Adversarial Variational Bayes:
Unifying Variational Autoencoders and Generative Adversarial Networks

Lars Mescheder [1]   **Sebastian Nowozin** [2]   **Andreas Geiger** [1 3]

Abstract

Variational Autoencoders (VAEs) are expressive latent variable models that can be used to learn complex probability distributions from training data. However, the quality of the resulting model crucially relies on the expressiveness of the inference model. We introduce Adversarial Variational Bayes (AVB), a technique for training Variational Autoencoders with arbitrarily expressive inference models. We achieve this by introducing an auxiliary discriminative network that allows to rephrase the maximum-likelihood-problem as a two-player game, hence establishing a principled connection between VAEs and Generative Adversarial Networks (GANs). We show that in the nonparametric limit our method yields an *exact* maximum-likelihood assignment for the parameters of the generative model, as well as the *exact* posterior distribution over the latent variables given an observation. Contrary to competing approaches which combine VAEs with GANs, our approach has a clear theoretical justification, retains most advantages of standard Variational Autoencoders and is easy to implement.

1. Introduction

Generative models in machine learning are models that can be trained on an unlabeled dataset and are capable of generating new data points after training is completed. As generating new content requires a good understanding of the training data at hand, such models are often regarded as a key ingredient to unsupervised learning.

In recent years, generative models have become more and more powerful. While many model classes such as Pixel-
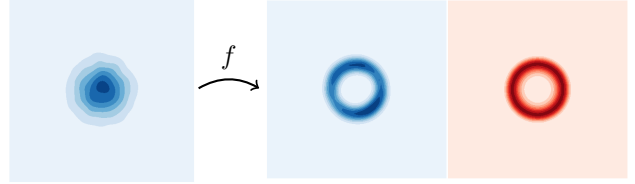
[1]Autonomous Vision Group, MPI Tübingen [2]Microsoft Research Cambridge [3]Computer Vision and Geometry Group, ETH Zürich. Correspondence to: Lars Mescheder <lars.mescheder@tuebingen.mpg.de>.

Figure 1. We propose a method which enables neural samplers with intractable density for Variational Bayes and as inference models for learning latent variable models. This toy example demonstrates our method's ability to accurately approximate complex posterior distributions like the one shown on the right.

RNNs (van den Oord et al., 2016b), PixelCNNs (van den Oord et al., 2016a), real NVP (Dinh et al., 2016) and Plug & Play generative networks (Nguyen et al., 2016) have been introduced and studied, the two most prominent ones are Variational Autoencoders (VAEs) (Kingma & Welling, 2013; Rezende et al., 2014) and Generative Adversarial Networks (GANs) (Goodfellow et al., 2014).

Both VAEs and GANs come with their own advantages and disadvantages: while GANs generally yield visually sharper results when applied to learning a representation of natural images, VAEs are attractive because they naturally yield both a generative model and an inference model. Moreover, it was reported, that VAEs often lead to better log-likelihoods (Wu et al., 2016). The recently introduced BiGANs (Donahue et al., 2016; Dumoulin et al., 2016) add an inference model to GANs. However, it was observed that the reconstruction results often only vaguely resemble the input and often do so only semantically and not in terms of pixel values.

The failure of VAEs to generate sharp images is often attributed to the fact that the inference models used during training are usually not expressive enough to capture the true posterior distribution. Indeed, recent work shows that using more expressive model classes can lead to substantially better results (Kingma et al., 2016), both visually and in terms of log-likelihood bounds. Recent work (Chen et al., 2016) also suggests that highly expressive inference models are essential in presence of a strong decoder to allow the model to make use of the latent space at all.

In this paper, we present Adversarial Variational Bayes

(AVB) [1] , a technique for training Variational Autoencoders with arbitrarily flexible inference models parameterized by neural networks. We can show that in the nonparametric limit we obtain a maximum-likelihood assignment for the generative model together with the correct posterior distribution.

While there were some attempts at combining VAEs and GANs (Makhzani et al., 2015; Larsen et al., 2015), most of these attempts are not motivated from a maximum-likelihood point of view and therefore usually do not lead to maximum-likelihood assignments. For example, in Adversarial Autoencoders (AAEs) (Makhzani et al., 2015) the Kullback-Leibler regularization term that appears in the training objective for VAEs is replaced with an adversarial loss that encourages the aggregated posterior to be close to the prior over the latent variables. Even though AAEs do *not* maximize a lower bound to the maximum-likelihood objective, we show in Section 6.2 that AAEs can be interpreted as an approximation to our approach, thereby establishing a connection of AAEs to maximum-likelihood learning.

Outside the context of generative models, AVB yields a new method for performing Variational Bayes (VB) with neural samplers. This is illustrated in Figure 1, where we used AVB to train a neural network to sample from a nontrivial unnormalized probability density. This allows to accurately approximate the posterior distribution of a probabilistic model, e.g. for Bayesian parameter estimation. The only other variational methods we are aware of that can deal with such expressive inference models are based on Stein Discrepancy (Ranganath et al., 2016; Liu & Feng, 2016). However, those methods usually do not directly target the reverse Kullback-Leibler-Divergence and can therefore not be used to approximate the variational lower bound for learning a latent variable model. Our contributions are as follows:

- We enable the usage of arbitrarily complex inference models for Variational Autoencoders using adversarial training.
- We give theoretical insights into our method, showing that in the nonparametric limit our method recovers the true posterior distribution as well as a true maximum-likelihood assignment for the parameters of the generative model.
- We empirically demonstrate that our model is able

---

[1] Concurrently to our work, several researchers have described similar ideas. Some ideas of this paper were described independently by Huszár in a blog post on http://www.inference.vc. The idea to use adversarial training to improve the encoder network was also suggested by Goodfellow in an exploratory talk he gave at NIPS 2016. A similar idea was also mentioned by Karaletsos (2016) in the context of message passing in graphical models .
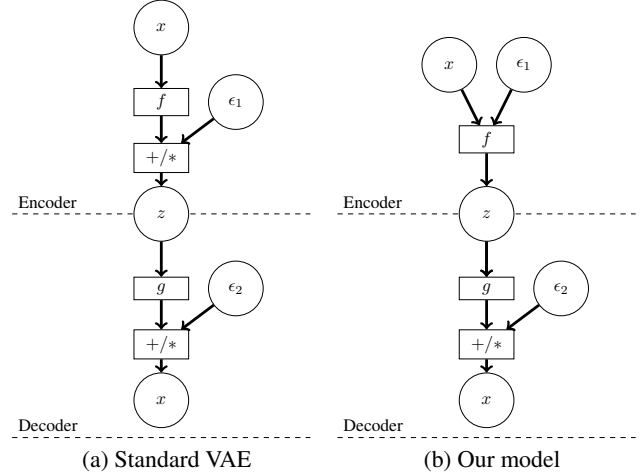

(a) Standard VAE   (b) Our model

Figure 2. Schematic comparison of standard VAE and VAE with black-box inference model, where ϵ_1 and ϵ_2 denote samples from some noise distribution. While more complicated inference models for Variational Autoencoders are possible, they are usually not as flexible as our black-box approach.

to learn rich posterior distributions and show that the model is able to generate compelling samples for complex data sets.

2. Background

As our model is an extension of Variational Autoencoders (VAEs) (Kingma & Welling, 2013; Rezende et al., 2014), we start with a brief review of VAEs.

VAEs are specified by a parametric generative model $p_\theta(x \mid z)$ of the visible variables given the latent variables, a prior $p(z)$ over the latent variables and an approximate inference model $q_\phi(z \mid x)$ over the latent variables given the visible variables. It can be shown that

$$\log p_\theta(x) \geq -\mathrm{KL}(q_\phi(z \mid x), p(z))$$
$$+ \mathrm{E}_{q_\phi(z|x)} \log p_\theta(x \mid z). \quad (2.1)$$

The right hand side of (2.1) is called the variational lower bound or evidence lower bound (ELBO). If there is ϕ such that $q_\phi(z \mid x) = p_\theta(z \mid x)$, we would have

$$\log p_\theta(x) = \max_\phi -\mathrm{KL}(q_\phi(z \mid x), p(z))$$
$$+ \mathrm{E}_{q_\phi(z|x)} \log p_\theta(x \mid z). \quad (2.2)$$

However, in general this is not true, so that we only have an inequality in (2.2).

When performing maximum-likelihood training, our goal is to optimize the marginal log-likelihood

$$\mathrm{E}_{p_\mathcal{D}(x)} \log p_\theta(x), \quad (2.3)$$

where $p_\mathcal{D}$ is the data distribution. Unfortunately, computing $\log p_\theta(x)$ requires marginalizing out z in $p_\theta(x, z)$

which is usually intractable. Variational Bayes uses inequality (2.1) to rephrase the intractable problem of optimizing (2.3) into

$$\max_\theta \max_\phi \mathrm{E}_{p_\mathcal{D}(x)}\Big[-\mathrm{KL}(q_\phi(z \mid x), p(z))$$
$$+ \mathrm{E}_{q_\phi(z|x)} \log p_\theta(x \mid z)\Big]. \quad (2.4)$$

Due to inequality (2.1), we still optimize a lower bound to the true maximum-likelihood objective (2.3).

Naturally, the quality of this lower bound depends on the expressiveness of the inference model $q_\phi(z \mid x)$. Usually, $q_\phi(z \mid x)$ is taken to be a Gaussian distribution with diagonal covariance matrix whose mean and variance vectors are parameterized by neural networks with x as input (Kingma & Welling, 2013; Rezende et al., 2014). While this model is very flexible in its dependence on x, its dependence on z is very restrictive, potentially limiting the quality of the resulting generative model. Indeed, it was observed that applying standard Variational Autoencoders to natural images often results in blurry images (Larsen et al., 2015).

3. Method

In this work we show how we can instead use a black-box inference model $q_\phi(z \mid x)$ and use adversarial training to obtain an approximate maximum likelihood assignment θ^* to θ and a close approximation $q_{\phi^*}(z \mid x)$ to the true posterior $p_{\theta^*}(z \mid x)$. This is visualized in Figure 2: on the left hand side the structure of a typical VAE is shown. The right hand side shows our flexible black-box inference model. In contrast to a VAE with Gaussian inference model, we include the noise ϵ_1 as additional input to the inference model instead of adding it at the very end, thereby allowing the inference network to learn complex probability distributions.

3.1. Derivation

To derive our method, we rewrite the optimization problem in (2.4) as

$$\max_\theta \max_\phi \mathrm{E}_{p_\mathcal{D}(x)} \mathrm{E}_{q_\phi(z|x)} \big(\log p(z)$$
$$- \log q_\phi(z \mid x) + \log p_\theta(x \mid z)\big). \quad (3.1)$$

When we have an explicit representation of $q_\phi(z \mid x)$ such as a Gaussian parameterized by a neural network, (3.1) can be optimized using the reparameterization trick (Kingma & Welling, 2013; Rezende & Mohamed, 2015) and stochastic gradient descent. Unfortunately, this is not the case when we define $q_\phi(z \mid x)$ by a black-box procedure as illustrated in Figure 2b.

The idea of our approach is to circumvent this problem by

implicitly representing the term

$$\log p(z) - \log q_\phi(z \mid x) \quad (3.2)$$

as the optimal value of an additional real-valued discriminative network $T(x, z)$ that we introduce to the problem.

More specifically, consider the following objective for the discriminator $T(x, z)$ for a given $q_\phi(x \mid z)$:

$$\max_T \mathrm{E}_{p_\mathcal{D}(x)} \mathrm{E}_{q_\phi(z|x)} \log \sigma(T(x, z))$$
$$+ \mathrm{E}_{p_\mathcal{D}(x)} \mathrm{E}_{p(z)} \log\left(1 - \sigma(T(x, z))\right). \quad (3.3)$$

Here, $\sigma(t) := (1 + \mathrm{e}^{-t})^{-1}$ denotes the sigmoid-function. Intuitively, $T(x, z)$ tries to distinguish pairs (x, z) that were sampled independently using the distribution $p_\mathcal{D}(x)p(z)$ from those that were sampled using the current inference model, i.e., using $p_\mathcal{D}(x)q_\phi(z \mid x)$.

To simplify the theoretical analysis, we assume that the model $T(x, z)$ is flexible enough to represent any function of the two variables x and z. This assumption is often referred to as the nonparametric limit (Goodfellow et al., 2014) and is justified by the fact that deep neural networks are universal function approximators (Hornik et al., 1989).

As it turns out, the optimal discriminator $T^*(x, z)$ according to the objective in (3.3) is given by the negative of (3.2).

Proposition 1. *For $p_\theta(x \mid z)$ and $q_\phi(z \mid x)$ fixed, the optimal discriminator T^* according to the objective in (3.3) is given by*

$$T^*(x, z) = \log q_\phi(z \mid x) - \log p(z). \quad (3.4)$$

Proof. The proof is analogous to the proof of Proposition 1 in Goodfellow et al. (2014). See the Supplementary Material for details. □

Together with (3.1), Proposition 1 allows us to write the optimization objective in (2.4) as

$$\max_{\theta,\phi} \mathrm{E}_{p_\mathcal{D}(x)} \mathrm{E}_{q_\phi(z|x)}\big(- T^*(x, z) + \log p_\theta(x \mid z)\big), \quad (3.5)$$

where $T^*(x, z)$ is defined as the function that maximizes (3.3).

To optimize (3.5), we need to calculate the gradients of (3.5) with respect to θ and ϕ. While taking the gradient with respect to θ is straightforward, taking the gradient with respect to ϕ is complicated by the fact that we have defined $T^*(x, z)$ indirectly as the solution of an auxiliary optimization problem which itself depends on ϕ. However, the following Proposition shows that taking the gradient with respect to the explicit occurrence of ϕ in $T^*(x, z)$ is not necessary:

Algorithm 1 Adversarial Variational Bayes (AVB)

1: $i \leftarrow 0$
2: **while** not converged **do**
3: Sample $\{x^{(1)}, \ldots, x^{(m)}\}$ from data distrib. $p_{\mathcal{D}}(x)$
4: Sample $\{z^{(1)}, \ldots, z^{(m)}\}$ from prior $p(z)$
5: Sample $\{\epsilon^{(1)}, \ldots, \epsilon^{(m)}\}$ from $\mathcal{N}(0, 1)$
6: Compute θ-gradient (eq. 3.9):
$$g_\theta \leftarrow \nabla_\theta \frac{1}{m} \sum_{i=1}^m \log p_\theta \left(x^{(i)} \mid z_\phi \left(x^{(i)}, \epsilon^{(i)} \right) \right)$$
7: Compute ϕ-gradient (eq. 3.9):
$$g_\phi \leftarrow \nabla_\phi \frac{1}{m} \sum_{i=1}^m \left[-T_\psi \left(x^{(i)}, z_\phi(x^{(i)}, \epsilon^{(i)}) \right) \right.$$
$$\left. + \log p_\theta \left(x^{(i)} \mid z_\phi(x^{(i)}, \epsilon^{(i)}) \right) \right]$$
8: Compute ψ-gradient (eq. 3.3) :
$$g_\psi \leftarrow \nabla_\psi \frac{1}{m} \sum_{i=1}^m \left[\log \left(\sigma(T_\psi(x^{(i)}, z_\phi(x^{(i)}, \epsilon^{(i)}))) \right) \right.$$
$$\left. + \log \left(1 - \sigma(T_\psi(x^{(i)}, z^{(i)})) \right) \right]$$
9: Perform SGD-updates for θ, ϕ and ψ:
$$\theta \leftarrow \theta + h_i\, g_\theta, \quad \phi \leftarrow \phi + h_i\, g_\phi, \quad \psi \leftarrow \psi - h_i\, g_\psi$$
10: $i \leftarrow i + 1$
11: **end while**

Proposition 2. *We have*
$$\mathrm{E}_{q_\phi(z|x)} \left(\nabla_\phi T^*(x, z) \right) = 0. \tag{3.6}$$

Proof. By Proposition 1,
$$\mathrm{E}_{q_\phi(z|x)} \left(\nabla_\phi T^*(x, z) \right)$$
$$= \mathrm{E}_{q_\phi(z|x)} \left(\nabla_\phi \log q_\phi(z \mid x) \right). \tag{3.7}$$

For an arbitrary family of probability densities q_ϕ we have

$$\mathrm{E}_{q_\phi} \left(\nabla_\phi \log q_\phi \right) = \int q_\phi(z) \frac{\nabla_\phi q_\phi(z)}{q_\phi(z)} \mathrm{d}z$$
$$= \nabla_\phi \int q_\phi(z) \mathrm{d}z = \nabla_\phi 1 = 0. \tag{3.8}$$

Together with (3.7), this implies (3.6). $\qquad\square$

Using the reparameterization trick (Kingma & Welling, 2013; Rezende et al., 2014), (3.5) can be rewritten in the form

$$\max_{\theta, \phi} \mathrm{E}_{p_{\mathcal{D}}(x)} \mathrm{E}_\epsilon \big(- T^*(x, z_\phi(x, \epsilon))$$
$$+ \log p_\theta(x \mid z_\phi(x, \epsilon)) \big) \tag{3.9}$$

for a suitable function $z_\phi(x, \epsilon)$. Together with Proposition 1, (3.9) allows us to take unbiased estimates of the gradients of (3.5) with respect to ϕ and θ.

3.2. Algorithm

In theory, Propositions 1 and 2 allow us to apply Stochastic Gradient Descent (SGD) directly to the objective in (2.4).

However, this requires keeping $T^*(x, z)$ optimal which is computationally challenging. We therefore regard the optimization problems in (3.3) and (3.9) as a two-player game. Propositions 1 and 2 show that any Nash-equilibrium of this game yields a stationary point of the objective in (2.4).

In practice, we try to find a Nash-equilibrium by applying SGD jointly to (3.3) and (3.9), see Algorithm 1. Here, we parameterize the neural network T with a vector ψ. Even though we have no guarantees that this algorithm converges, any fix point of this algorithm yields a stationary point of the objective in (2.4).

Note that optimizing (3.5) with respect to ϕ while keeping θ and T fixed makes the encoder network collapse to a deterministic function. This is also a common problem for regular GANs (Radford et al., 2015). It is therefore crucial to keep the discriminative T network close to optimality while optimizing (3.5). A variant of Algorithm 1 therefore performs several SGD-updates for the adversary for one SGD-update of the generative model. However, throughout our experiments we use the simple 1-step version of AVB unless stated otherwise.

3.3. Theoretical results

In Sections 3.1 we derived AVB as a way of performing stochastic gradient descent on the variational lower bound in (2.4). In this section, we analyze the properties of Algorithm 1 from a game theoretical point of view.

As the next proposition shows, global Nash-equilibria of Algorithm 1 yield global optima of the objective in (2.4):

Proposition 3. *Assume that T can represent any function of two variables. If (θ^*, ϕ^*, T^*) defines a Nash-equilibrium of the two-player game defined by (3.3) and (3.9), then*

$$T^*(x, z) = \log q_{\phi^*}(z \mid x) - \log p(z) \tag{3.10}$$

and (θ^*, ϕ^*) is a global optimum of the variational lower bound in (2.4).

Proof. The proof can be found in the Supplementary Material. $\qquad\square$

Our parameterization of $q_\phi(z \mid x)$ as a neural network allows $q_\phi(z \mid x)$ to represent almost any probability density on the latent space. This motivates

Corollary 4. *Assume that T can represent any function of two variables and $q_\phi(z \mid x)$ can represent any probability density on the latent space. If (θ^*, ϕ^*, T^*) defines a Nash-equilibrium for the game defined by (3.3) and (3.9), then*

1. *θ^* is a maximum-likelihood assignment*

2. *$q_{\phi^*}(z \mid x)$ is equal to the true posterior $p_{\theta^*}(z \mid x)$*

3. T^* is the pointwise mutual information between x and z, i.e.

$$T^*(x, z) = \log \frac{p_{\theta^*}(x, z)}{p_{\theta^*}(x) p(z)}. \qquad (3.11)$$

Proof. This is a straightforward consequence of Proposition 3, as in this case (θ^*, ϕ^*) optimizes the variational lower bound in (2.4) if and only if 1 and 2 hold. Inserting the result from 2 into (3.10) yields 3. □

4. Adaptive Contrast

While in the nonparametric limit our method yields the correct results, in practice $T(x, z)$ may fail to become sufficiently close to the optimal function $T^*(x, z)$. The reason for this problem is that AVB calculates a contrast between the two densities $p_{\mathcal{D}}(x) q_\phi(z \mid x)$ to $p_{\mathcal{D}}(x) p(z)$ which are usually very different. However, it is known that logistic regression works best for likelihood-ratio estimation when comparing two very similar densities (Friedman et al., 2001).

To improve the quality of the estimate, we therefore propose to introduce an auxiliary conditional probability distribution $r_\alpha(z \mid x)$ with known density that approximates $q_\phi(z \mid x)$. For example, $r_\alpha(z \mid x)$ could be a Gaussian distribution with diagonal covariance matrix whose mean and variance matches the mean and variance of $q_\phi(z \mid x)$.

Using this auxiliary distribution, we can rewrite the variational lower bound in (2.4) as

$$\mathrm{E}_{p_{\mathcal{D}}(x)} \Big[-\mathrm{KL}\left(q_\phi(z \mid x), r_\alpha(z \mid x)\right)$$
$$+ \mathrm{E}_{q_\phi(z|x)} \left(-\log r_\alpha(z \mid x) + \log p_\phi(x, z)\right) \Big]. \quad (4.1)$$

As we know the density of $r_\alpha(z \mid x)$, the second term in (4.1) is amenable to stochastic gradient descent with respect to θ and ϕ. However, we can estimate the first term using AVB as described in Section 3. If $r_\alpha(z \mid x)$ approximates $q_\phi(z \mid x)$ well, $\mathrm{KL}\left(q_\phi(z \mid x), r_\alpha(z \mid x)\right)$ is usually much smaller than $\mathrm{KL}\left(q_\phi(z \mid x), p(z)\right)$, which makes it easier for the adversary to learn the correct probability ratio.

We call this technique Adaptive Contrast (AC), as we are now contrasting the current inference model $q_\phi(z \mid x)$ to an adaptive distribution $r_\alpha(z \mid x)$ instead of the prior $p(z)$. Using Adaptive Contrast, the generative model $p_\theta(x \mid z)$ and the inference model $q_\phi(z \mid x)$ are trained to maximize

$$\mathrm{E}_{p_{\mathcal{D}}(x)} \mathrm{E}_{q_\phi(z|x)} \big(- T^*(x, z)$$
$$- \log r_\alpha(z \mid x) + \log p_\theta(x, z) \big), \quad (4.2)$$

where $T^*(x, z)$ is the optimal discriminator distinguishing samples from $r_\alpha(z \mid x)$ and $q_\phi(z \mid x)$.
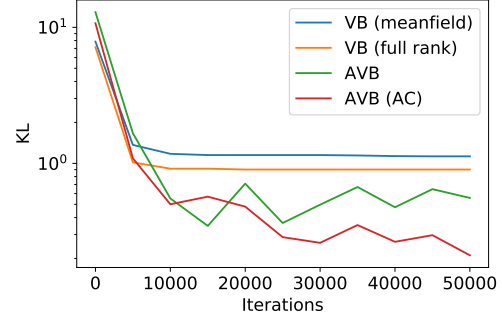

Figure 3. Comparison of KL to ground truth posterior obtained by Hamiltonian Monte Carlo (HMC).

Consider now the case that $r_\alpha(z \mid x)$ is given by a Gaussian distribution with diagonal covariance matrix whose mean $\mu(x)$ and variance vector $\sigma(x)$ match the mean and variance of $q_\phi(z \mid x)$. As the Kullback-Leibler divergence is invariant under reparameterization, the first term in (4.1) can be rewritten as

$$\mathrm{E}_{p_{\mathcal{D}}(x)} \mathrm{KL}\left(\tilde{q}_\phi(\tilde{z} \mid x), r_0(\tilde{z})\right) \qquad (4.3)$$

where $\tilde{q}_\phi(\tilde{z} \mid x)$ denotes the distribution of the normalized vector $\tilde{z} := \frac{z - \mu(x)}{\sigma(x)}$ and $r_0(\tilde{z})$ is a Gaussian distribution with mean 0 and variance 1. This way, the adversary only has to account for the deviation of $q_\phi(z \mid x)$ from a Gaussian distribution, not its location and scale.

In practice, we estimate $\mu(x)$ and $\sigma(x)$ using a Monte-Carlo estimate. In the Supplementary Material we describe a network architecture for $q_\phi(z \mid x)$ that makes the computation of this estimate particularly efficient.

5. Experiments

We tested our method both as a black-box method for variational inference and for learning generative models. The former application corresponds to the case where we fix the generative model and a data point x and want to learn the posterior $q_\phi(z \mid x)$.

An additional experiment on the celebA dataset (Liu et al., 2015) can be found in the Supplementary Material.

5.1. Variational Inference

When the generative model and a data point x is fixed, AVB gives a new technique for Variational Bayes with arbitrarily complex approximating distributions. We applied this to the "Eight School" example from Gelman et al. (2014). In this example, the coaching effects y_i, $i = 1, \ldots, 8$ for eight schools are modeled as

$$y_i \sim \mathcal{N}(\mu + \theta \cdot \eta_i, \sigma_i),$$

where μ, τ and the η_i are the model parameters to be inferred. We place a $\mathcal{N}(0, 1)$ prior on all the parameters
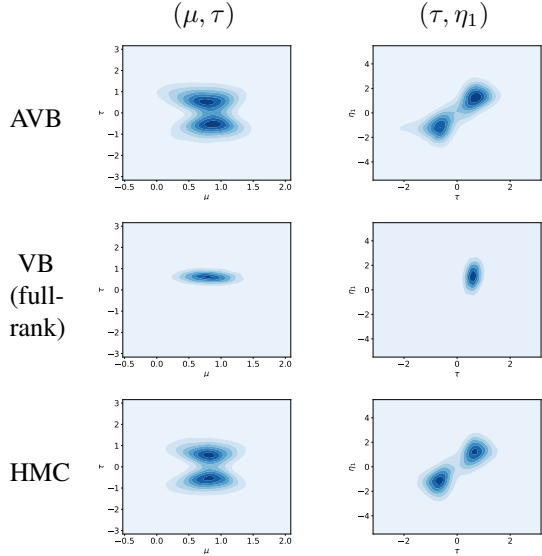
Figure 4. Comparison of AVB to VB on the "Eight Schools" example by inspecting two marginal distributions of the approximation to the 10-dimensional posterior. We see that AVB accurately captures the multi-modality of the posterior distribution. In contrast, VB only focuses on a single mode. The ground truth is shown in the last row and has been obtained using HMC.

of the model. We compare AVB against two variational methods with Gaussian inference model (Kucukelbir et al., 2015) as implemented in STAN (Stan Development Team, 2016). We used a simple two layer model for the posterior and a powerful 5-layer network with RESNET-blocks (He et al., 2015) for the discriminator. For every posterior update step we performed two steps for the adversary. The ground-truth data was obtained by running Hamiltonian Monte-Carlo (HMC) for 500000 steps using STAN.

We evaluate all methods by estimating the Kullback-Leibler-Divergence to the ground-truth data using the ITE-package (Szabo, 2013) applied to 10000 samples from the ground-truth data and the respective approximation. The resulting Kullback-Leibler divergence over the number of iterations for the different methods is plotted in Figure 3. We see that our method clearly outperforms the methods with Gaussian inference model. For a qualitative visualization, we also applied Kernel-density-estimation to the 2-dimensional marginals of the (μ, τ)- and (τ, η_1)-variables as illustrated in Figure 4. In contrast to variational Bayes with Gaussian inference model, our approach clearly captures the multi-modality of the posterior distribution. We also observed that Adaptive Contrast makes learning more robust and improves the quality of the resulting model.

5.2. Generative Models

Synthetic Example To illustrate the application of our method to learning a generative model, we trained the neural networks on a simple synthetic dataset containing only

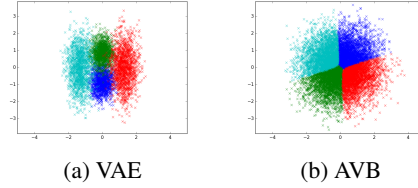

Figure 5. Training examples in the synthetic dataset.

(a) VAE (b) AVB

Figure 6. Distribution of latent code for VAE and AVB trained on synthetic dataset.

	VAE	AVB
log-likelihood	-1.568	**-1.403**
reconstruction error	$88.5 \cdot 10^{-3}$	$5.77 \cdot 10^{-3}$
ELBO	-1.697	$\approx$ -1.421
KL$(q_\phi(z), p(z))$	≈ 0.165	$\approx$ **0.026**

Table 1. Comparison of VAE and AVB on synthetic dataset. The optimal log-likelihood score on this dataset is $-\log(4) \approx -1.386$.

the 4 data points from the space of 2×2 binary images shown in Figure 5 and a 2-dimensional latent space. Both the encoder and decoder are parameterized by 2-layer fully connected neural networks with 512 hidden units each. The encoder network takes as input a data point x and a vector of Gaussian random noise ϵ and produces a latent code z. The decoder network takes as input a latent code z and produces the parameters for four independent Bernoulli-distributions, one for each pixel of the output image. The adversary is parameterized by two neural networks with two 512-dimensional hidden layers each, acting on x and z respectively, whose 512-dimensional outputs are combined using an inner product.

We compare our method to a Variational Autoencoder with a diagonal Gaussian posterior distribution. The encoder and decoder networks are parameterized as above, but the encoder does not take the noise ϵ as input and produces a mean and variance vector instead of a single sample.

We visualize the resulting division of the latent space in Figure 6, where each color corresponds to one state in the x-space. Whereas the Variational Autoencoder divides the space into a mixture of 4 Gaussians, the Adversarial Variational Autoencoder learns a complex posterior distribution. Quantitatively this can be verified by computing the KL-divergence between the prior $p(z)$ and the aggregated posterior $q_\phi(z) := \int q_\phi(z \mid x) p_\mathcal{D}(x) \mathrm{d}x$, which we estimate using the ITE-package (Szabo, 2013), see Table 1. Note that the variations for different colors in Figure 6 are solely due to the noise ϵ used in the inference model.

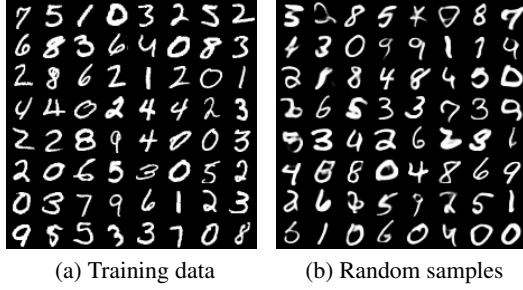The ability of AVB to learn more complex posterior models leads to improved performance as Table 1 shows. In

(a) Training data   (b) Random samples

Figure 7. Independent samples for a model trained on MNIST

	ELBO	AIS	
AVB (8-dim)	$(\approx -83.6 \pm 0.4)$	-91.2 ± 0.6	
AVB + AC (8-dim)	$\approx -96.3 \pm 0.4$	-89.6 ± 0.6	
AVB + AC (32-dim)	$\approx -79.5 \pm 0.3$	-80.2 ± 0.4	
VAE (8-dim)	-98.1 ± 0.5	-90.9 ± 0.6	
VAE (32-dim)	-87.2 ± 0.3	-81.8 ± 0.5	
VAE + HF (T=2)	-79.5	$-$	(Tomczak & Welling, 2016)
VAE + NF (T=80)	-85.1	$-$	(Rezende & Mohamed, 2015)
VAE + NICE (T=80)	-88.3	$-$	(Dinh et al., 2014)
VAE + HVI (T=16)	-88.3	$-$	(Salimans et al., 2015)
convVAE + HVI (T=16)	-84.1	$-$	(Salimans et al., 2015)
VAE + VGP (2hl)	-81.3	$-$	(Tran et al., 2015)
DRAW + VGP	-79.9	$-$	(Tran et al., 2015)
VAE + IAF	-80.8	$-$	(Kingma et al., 2016)
Auxiliary VAE (L=2)	-83.0	$-$	(Maaløe et al., 2016)

Table 2. Log-likelihoods on binarized MNIST for AVB and other methods improving on VAEs. We see that our method achieves state of the art log-likelihoods on binarized MNIST.

particular, AVB leads to a higher likelihood score that is close to the optimal value of $-\log(4)$ compared to a standard VAE that struggles with the fact that it cannot divide the latent space appropriately. Moreover, we see that the reconstruction error given by the mean cross-entropy between an input x and its reconstruction using the encoder and decoder networks is much lower when using AVB instead of a VAE with diagonal Gaussian inference model. We also observe that the estimated variational lower bound is close to the true log-likelihood, indicating that the adversary has learned the correct function.

MNIST In addition, we trained deep convolutional networks based on the DC-GAN-architecture (Radford et al., 2015) on the binarized MNIST-dataset (LeCun et al., 1998). For the decoder network, we use a 5-layer deep convolutional neural network. For the encoder network, we use a network architecture that allows for the efficient computation of the moments of $q_\phi(z \mid x)$. The idea is to define the encoder as a linear combination of learned basis noise vectors, each parameterized by a small fully-connected neural network, whose coefficients are parameterized by a neural network acting on x, please see the Supplementary Material for details. For the adversary, we replace the fully connected neural network acting on z and x with a fully connected 4-layer neural networks with 1024 units in each hidden layer. In addition, we added the result of neural networks acting on x and z alone to the end result.

To validate our method, we ran Annealed Importance Sampling (AIS) (Neal, 2001), the gold standard for evaluating decoder based generative models (Wu et al., 2016) with 1000 intermediate distributions and 5 parallel chains on 2048 test examples. The results are reported in Table 2. Using AIS, we see that AVB without AC overestimates the true ELBO which degrades its performance. Even though the results suggest that AVB with AC can also overestimate the true ELBO in higher dimensions, we note that the log-likelihood estimate computed by AIS is also only a lower bound to the true log-likelihood (Wu et al., 2016).

Using AVB with AC, we see that we improve both on a standard VAE and AVB without AC. When comparing to

other state of the art methods, we see that our method achieves state of the art results on binarized MNIST. Some random samples for MNIST are shown in Figure 7. We see that our model produces random samples that are perceptually close to the training set.

6. Related Work

6.1. Connection to Variational Autoencoders

Adversarial Variational Bayes strives to optimize the same objective as a standard Variational Autoencoder (Kingma & Welling, 2013; Rezende et al., 2014), but approximates the Kullback-Leibler divergence using an adversary instead of relying on a closed-form formula.

Substantial work has focused on making the class of approximate inference models more expressive. Normalizing flows (Rezende & Mohamed, 2015; Kingma et al., 2016) make the posterior more complex by composing a simple Gaussian posterior with an invertible smooth mapping for which the determinant of the Jacobian is tractable. Auxiliary Variable VAEs (Maaløe et al., 2016) add auxiliary variables to the posterior to make it more flexible. However, no other approach that we are aware of allows to use black-box inference models parameterized by a general neural network that takes as input a data point and a noise vector and produces a sample from the approximate posterior to optimize the ELBO.

6.2. Connection to Adversarial Autoencoders

Makhzani et al. (Makhzani et al., 2015) introduced the concept of Adversarial Autoencoders. The idea is to replace the term

$$\text{KL}(q_\phi(z \mid x), p(z)) \qquad (6.1)$$

in (2.4) with an adversarial loss that tries to enforce that upon convergence

$$\int q_\phi(z \mid x) p_{\mathcal{D}}(x) \mathrm{d}x \approx p(z). \qquad (6.2)$$

While related to our approach, the approach by Makhzani et al. modifies the variational objective while our approach retains the objective.

Interestingly, the approach by Makhzani et al. can be regarded as an approximation to our approach, where $T(x, z)$ is restricted to the class of functions that do not depend on x. Indeed, an ideal discriminator that only depends on z maximizes

$$\int \int p_{\mathcal{D}}(x)q(z \mid x) \log \sigma(T(z))\mathrm{d}x\mathrm{d}z$$
$$+ \int \int p_{\mathcal{D}}(x)p(z) \log (1 - \sigma(T(z)))\mathrm{d}x\mathrm{d}z \quad (6.3)$$

which is the case if and only if

$$T(z) = \log \int q(z \mid x)p_{\mathcal{D}}(x)\mathrm{d}x - \log p(z). \quad (6.4)$$

Clearly, this simplification is a crude approximation to our formulation from Section 3, but Makhzani et al. (2015) show that this method can still lead to good sampling results. In theory, restricting $T(x, z)$ in this way ensures that upon convergence we approximately have

$$\int q_\phi(z \mid x)p_{\mathcal{D}}(x)\mathrm{d}x = p(z), \quad (6.5)$$

but $q_\phi(z \mid x)$ need not be close to the true posterior $p_\theta(z \mid x)$. Intuitively, while mapping $p_{\mathcal{D}}(x)$ through $q_\phi(z \mid x)$ results in the correct marginal distribution, the contribution of each x to this distribution can be very inaccurate. When using this model as a generative model, the resulting sampler therefore has no guarantee to generate samples according to the true probability of the data points in the data distribution, even if the encoder and decoder networks are allowed to be arbitrarily complex.

6.3. Connection to f-GANs

Nowozin et al. (Nowozin et al., 2016) proposed to generalize Generative Adversarial Networks (Goodfellow et al., 2014) to f-divergences (Ali & Silvey, 1966) based on results by Nguyen et al. (Nguyen et al., 2010). In this paragraph we show that f-divergences allow to represent AVB as a zero-sum two-player game.

The family of f-divergences is given by

$$D_f(p\|q) = \mathrm{E}_p f \left(\frac{q(x)}{p(x)} \right). \quad (6.6)$$

for some convex functional $f : \mathbb{R} \to \mathbb{R}_\infty$ with $f(1) = 0$.

Nguyen et al. (2010) show that by using the convex conjugate f^* of f, (Hiriart-Urruty & Lemaréchal, 2013), we obtain

$$D_f(p\|q) = \sup_T \mathrm{E}_{q(x)} \left[T(x) \right] - \mathrm{E}_{p(x)} \left[f^*(T(x)) \right], \quad (6.7)$$

where T is a real-valued function. In particular, this is true for the reverse Kullback-Leibler divergence with $f(t) = t \log t$. We therefore obtain

$$\mathrm{KL}(q(z \mid x), p(z)) = D_f(p(z), q(z \mid x))$$
$$= \sup_T \mathrm{E}_{q(z|x)}T(x, z) - \mathrm{E}_{p(z)}f^*(T(x, z)), \quad (6.8)$$

with $f^*(\xi) = \exp(\xi - 1)$ the convex conjugate of $f(t) = t \log t$.

All in all, this yields

$$\max_\theta \mathrm{E}_{p_{\mathcal{D}}(x)} \log p_\theta(x) \quad (6.9)$$
$$= \max_{\theta,q} \min_T \mathrm{E}_{p_D(x)}\mathrm{E}_{p(z)}f^*(T(x, z))$$
$$+ \mathrm{E}_{p_D(x)}\mathrm{E}_{q(z|x)}(\log p_\theta(x \mid z) - T(x, z)).$$

By replacing the objective (3.3) for the discriminator with

$$\min_T \mathrm{E}_{p_D(x)} \left[\mathrm{E}_{p(z)}\mathrm{e}^{T(x,z)-1} - \mathrm{E}_{q(z|x)}T(x, z) \right], \quad (6.10)$$

we can reformulate the maximum-likelihood-problem as a mini-max zero-sum game. In fact, the derivations from Section 3 remain valid for any f-divergence that we use to train the discriminator. This is similar to the approach taken by Poole et al. (Poole et al., 2016) to improve the GAN-objective. In practice, we observed that the objective (6.10) results in unstable training. We therefore used the standard GAN-objective (3.3), which corresponds to the Jensen-Shannon-divergence.

6.4. Connection to BiGANs

BiGANs (Donahue et al., 2016; Dumoulin et al., 2016) are a recent extension to Generative Adversarial Networks with the goal to add an inference network to the generative model. Similarly to our approach, the authors introduce an adversary that acts on pairs (x, z) of data points and latent codes. However, whereas in BiGANs the adversary is used to optimize the generative and inference networks separately, our approach optimizes the generative and inference model jointly. As a result, our approach obtains good reconstructions of the input data, whereas for BiGANs we obtain these reconstructions only indirectly.

7. Conclusion

We presented a new training procedure for Variational Autoencoders based on adversarial training. This allows us to make the inference model much more flexible, effectively allowing it to represent almost any family of conditional distributions over the latent variables.

We believe that further progress can be made by investigating the class of neural network architectures used for the adversary and the encoder and decoder networks as well as finding better contrasting distributions.

Acknowledgements

References

Ali, Syed Mumtaz and Silvey, Samuel D. A general class of coefficients of divergence of one distribution from another. *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 131–142, 1966.

Chen, Xi, Kingma, Diederik P, Salimans, Tim, Duan, Yan, Dhariwal, Prafulla, Schulman, John, Sutskever, Ilya, and Abbeel, Pieter. Variational lossy autoencoder. *arXiv preprint arXiv:1611.02731*, 2016.

Dinh, Laurent, Krueger, David, and Bengio, Yoshua. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014.

Dinh, Laurent, Sohl-Dickstein, Jascha, and Bengio, Samy. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

Donahue, Jeff, Krähenbühl, Philipp, and Darrell, Trevor. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016.

Dumoulin, Vincent, Belghazi, Ishmael, Poole, Ben, Lamb, Alex, Arjovsky, Martin, Mastropietro, Olivier, and Courville, Aaron. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016.

Friedman, Jerome, Hastie, Trevor, and Tibshirani, Robert. *The elements of statistical learning*, volume 1. Springer series in statistics Springer, Berlin, 2001.

Gelman, Andrew, Carlin, John B, Stern, Hal S, and Rubin, Donald B. *Bayesian data analysis*, volume 2. Chapman & Hall/CRC Boca Raton, FL, USA, 2014.

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

He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, and Sun, Jian. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

Hiriart-Urruty, Jean-Baptiste and Lemaréchal, Claude. *Convex analysis and minimization algorithms I: fundamentals*, volume 305. Springer science & business media, 2013.

Hornik, Kurt, Stinchcombe, Maxwell, and White, Halbert. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

Karaletsos, Theofanis. Adversarial message passing for graphical models. *arXiv preprint arXiv:1612.05048*, 2016.

Kingma, Diederik P and Welling, Max. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kingma, Diederik P, Salimans, Tim, and Welling, Max. Improving variational inference with inverse autoregressive flow. *arXiv preprint arXiv:1606.04934*, 2016.

Kucukelbir, Alp, Ranganath, Rajesh, Gelman, Andrew, and Blei, David. Automatic variational inference in stan. In *Advances in neural information processing systems*, pp. 568–576, 2015.

Larsen, Anders Boesen Lindbo, Sønderby, Søren Kaae, and Winther, Ole. Autoencoding beyond pixels using a learned similarity metric. *arXiv preprint arXiv:1512.09300*, 2015.

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

Liu, Qiang and Feng, Yihao. Two methods for wild variational inference. *arXiv preprint arXiv:1612.00081*, 2016.

Liu, Ziwei, Luo, Ping, Wang, Xiaogang, and Tang, Xiaoou. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.

Maaløe, Lars, Sønderby, Casper Kaae, Sønderby, Søren Kaae, and Winther, Ole. Auxiliary deep generative models. *arXiv preprint arXiv:1602.05473*, 2016.

Makhzani, Alireza, Shlens, Jonathon, Jaitly, Navdeep, and Goodfellow, Ian. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

Neal, Radford M. Annealed importance sampling. *Statistics and Computing*, 11(2):125–139, 2001.

Nguyen, Anh, Yosinski, Jason, Bengio, Yoshua, Dosovitskiy, Alexey, and Clune, Jeff. Plug & play generative networks: Conditional iterative generation of images in latent space. *arXiv preprint arXiv:1612.00005*, 2016.

Nguyen, XuanLong, Wainwright, Martin J, and Jordan, Michael I. Estimating divergence functionals and the likelihood ratio by convex risk minimization. *IEEE*

Transactions on Information Theory, 56(11):5847–5861, 2010.

Nowozin, Sebastian, Cseke, Botond, and Tomioka, Ryota. f-gan: Training generative neural samplers using variational divergence minimization. *arXiv preprint arXiv:1606.00709*, 2016.

Poole, Ben, Alemi, Alexander A, Sohl-Dickstein, Jascha, and Angelova, Anelia. Improved generator objectives for gans. *arXiv preprint arXiv:1612.02780*, 2016.

Radford, Alec, Metz, Luke, and Chintala, Soumith. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Ranganath, Rajesh, Tran, Dustin, Altosaar, Jaan, and Blei, David. Operator variational inference. In *Advances in Neural Information Processing Systems*, pp. 496–504, 2016.

Rezende, Danilo Jimenez and Mohamed, Shakir. Variational inference with normalizing flows. *arXiv preprint arXiv:1505.05770*, 2015.

Rezende, Danilo Jimenez, Mohamed, Shakir, and Wierstra, Daan. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

Salimans, Tim, Kingma, Diederik P, Welling, Max, et al. Markov chain monte carlo and variational inference: Bridging the gap. In *ICML*, volume 37, pp. 1218–1226, 2015.

Stan Development Team. Stan modeling language users guide and reference manual, Version 2.14.0, 2016. URL http://mc-stan.org.

Szabo, Zoltán. Information theoretical estimators (ite) toolbox. 2013.

Tomczak, Jakub M and Welling, Max. Improving variational auto-encoders using householder flow. *arXiv preprint arXiv:1611.09630*, 2016.

Tran, Dustin, Ranganath, Rajesh, and Blei, David M. The variational gaussian process. *arXiv preprint arXiv:1511.06499*, 2015.

van den Oord, Aaron, Kalchbrenner, Nal, Espeholt, Lasse, Vinyals, Oriol, Graves, Alex, et al. Conditional image generation with pixelcnn decoders. In *Advances In Neural Information Processing Systems*, pp. 4790–4798, 2016a.

van den Oord, Aaron van den, Kalchbrenner, Nal, and Kavukcuoglu, Koray. Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759*, 2016b.

Wu, Yuhuai, Burda, Yuri, Salakhutdinov, Ruslan, and Grosse, Roger. On the quantitative analysis of decoder-based generative models. *arXiv preprint arXiv:1611.04273*, 2016.

Supplementary Material for
Adversarial Variational Bayes: Unifying Variational Autoencoders and Generative Adversarial Networks

Lars Mescheder [1] **Sebastian Nowozin** [2] **Andreas Geiger** [1 3]

Abstract

In the main text we derived Adversarial Variational Bayes (AVB) and demonstrated its usefulness both for black-box Variational Inference and for learning latent variable models. This document contains proofs that were omitted in the main text as well as some further details about the experiments and additional results.

Proofs

This section contains the proofs that were omitted in the main text.

The derivation of AVB in Section 3.1 relies on the fact that we have an explicit representation of the optimal discriminator $T^*(x, z)$. This was stated in the following Proposition:

Proposition 1. *For $p_\theta(x \mid z)$ and $q_\phi(z \mid x)$ fixed, the optimal discriminator T^* according to the objective in (3.3) is given by*

$$T^*(x, z) = \log q_\phi(z \mid x) - \log p(z). \qquad (3.4)$$

Proof. As in the proof of Proposition 1 in Goodfellow et al. (2014), we rewrite the objective in (3.3) as

$$\int \Big(p_\mathcal{D}(x) q_\phi(z \mid x) \log \sigma(T(x, z))$$
$$+ p_\mathcal{D}(x) p(z) \log(1 - \sigma(T(x, z))) \Big) \mathrm{d}x \mathrm{d}z. \quad (7.1)$$

This integral is maximal as a function of $T(x, z)$ if and only if the integrand is maximal for every (x, z). However, the function

$$t \mapsto a \log(t) + b \log(1 - t) \qquad (7.2)$$

attains its maximum at $t = \frac{a}{a+b}$, showing that

$$\sigma(T^*(x, z)) = \frac{q_\phi(z \mid x)}{q_\phi(z \mid x) + p(z)} \qquad (7.3)$$

or, equivalently,

$$T^*(x, z) = \log q_\phi(z \mid x) - \log p(z). \qquad (7.4)$$

$\square$

In Section 3.3 we characterized the Nash-equilibria of the two-player game defined by our algorithm. The following Proposition shows that in the nonparametric limit for $T(x, z)$ any Nash-equilibrium defines a global optimum of the variational lower bound:

Proposition 3. *Assume that T can represent any function of two variables. If (θ^*, ϕ^*, T^*) defines a Nash-equilibrium of the two-player game defined by (3.3) and (3.9), then*

$$T^*(x, z) = \log q_{\phi^*}(z \mid x) - \log p(z) \qquad (3.10)$$

and (θ^*, ϕ^*) is a global optimum of the variational lower bound in (2.4).

Proof. If (θ^*, ϕ^*, T^*) defines a Nash-equilibrium, Proposition 1 shows (3.10). Inserting (3.10) into (3.5) shows that (ϕ^*, θ^*) maximizes

$$\mathrm{E}_{p_\mathcal{D}(x)} \mathrm{E}_{q_\phi(z \mid x)} \big(-\log q_{\phi^*}(z \mid x) + \log p(z)$$
$$+ \log p_\theta(x \mid z) \big) \quad (7.5)$$

as a function of ϕ and θ. A straightforward calculation shows that (7.5) is equal to

$$\mathcal{L}(\theta, \phi) + \mathrm{E}_{p_\mathcal{D}(x)} \mathrm{KL}(q_\phi(z \mid x), q_{\phi^*}(z \mid x)) \qquad (7.6)$$

where

$$\mathcal{L}(\theta, \phi) := \mathrm{E}_{p_\mathcal{D}(x)} \Big[-\mathrm{KL}(q_\phi(z \mid x), p(z))$$
$$+ \mathrm{E}_{q_\phi(z \mid x)} \log p_\theta(x \mid z) \Big] \quad (7.7)$$

is the variational lower bound in (2.4).

Notice that (7.6) evaluates to $\mathcal{L}(\theta^*, \phi^*)$ when we insert (θ^*, ϕ^*) for (θ, ϕ).

Assume now, that (θ^*, ϕ^*) does not maximize the variational lower bound $\mathcal{L}(\theta, \phi)$. Then there is (θ', ϕ') with

$$\mathcal{L}(\theta', \phi') > \mathcal{L}(\theta^*, \phi^*). \qquad (7.8)$$

Inserting (θ', ϕ') for (θ, ϕ) in (7.6) we obtain

$$\mathcal{L}(\theta', \phi') + \mathrm{E}_{p_\mathcal{D}(x)} \mathrm{KL}(q_{\phi'}(z \mid x), q_{\phi^*}(z \mid x)), \qquad (7.9)$$

which is strictly bigger than $\mathcal{L}(\theta^*, \phi^*)$, contradicting the fact that (θ^*, ϕ^*) maximizes (7.6). Together with (3.10), this proves the theorem. $\qquad\square$

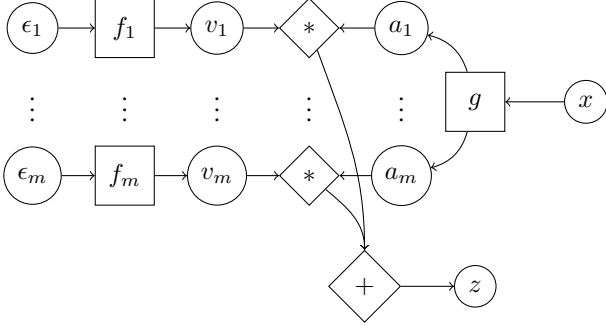## Architecture for MNIST-experiment

Figure 8. Architecture of the network used for the MNIST-experiment

To apply Adaptive Contrast to our method, we have to be able to efficiently estimate the moments of the current inference model $q_\phi(z \mid x)$. To this end, we propose a network architecture like in Figure 8. The final output z of the network is a linear combination of basis noise vectors where the coefficients depend on the data point x, i.e.

$$z_k = \sum_{i=1}^{m} v_{i,k}(\epsilon_i) a_{i,k}(x). \qquad (7.10)$$

The noise basis vectors $v_i(\epsilon_i)$ are defined as the output of small fully-connected neural networks f_i acting on normally-distributed random noise ϵ_i, the coefficient vectors $a_i(x)$ are defined as the output of a deep convolutional neural network g acting on x.

The moments of the z_i are then given by

$$\mathrm{E}(z_k) = \sum_{i=1}^{m} \mathrm{E}[v_{i,k}(\epsilon_i)] a_{i,k}(x). \qquad (7.11)$$

$$\mathrm{Var}(z_k) = \sum_{i=1}^{m} \mathrm{Var}[v_{i,k}(\epsilon_i)] a_{i,k}(x)^2. \qquad (7.12)$$

By estimating $\mathrm{E}[v_{i,k}(\epsilon_i)]$ and $\mathrm{Var}[v_{i,k}(\epsilon_i)]$ via sampling once per mini-batch, we can efficiently compute the moments of $q_\phi(z \mid x)$ for all the data points x in a single mini-batch.

Experiment on celebA

We also used AVB (without AC) to train a deep convolutional network on the celebA-dataset (Liu et al., 2015) for a 64-dimensional latent space with $\mathcal{N}(0, 1)$-prior. For the decoder and adversary we use two deep convolutional

(a) Training data   (b) Random samples

Figure 9. Independent samples for a model trained on celebA.

Figure 10. Interpolation experiments for celebA

neural networks acting on x like in Radford et al. (2015). We add the noise ϵ and the latent code z to each hidden layer via a learned projection matrix. Moreover, in the encoder and decoder we use three RESNET-blocks (He et al., 2015) at each scale of the neural network. We add the log-prior $\log p(z)$ explicitly to the adversary $T(x, z)$, so that it only has to learn the log-density of the inference model $q_\phi(z \mid x)$.

The samples for celebA are shown in Figure 9. We see that our model produces visually sharp images of faces. To demonstrate that the model has indeed learned an abstract representation of the data, we show reconstruction results and the result of linearly interpolating the z-vector in the latent space in Figure 10. We see that the reconstructions are reasonably sharp and the model produces realistic images for all interpolated z-values.