	$\mathbf{z} \in \mathbb{R}^c \sim \mathrm{Ber}(\frac{1}{1+\mathrm{e}^{-\boldsymbol{\eta}}})$	FC 1024/ReLu
or FC_CVAE		FC c/linear	or $\mathbf{z} \in \mathbb{R}^c \sim \mathrm{Cat}(\frac{\mathrm{e}^{\boldsymbol{\eta}}}{\mathrm{e}^{\mathbf{1}^T\boldsymbol{\eta}}})$	FC 784/sigmoid
CNN_BVAE	28x28x1	Conv 3x3x32/2/ReLu	$\mathbf{z} \in \mathbb{R}^c \sim \mathrm{Ber}(\frac{1}{1+\mathrm{e}^{-\boldsymbol{\eta}}})$	FC c/ReLu
or CNN_CVAE		Conv 3x3x64/2/ReLu	or $\mathbf{z} \in \mathbb{R}^c \sim \mathrm{Cat}(\frac{\mathrm{e}^{\boldsymbol{\eta}}}{\mathrm{e}^{\mathbf{1}^T\boldsymbol{\eta}}})$	reshape
		Conv 3x3x64/2/ReLu		Conv^{-1} 3x3x64/2/ReLu
		Conv 3x3x128/2/ReLu		Conv^{-1} 3x3x64/2/ReLu
		flatten		Conv^{-1} 3x3x32/2/ReLu
		FC c/linear		Conv^{-1} 3x3x1/2/sigmoid

VI. Experiments

In the following section, we show our preliminary experiments on the MNIST and Fashion MNIST datasets [15, 16]. Two different kinds of VAEs have been evaluated:

1) The BVAE with Bernoulli distributed $\mathbf{z} \in \{0,1\}^c$.
2) The CVAE with categorically distributed $\mathbf{z} \in \{\mathbf{e}_1, ..., \mathbf{e}_c\}$.

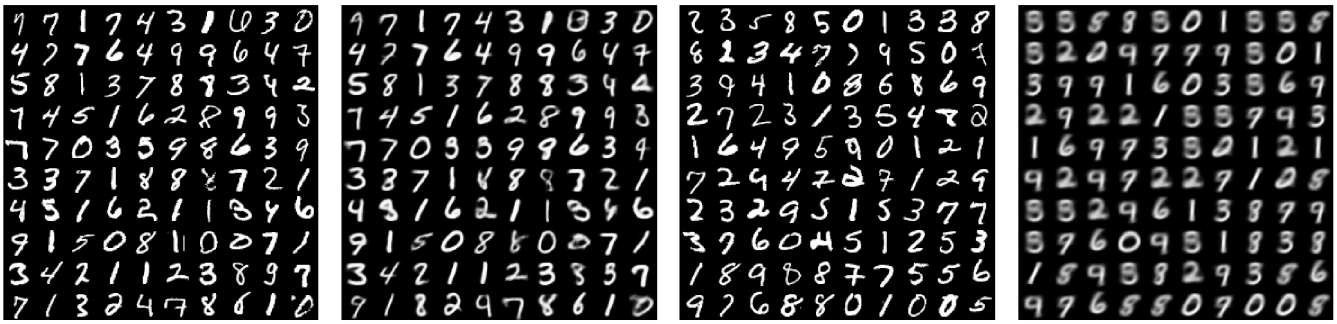
To train both architectures, the estimator $\hat{\mathcal{L}}_L^I(\boldsymbol{\theta})$ derived in Sec. IV and V is used.

Both BVAE and CVAE are tested with two different architectures given in Tab. I. The fully connected architecture has 2 dense encoder and decoder layers. The encoder and decoder networks of the convolutional architecture consist of 4 convolutional layers and one dense layer each.

In our first experiment, we train a FC_BVAE with $c = 50$, i.e. $\mathbf{z} \in \{0,1\}^{50}$ and a FC_CVAE with $c = 100$, i.e. $\mathbf{z} \in \{\mathbf{z}_1, ..., \mathbf{z}_{100}\}$. We train them for 300 epochs on the MNIST dataset, using SGVB with our proposed estimator $\hat{\mathcal{L}}_L^I(\boldsymbol{\theta})$, to estimate the expected log-likelihood, and ADAM as optimizer. Fig. 5 shows the convergence of the loss and the log-likelihood the VAEs assign to the training data $\ln p(\mathbf{x}|\mathbf{z})$ for a learning rate of 10^{-3} and a batch size of 2048. During training, the loss decreases steadily without oscillation. Furthermore, the results of the corresponding simulations with the CNN_BVAE and the CNN_CVAE are shown in Fig. 5.

The performance of the FC_CVAE is worse than the performance of the FC_BVAE. Training converges to a lower log-likelihood $\ln p(\mathbf{x}|\mathbf{z})$, because the maximal information content $H_{\mathrm{CVAE}}(\mathbf{z}) \leq \ln(100)$ of the latent variables of the FC_CVAE is much less than the maximal information content $H_{\mathrm{BVAE}}(\mathbf{z}) \leq c\ln(2)$ of the latent variables of the FC_BVAE. The FC_CVAE can at maximum learn to generate 100 different handwritten digits, what is a small number compared to the 2^{50} different images that the FC_BVAE can learn to generate.

Fig. 4 shows handwritten digits that are generated by the FC_BVAE and the FC_CVAE if we sample $\mathbf{z}$ from the variational posterior $q(\mathbf{z}|\mathbf{x})$. To draw samples from $q(\mathbf{z}|\mathbf{x})$, we feed test data which has not been seen during training to the encoders. The test data is shown in Fig. 4a and Fig. 4c. The corresponding reconstructions generated by the decoders are shown in Fig. 4b and Fig. 4d. Both input and reconstructed images are very similar in case of the FC_BVAE, meaning that it can approximate the data generating distribution $p(\mathbf{x})$ well. However, in case of the FC_CVAE, the generated digits are blurry and look very different than the input of the encoder. As shown in Fig. 6, our estimator is also applicable to deeper convolutional architectures. The resulting image quality of the generated images of the CNN_BVAE and CNN_CVAE is comparable to the one of the FC_BVAE and the FC_CVAE. Additionally, the input and reconstructed images for the second

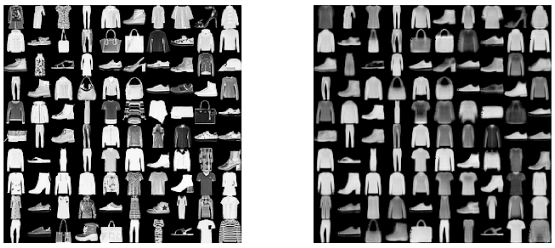


(a) Input CNN_BVAE (b) Generated CNN_BVAE (c) Input CNN_CVAE (d) Generated CNN_CVAE

Fig. 6: Test input images and generated handwritten images of the CNN_BVAE and CNN_CVAE

dataset of the FC_BVAE are shown in Fig. 7. Because of the higher complexity of the FMNIST dataset and the same low model capacity, the quality of the generated images is lower than for the MNIST dataset.

A major drawback of the FC_CVAE is that the latent space of the FC_CVAE can encode only very little information. Since the FC_CVAE can only learn to generate 100 different images, its decoder learns to generate template images that fit well to all the training images. We observe that some latent representations are decoded to meaningless patterns that just fit well to the data in avarage. However, the decoder also learned to generate at least one template image for each class of handwritten digits. Hence, the categorical latent representation could be interpreted as the cluster affiliation and the encoder of the FC_CVAE automatically learns to cluster the data. However, we think that they can be increased considerably if we allow a hybrid latent space with some continuous latent variables as proposed in [17]. This could lead to a powerful model for nonlinear clustering.



(a) Input FC_BVAE (b) Generated FC_BVAE

Fig. 7: Test input images and generated images of the FC_BVAE on the fashion MNIST dataset.

VII. Conclusion

In this paper, we derived an estimator for the gradient of the ELBO which does not rely on reparametrized sampling and therefore can be used to obtain differentiable estimates, even if reparametrization is not possible, e.g. if the latent variables $\mathbf{z}$ are Bernoulli or categorically distributed. We have shown the results with two different architectures on two datasets.

References

[1] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," *ArXiv e-prints*, Dec. 2013.

[2] D. P. Kingma *et al.*, "Improving variational inference with inverse autoregressive flow," *CoRR*, vol. abs/1606.04934, 2016. [Online]. Available: http://arxiv.org/abs/1606.04934

[3] T. White, "Sampling generative networks: Notes on a few effective techniques," *CoRR*, vol. abs/1609.04468, 2016. [Online]. Available: http://arxiv.org/abs/1609.04468

[4] J. An and S. Cho, "Variational autoencoder based anomaly detection using reconstruction probability," 2015.

[5] H. Xu *et al.*, "Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications," *CoRR*, vol. abs/1802.03903, 2018. [Online]. Available: http://arxiv.org/abs/1802.03903

[6] Z. Jiang *et al.*, "Variational deep embedding: A generative approach to clustering," *CoRR*, vol. abs/1611.05148, 2016. [Online]. Available: http://arxiv.org/abs/1611.05148

[7] N. Dilokthanakul *et al.*, "Deep unsupervised clustering with gaussian mixture variational autoencoders," *CoRR*, vol. abs/1611.02648, 2016. [Online]. Available: http://arxiv.org/abs/1611.02648

[8] V. E. Liong *et al.*, "Deep hashing for compact binary codes learning," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 2475–2483.

[9] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, "Neural discrete representation learning," *CoRR*, vol. abs/1711.00937, 2017. [Online]. Available: http://arxiv.org/abs/1711.00937

[10] Y. Bengio, N. Léonard, and A. C. Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation," *CoRR*, vol. abs/1308.3432, 2013. [Online]. Available: http://arxiv.org/abs/1308.3432

[11] W. Grathwohl *et al.*, "Backpropagation through the void: Optimizing control variates for black-box gradient estimation," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=SyzKd1bCW

[12] E. Jang *et al.*, "Categorical reparameterization with gumbel-softmax," 2017. [Online]. Available: https://arxiv.org/abs/1611.01144

[13] C. J. Maddison *et al.*, "The concrete distribution: A continuous relaxation of discrete random variables," *CoRR*, vol. abs/1611.00712, 2016. [Online]. Available: http://arxiv.org/abs/1611.00712

[14] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine Learning*, vol. 8, no. 3, pp. 229–256, May 1992. [Online]. Available: https://doi.org/10.1007/BF00992696

[15] Y. LeCun and C. Cortes, "MNIST handwritten digit database," 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[16] H. Xiao *et al.*, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *CoRR*, vol. abs/1708.07747, 2017. [Online]. Available: http://arxiv.org/abs/1708.07747

[17] X. Chen *et al.*, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," *CoRR*, vol. abs/1606.03657, 2016. [Online]. Available: http://arxiv.org/abs/1606.03657