Adversarial training of partially invertible variational autoencoders

Thomas Lucas, Konstantin Shmelkov, Karteek Alahari, Cordelia Schmid, Jakob Verbeek

Adversarial training of partially invertible variational autoencoders

Thomas Lucas * [1] **Konstantin Shmelkov** * [1] **Karteek Alahari** [1] **Cordelia Schmid** [1] **Jakob Verbeek** [1]

Abstract

Adversarial generative image models yield outstanding sample quality, but suffer from two drawbacks: (i) they mode-drop, *i.e.*, do not cover the full support of the target distribution, and (ii) they do not allow for likelihood evaluations on held-out data. Conversely, maximum likelihood estimation encourages models to cover the full support of the training data, but yields poor samples. To address these mutual shortcomings, we propose a generative model that can be jointly trained with both procedures. In our approach, the conditional independence assumption typically made in variational autoencoders is relaxed by leveraging invertible models. This leads to improved sample quality, as well as improved likelihood on held-out data. Our model significantly improves on existing hybrid models, yielding GAN-like samples, and IS and FID scores that are competitive with fully adversarial models, while offering likelihoods measures on held-out data comparable to recent likelihood-based methods.

1. Introduction

Successful recent generative models of natural images can be divided into two broad families, which are trained in fundamentally different ways. The first is trained using likelihood-based criteria, which ensure that all training data points are well covered by the model. This category includes variational autoencoders (VAEs) (Kingma & Welling, 2014; Kingma et al., 2016), autoregressive models such as PixelCNNs (van den Oord et al., 2016; Salimans et al., 2017), and flow-based models such as Real-NVP (Dinh et al., 2017). The second category is trained based on a signal that measures to what extent (statistics of) samples from the model can be distinguished from (statistics of) the training data, *i.e.*, based on the quality of samples drawn from the model. This is the case for generative adversarial networks (GANs) (Goodfellow et al., 2014; Karras et al., 2018), as well as moment matching methods (Li et al., 2015).

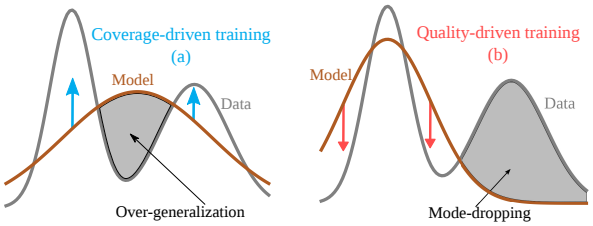

Figure 1: Illustration of coverage-driven (*i.e.*, maximum likelihood) and quality-driven (*i.e.*, adversarial) training, in a one-dimensional setting. The former pulls probability mass towards points from regions of high density of the distribution underlying the data, while the latter pushes mass out of low-density regions.

Motivation. Despite recent progress, existing methods exhibit a number of drawbacks. Likelihood-based models are trained to put probability mass on all elements of the training set. However, covering all modes of the training distribution forces models to over-generalize and assign probability mass on non-realistic images due to the lack of flexibility, as illustrated in Figure 1a. Limiting factors in such models include the use of fully factorized decoders in variational autoencoders, and restriction to the class of fully invertible functions in Real-NVP. Addressing these limitations is key to improving the sample quality.

Adversarial training on the other hand pushes samples to be indistinguishable from training images, at the expense of covering the full support of the training distribution. This phenomenon, known as "mode collapse" (Arjovsky et al., 2017), is illustrated in Figure 1b. Moreover, adversarial models have a low-dimensional support, so that held-out data typically has zero probability under the learned model. This, together with the lack of an inference mechanism prevents the use of likelihood to assess coverage of held-out data, and thus complicates evaluation of GANs.

Contribution. Prior attempts have been made to leverage the complementarity of quality and coverage driven training using an inference network, for instance the VAE-GAN model (Larsen et al., 2016), and approaches that learn an

*Equal contribution [1]Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LJK 38000 Grenoble, France. Correspondence to: Konstantin Shmelkov <konstantin.shmelkov@inria.fr>, Thomas Lucas <thomas.lucas@inria.fr>.

inference network adversarially (Dumoulin et al., 2017a; Donahue et al., 2017; Ulyanov et al., 2018). In contrast to these approaches, our model is directly optimized on a valid measure of log-likelihood performance in the RGB space, which we then report on a held-out dataset. As illustrated in Figure 2, our model uses non-volume preserving invertible transformations close to the output, optimized to increase the volume of data points. This relaxes naive independence assumptions on pixels given the latent variables, which are typical in VAEs. The invertibility of the feature is a crucial difference with Larsen et al. (2016), as it enables likelihood computations and ensures that separate data points cannot collapse in feature space. Experimental results show this extension to be beneficial for both the sample quality and the likelihood of held-out data. An adversarial loss is then used to explicitly optimize the sample quality.

We experimentally validate our approach on the CIFAR-10 dataset. Using the same architecture, our proposed model yields substantially improved samples over VAE models, as measured by the IS and FID scores, and improved likelihoods compared to a modified GAN model. Our model significantly improves upon existing hybrid models, producing GAN-like samples, and IS and FID scores that are competitive with fully adversarial models, while offering likelihoods on held-out data comparable to recent likelihood-based methods. We further confirm these observations with qualitative and quantitative experimental results on the CelebA dataset, STL-10, ImageNet, and LSUN-Bedrooms. We are the first to report IS and FID scores together with held-out likelihoods on all these five datasets. We also assess the performance of conditional versions of our models with the data augmentation based GAN evaluation procedure proposed in Shmelkov et al. (2018).

2. Related work

The complementary properties of auto-encoders and GANs have motivated several hybrid approaches. The VAE-GAN model of Larsen et al. (2016) uses the intermediate layers of a GAN discriminator as target space for the VAE. This model goes beyond the pixel-wise reconstruction loss. A drawback of this model, however, is that it does not define a density model in the image space.

Another line of research has focused on using adversarial methods to train an inference network. Dumoulin et al. (2017a) and Donahue et al. (2017) show that it is possible to learn an encoder and decoder model with a fully adversarial procedure. Given pairs of images and latent variable vectors, (x, z), a discriminator has to predict if z was encoded from a real image, or if x was decoded from a z sampled from the prior. A similar approach is taken by Chen et al. (2018), which showed that it is possible to approximate the symmetric KL in a fully adversarial setup,

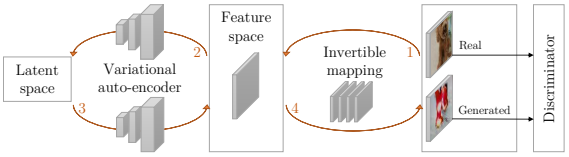

Figure 2: Overview of our model: An invertible non-linear mapping f maps an image x to a feature space with the same dimension (arrows 1 and 4). The encoder takes $f(x)$ and maps it to a posterior distribution $q_\phi(z|x)$ over the latent variable (arrow 2), while the decoder maps z to a distribution over the feature space (arrow 3). The discriminator $D(x)$ assesses sample quality in the image space.

and additionally uses reconstruction losses to improve the correspondence between reconstructed and target variables for x and z. Along the same line of research, Ulyanov et al. (2018) have shown that it is possible to collapse the encoder and the discriminator into one network, that encodes both real images and generated samples, and tries to spread their posteriors apart. Rosca et al. (2017) use a discriminator to replace the Kullback-Leibler divergence terms in the variational lower bound used to train VAEs with the density ratio trick. Makhzani et al. (2016) show that the regularization term on the latent variables of a VAE can be replaced with a discriminator that compares latent variables from the prior and from the posterior. This regularization is more flexible than the standard Kullback-Liebler divergence, but does not lead to a valid density measure on images either.

In the generative adversarial networks literature, mode-collapse has recently received considerable attention as one of the main failure modes of GANs. One line of research focuses on allowing the discriminator to access batch statistics of generated images, as pioneered by Salimans et al. (2016); Karras et al. (2018), and further generalized by Lucas et al. (2018); Lin et al. (2018). This is based on the idea that a batch of samples should behave like a batch of real images. For this to happen, individual samples should look realistic, but should also have as much variability as a batch of real images, which approximates a form of coverage-driven training as the batch size increases. These approaches, however, do not define likelihood or other measure to assess the model on held-out data, and suffer from typical adversarial training instabilities.

In our work, in contrast to these approaches, we focus on models which define a valid likelihood measure in the data space, which we report on held-out data. To achieve this, we leverage the merits of invertible transformations together with the VAE framework. This allows us to avoid the severely limiting conditional independence assumption commonly made in VAEs. Some recent work (Gulrajani et al., 2017b; Chen et al., 2017; Lucas & Verbeek, 2018) has proposed using autoregressive decoders to go beyond the VAE independence assumption in pixel space. They,

however, are not amenable to be used for adversarial training, since a prohibitively slow sequential pixel sampling is required in these models.

3. Preliminaries

In this section we briefly review coverage and quality driven training, and their respective shortcomings.

3.1. Maximum-likelihood and over-generalization

The de-facto standard approach for training generative models is maximum-likelihood estimation (MLE). It maximizes the probability of data sampled from an unknown data generating distribution p^* under the model p_θ w.r.t. the model parameters θ, which is equivalent to minimizing the Kullback-Liebler (KL) divergence, $\mathcal{D}_{\mathrm{KL}}(p^*||p_\theta)$, between p^* and p_θ. This yields models that tend to cover all the modes of the data, but put mass in spurious regions of the target space; a phenomenon known as *over-generalization* (Bishop, 2006), and manifested by unrealistic samples in the context of generative image models, see Figure 1a.

Over-generalization is inherent to the optimization of KL divergence. Real images are sampled from p^*, and p_θ is explicitly optimized to cover all of them. The training procedure, however, fails to sample from p_θ and evaluate the quality of these samples; ideally using the inaccessible $p^*(x)$ as a score. Therefore p_θ may put mass in spurious regions of the space without being heavily penalized. We refer to this kind of training procedure as "coverage-driven training" (CDT). This optimizes a loss of the form $\mathcal{L}_C(p_\theta) = \int_{x\in X} p^*(x) s_c(x, p_\theta)\mathrm{d}x$,

where $s_c(x, p_\theta) = \ln p_\theta(x)$ evaluates how well a sample x is covered by the model.

Explicitly evaluating sample quality is redundant in the regime of models with infinite capacity and infinite training data. Indeed, putting mass on spurious regions takes it away from the support of p^*, and thus reduces the likelihood of the training data. In practice, however, datasets and model capacity are finite, and models need to put mass outside the finite training set in order to generalize. The maximum likelihood criterion, by construction, only measures *how much* mass goes off the training data, not *where* it goes. In classic MLE, generalization is controlled in two ways: (i) inductive bias, in the form of model architecture, controls *where* the off-dataset mass goes, and (ii) regularization controls to which extent this happens. An adversarial loss, that considers samples from the model p_θ, can provide a second handle to control where the off-dataset mass goes. In contrast to model architecture design, an adversarial loss provides a "trainable" form of inductive bias.

3.2. Adversarial models and mode collapse

Adversarially trained models, *e.g.*, GANs (Goodfellow et al., 2014), produce samples of excellent quality compared to MLE. However, their main drawback is that they typically do not cover the full support of data. This well-recognized phenomenon is known as "mode-dropping" (Bishop, 2006; Arjovsky et al., 2017). Another drawback is that they do not provide a measure to assess mode-dropping, or their quantitative performance in general. The reasons for this are two-fold. First, defining a valid likelihood requires adding volume to the low-dimensional manifold learned by GANs to define a density under which training and test data have non-zero density. Second, computing the density of a data point under the defined probability distribution requires marginalizing out the latent variables, which is not trivial in the absence of an efficient inference mechanism.

When a human expert subjectively evaluates the quality of generated images, samples from the model are compared to the expert's implicit approximation of p^*. This type of objective can be written as $\mathcal{L}_Q(p_\theta) = \int_{x\in X} p_\theta(x) s_q(x, p^*)\mathrm{d}x$, and we refer to it as "quality-driven training" (QDT). To see that GANs use this type of training, recall that the discriminator is trained with the loss (Goodfellow et al., 2014)

$$\mathcal{L}_{\mathrm{GAN}} = \int_x p^*(x)\ln D(x) + p_\theta(x)\ln(1 - D(x))\mathrm{d}x. \quad (1)$$

It is easy to show that the optimal discriminator equals $D^*(x) = p^*(x)/(p^*(x) + p_\theta(x))$. Substituting the optimal discriminator, $\mathcal{L}_{\mathrm{GAN}}$ equals (up to additive and multiplicative constants) the Jensen-Shannon divergence

$$\begin{aligned}\mathcal{D}_{\mathrm{JS}}(p^*||p_\theta) = &\frac{1}{2}\mathcal{D}_{\mathrm{KL}}(p^*||\frac{1}{2}(p_\theta + p^*)) \\ &+\frac{1}{2}\mathcal{D}_{\mathrm{KL}}(p_\theta||\frac{1}{2}(p_\theta + p^*)).\end{aligned} \quad (2)$$

This loss, approximated by the discriminator, is symmetric and contains two KL divergence terms. Note that $\mathcal{D}_{\mathrm{KL}}(p_\theta||\frac{1}{2}(p_\theta + p^*))$ is an integral on p^*, so *coverage driven*. However the term that approximates it in Eq. (1), *i.e.*, $\int_x p^*(x)\ln D(x)$, is independent from the generative model, and disappears when differentiating. Therefore, it cannot be used to perform coverage-driven training, and the generator is trained to minimize $\log(1 - D(G(z)))$,[1] where $G(z)$ is the deterministic generator that maps latent variables z to the data space. Assuming $D = D^*$, this yields

$$\begin{aligned}\int_z \ln(1 - D^*(G(z))) &= \int_x p_\theta(x)\ln\frac{p_\theta(x)}{p_\theta(x) + p^*(x)} \\ &= \mathcal{D}_{\mathrm{KL}}(p_\theta||p_\theta + p^*),\end{aligned} \quad (3)$$

which is a quality-driven criterion. Both human assessment of generative models as well as the GAN objective are

[1] Or to maximize $\ln D(G(z))$ (Goodfellow et al., 2014).

quality-driven, and thus favor quality over support coverage. This may explain why images produced by GANs typically correlate well with human judgment.

4. Our approach

In this section we describe our generative model, and how we train it to ensure both quality and coverage.

4.1. Partially Invertible Variational Autoencoders

Adversarial training requires continuous sampling from the model during training. As VAEs and flow-based models allow for efficient feed-forward sampling, they are suitable likelihood-based models to build our approach on. VAEs rely on an inference network $q_\phi(z|x)$, or "encoder", to construct a variational evidence lower-bound (ELBO),

$$\begin{aligned}\mathcal{L}_{elbo}(x,\phi,\theta) = &\mathop{\mathbb{E}}_{q_\phi(z|x)}\left[\log(p_\theta(x|z))\right.\\ &\left.-\mathcal{D}_{\mathrm{KL}}(q_\phi(z|x)||p_\theta(z))\right] \leq \log p_\theta(x).\end{aligned} \tag{4}$$

The "decoder" $p_\theta(x|z)$ has a convolutional architecture similar to that of a GAN generator, with the exception that the decoder maps the latent variable z to a distribution over images, rather to a single image. Another difference is that in a VAE the prior $p_\theta(z)$ is typically learned, and more flexible than in a GAN, which can significantly improve the likelihoods on held-out data (Kingma et al., 2016). Our generative model uses a latent variable hierarchy with top-down sampling similar to Sønderby et al. (2016); Bachman (2016); Kingma et al. (2016), see Appendix A.1. It also leverages inverse auto-regressive flow (Kingma et al., 2016) to obtain accurate posterior approximations, beyond commonly used factorized Gaussian approximations, see Appendix A.2.

Typically, strong independence assumptions are also made on the decoder, *e.g.* by constraining it to a fully factorized Gaussian, *i.e.* $p_\theta(x|z) = \prod_{i=1} \mathcal{N}(x_i; \mu_i(z), \sigma_i(z))$. In this case, all dependency structure across the pixels has to be modeled by the latent variable z, any correlations not captured by z are treated as independent per-pixel noise. Unless z captures each and every aspect of the image structure, this is a poor model for natural images, and leads the model to over-generalize with independent per-pixel noise around blurry non-realistic examples. Using the decoder to produce a sparse Cholesky decomposition of the inverse covariance matrix alleviates this problem to some extent (Dorta et al., 2018), but retains a limiting assumption of linear-Gaussian dependency across pixel values.

Flow-based models offer a more flexible alternative, allowing to depart from Gaussian or other parametric distributions. Models such as NVP (Dinh et al., 2017) map an image $x \in X$ from RGB space to a latent code $y \in Y$ using a bijection $f : X \to Y$, and rely on the change of variable formula to compute the likelihood

$$p_X(x) = p_Y(f(x)) \left|\det\left(\frac{\partial f(x)}{\partial x^T}\right)\right|. \tag{5}$$

To sample x, we first sample y from a parametric prior, *e.g.* a unit Gaussian, and use the reverse mapping f^{-1} to find the corresponding x. Despite allowing for exact inference and efficient sampling, current flow-based approaches are worse than state-of-the-art likelihood-based approaches in terms held-out likelihood, and sample quality.

In our model we use invertible network layers to map RGB images to an abstract feature space $f(x)$. A VAE is then trained to model the distribution of $f(x)$. This results in a non-factorial and non-parametric form of $p_\theta(x|z)$ in the space of RGB images. See Figure 2 for a schematic illustration of the model. Although the likelihood of this model is intractable to compute, we can rely on a lower bound for training:

$$\begin{aligned}\mathcal{L}_C(p_\theta) &= -\mathop{\mathbb{E}}_{p^*(x)}\left[\mathcal{L}_{elbo}(f(x),\phi,\theta) + \log\left|\det\frac{\partial f(x)}{\partial x^T}\right|\right]\\ &\leq \mathop{\mathbb{E}}_{p^*(x)}\left[-\log p_\theta(x)\right].\end{aligned} \tag{6}$$

The bound is obtained by combining the VAE variational lower bound of Eq. (4) with the change of variable formula of Eq. (5). Our model combines benefits from VAE and NVP: it uses efficient non-invertible (convolutional) layers of VAE, while using a limited number of invertible layers as in NVP to avoid factorization in the conditional distribution $p_\theta(x|z)$. An alternative interpretation of our model is to see it as a variant of NVP with a complex non-parametric prior distribution rather than a unit Gaussian. The Jacobian in Eq. (6) pushes the model to increases the volume around training images in feature space, and the VAE measures their density in that space. Experimentally, we find our partially invertible non-factorial decoder to improve both sample quality as well as the likelihood of held-out data.

4.2. Improving samples with adversarial training

When optimizing $\mathcal{L}_C(p_\theta)$, the regularization term $D_{KL}(q_\phi(z|x)||p_\theta(z))$ drives the posterior $q_\phi(z|x)$ and the prior $p_\theta(z)$ closer together. Ideally, the posterior marginalized across real images and the prior match, *i.e.* $p^*_\theta(z) = \int_x p^*(x) q^*_\phi(z|x)$. In this is the case, latent variables $z \sim p_\theta(z)$, and mapped through the feedforward decoder, should result in realistic samples. Adversarial training be leveraged for quality-driven training of the prior, thus enrich its training signal as previously discussed.

For quality-driven training, we train a discriminator using the modified objective proposed by Sønderby et al. (2017) that combines both generator losses considered by Goodfel-

low et al. (2014):

$$\mathcal{L}_Q(p_\theta) = -\mathop{\mathbb{E}}_{p_\theta(z)} \ln \frac{D(G_\theta(z))}{1 - D(G_\theta(z))}. \tag{7}$$

Assuming the discriminator D is trained to optimality at every step, it is easy to demonstrate that the generator is trained to optimize $\mathcal{D}_{\mathrm{KL}}(p_\theta||p^*)$. To regularize the training of the discriminator, we use the gradient penalty introduced by Gulrajani et al. (2017a), see App. A.3 for details.

The training procedure, written as an algorithm in Appendix E, alternates between two steps, similar to that of GANs. In the first step, the discriminator is trained to maximize $\mathcal{L}_Q(p_\theta)$, bringing it closer to it's optimal value $\mathcal{L}_Q^*(p_\theta) = \mathcal{D}_{\mathrm{KL}}(p_\theta||p^*)$. In the second step, the generative model is trained to minimize $\mathcal{L}_C(p_\theta) + \mathcal{L}_Q(p_\theta)$, the sum of the coverage-based loss in Eq. (6), and the quality-based loss in Eq. (7). Assuming that the discriminator is trained to optimality at every step, the generator is trained to minimize a bound on the sum of two symmetric KL divergences:

$$\mathcal{L}_C(p_\theta) + \mathcal{L}_Q^*(p_\theta) \geq \mathcal{D}_{\mathrm{KL}}(p^*||p_\theta) + \mathcal{D}_{\mathrm{KL}}(p_\theta||p^*) + \mathcal{H}(p^*),$$

where the entropy of the data generating distribution, $\mathcal{H}(p^*)$, is an additive constant that does not depend on the learned generative model p_θ.

5. Experimental evaluation

Below, we present our evaluation protocol (Section 5.1), followed by an ablation study to assess the importance of the components of our model (Section 5.2). In Section 5.3 we improve quantitative and qualitative performance using recent advances from the VAE and GAN literature. We then compare to the state of the art on the CIFAR-10 and STL-10 datasets (Section 5.4), and present additional results at higher resolutions and on other datasets (Section 5.5). Finally, we evaluate a class-conditional version of our model using the image classification framework of Shmelkov et al. (2018).

5.1. Evaluation protocol

To evaluate the how well models cover held-out data, we use the bits per dimension (BPD) measure. This measure is defined as the negative log-likelihood on held-out data, averaged across pixels and color channels (Dinh et al., 2017). Due to their degenerate low-dimensional support, GANs do not define a valid density in the image space, which prevents measuring BPD. To endow a GAN with a full support and a valid likelihood, we train a VAE "around it". In particular, we train an isotropic noise parameter σ that does not depend on z, as in our VAE decoder, as well as an inference network. As we train these, the weights of the GAN generator are kept fixed. For both GANs and VAEs, we use the inference network to compute a lower-bound to approximate the likelihood, *i.e.* an upper bound on BPD.

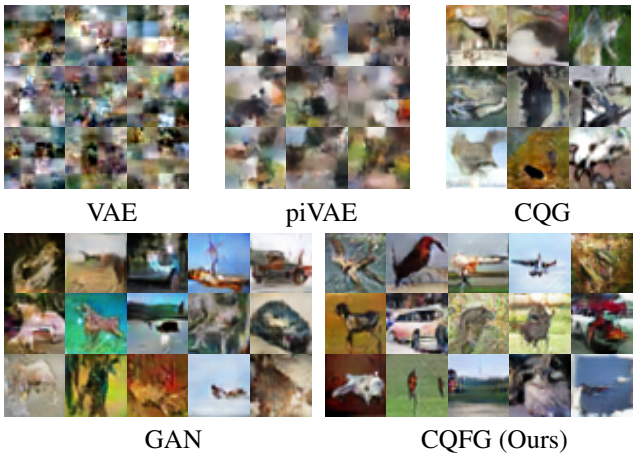

Figure 3: Samples from GAN and VAE baselines, and our CQG and CQFG models, all trained on CIFAR-10.

To evaluate the sample quality, we report Fréchet inception distance (FID) (Heusel et al., 2017) and inception score (IS) (Salimans et al., 2016), which are commonly used to quantitatively evaluate GANs (Zhang et al., 2018; Brock et al., 2019). Although IS and FID are used to evaluate sample quality, these metrics are also sensitive to coverage. In fact, any metric evaluating sample quality only would be degenerate, as collapsing to the mode of the target distribution would maximize it. However, in practice both metrics correlate stronger with sample quality than with support coverage, see Appendix B. We evaluate all measures using held-out data not used during training, which improves over common practice in the GAN literature, where train data is often used for evaluation.

5.2. Comparison to GAN and VAE baselines

Experimental setup. We evaluate our approach on the CIFAR-10 dataset, using 50k/10k train/test images of 32×32 pixels (standard split). We train our GAN baseline to optimize $\mathcal{L}_Q(p_\theta)$, and use the architecture of SNGAN (Miyato et al., 2018), which is stable and trains quickly. The same architecture and training hyper-parameters are used for all models in this experiments, see Appendix A for details.

We train our VAE baseline by optimizing $\mathcal{L}_C(p_\theta)$. We use the GAN generator architecture for the decoder, which produces the mean of a factorizing Gaussian distribution over pixel RGB values. We add a trainable isotropic variance σ, to ensure a valid density model. In the VAE model some feature maps in the decoder are treated as conditional latent variables, allowing for hierarchical top-down sampling. Experimentally, we find that similar top-down sampling is not effective for the GAN model.

To train the generator for both coverage and quality, we opti-

	$\mathcal{L}_Q$	$\mathcal{L}_C$	Flow	BPD ↓	IS ↑	FID ↓
GAN	✓			[7.0]	6.8	31.4
VAE		✓		4.4	2.0	171.0
piVAE		✓	✓	3.5	3.0	112.0
CQG	✓	✓		4.4	5.1	58.6
CQFG	✓	✓	✓	3.9	7.1	28.0

Table 1: Results for the GAN and VAE baselines, VAE with invertible flow layers (piVAE), and our models with (CQFG) and without (CQG) invertible layers. [Square brackets] denote that the value is approximated as described in Section 5.1.

	IAF	Res	BPD ↓	IS ↑	FID ↓
GAN			[7.0]	6.8	31.4
GAN		✓	—	7.4	24.0
CQFG			3.9	7.1	28.0
CQFG		✓	3.8	7.5	26.0
CQFG	✓	✓	3.8	7.9	20.1
CQFG (large-D)	✓	✓	3.7	8.1	18.6

Table 2: Evaluation of architectures using residual (Res) layers and inverse autoregressive flow (IAF) posterior approximation.

mize the sum of $\mathcal{L}_C(p_\theta)$ and $\mathcal{L}_Q(p_\theta)$. We refer to the model trained in this way as CQG. We refer to model that also includes invertible layers in the decoder as CQFG. The small invertible model uses a single scale with three invertible layers, each composed of two residual blocks and increases the number of weights in the generator by roughly 1.4% so we also slightly increase the width of the generator in the CQG version for fair comparison. All implementation details can be found in Appendix A. Our code will also be released upon publication for reproducibility.

Analysis of results. Form the experimental results in Table 1 we make several observations. As expected, the GAN baseline yields better sample quality (IS and FID) than the VAE baseline, *e.g.* obtaining inception scores of 6.8 and 2.0, respectively. Conversely, the VAE achieves better coverage, with a BPD of 4.4, compared to an estimated 7.0 for the GAN. The same generator trained for both quality and coverage, CQG, achieves *the same* BPD as the VAE baseline. The same quality of this model is in between that of the GAN and the VAE baselines. In Figure 3 we show samples from the different models, and these confirm the quantitative observations.

When adding the invertible layers to the VAE decoder, but using maximum likelihood training with $L_C(p_\theta)$ (piVAE), leads to improves sample quality with IS increasing from 2.0 to 3.0 and FID dropping from 171.0 to 112.0. Note that the quantitative sample quality is below that of the GAN baseline and our CQG model. When we combine the non-factorial decoder with coverage and quality driven training, CQFG, we obtain quantitative sample quality that is somewhat better than that of the GAN baseline: IS improving from 6.8 to 7.1, and FID decreasing from 31.4 to 28.0. The samples in Figure 3 confirm the high sample quality of the CQFG model. Note that the CQFG model also achieves a better BPD than the VAE baseline. These experimental observations demonstrate the importance of our contributions: our non-factorial decoder trained for coverage and quality improves both VAE and GAN in terms of held-out likelihood, and improves VAE sample quality to, or slightly beyond, that of the GAN.

In Appendix C we analyze the impact of the number of layers and scales in the invertible part of the decoder. In Appendix D we also provide reconstructions qualitatively demonstrating the inference abilities of our CQFG model. As is typical with expressive VAE model, ground-truth images and reconstructions are indistinguishable to the eye.

5.3. Evaluation of architectural refinements

To further improve quantitative and qualitative performance, we proceed to include two recent advances in the VAE and GAN literature. First, Gulrajani et al. (2017a) have shown a deeper discriminator with residual connections to be beneficial to training. We using such improved discriminators, we find it useful to make similar changes to the generator to mirror these modifications. Second, Kingma et al. (2016) improve VAE encoders by introducing inverse autoregressive flow (IAF) to allow for more accurate posterior approximations that go beyond factorized Gaussian approximations that are commonly used.

The results in Table 2 show consistent improvements across all metrics when adding residual connections and IAF. Increasing the size of the discriminator (denoted "large D") yields further improvements in IS and FID, while slightly degrading the BPD from 3.77 to 3.74. These results show

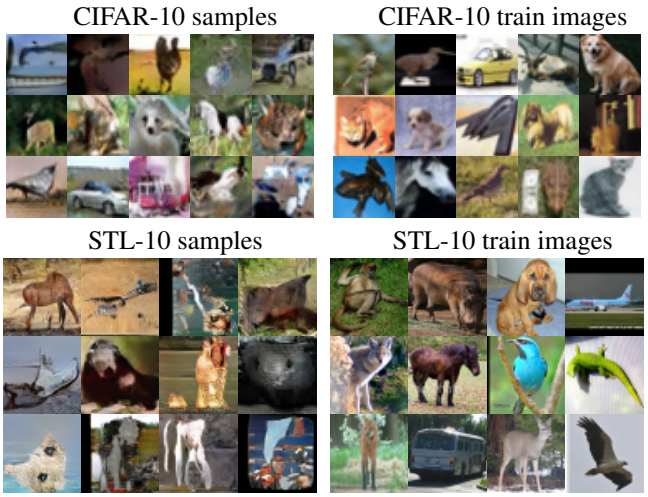

Figure 4: Random samples from our CQFG (large-D) trained on the CIFAR-10 and STL-10 (48×48) datasets.

that our model benefits from recent architectural advances in GANs and VAEs. In the remainder of our experiments we use the CQFG (large D).

5.4. Comparison to the state of the art

In Table 3 we compare the performance of our models with previously proposed hybrid approaches, as well as state-of-the-art adversarial and likelihood based models. Many entries in the table are missing, since the likelihood of held-out data is not defined for most adversarial methods, and most likelihood-based models do not report IS or FID scores. We present results for two variants of our CQFG model, the large-D variant from Table 2, as well as a variant that uses two scales in the invertible layers rather than one, denoted "S2". See Appendix C for details. The latter model achieves better BPD at the expense of worse IS and FID.

Compared to the best hybrid approaches, our large-D model yields a substantial improvement in IS to 8.1, while our S2 model yields a comparable value of 6.9. Compared to adversarially trained models, our large-D model obtains results that are comparable to the best results obtained by SNGAN using residual connections and hinge-loss. We note that the use of spectral normalization and hinge-loss for adversarial training could potentially improve our results, but we leave this for future work. Our S2 model is comparable to the basic SNGAN (somewhat better FID, somewhat worse IS) that does not use residual connections and hinge-loss. On STL-10 (48×48 pixels) our models trained using 100k/8k train/test images, also achieve competitive IS and FID scores; being only outperformed by SNGAN (Res-Hinge).

Using our S2 model we obtain a BPD of 3.5 that is comparable to Real-NVP, while our large-D model obtains a slightly worse value of 3.7. We computed IS and FID scores for VAE-IAF and PixelCNN++ using publicly released code and parameters. We find that these IS and FID scores are substantially worse than the ones we measured both of our model variants. To the best of our knowledge, we are the first to report BPD measurements on STL-10, and can therefore not compare to previous work in this metric.

We display samples from our CQFG (large-D) model on both datasets in Figure 4.

5.5. Results on additional datasets

To further validate our approach we train our CQFG (large-D) model on three additional datasets, and on STL-10 at 96×96 resolution. The architecture and training procedure are unchanged from the preceding experiments, up to the addition of convolutional layers to adapt to the increased resolution. For the CelebA dataset we used 196k/6.4k train/test images, resized to 96×96, and used central image crops

	CIFAR-10			STL-10		
	BPD ↓	IS ↑	FID ↓	BPD ↓	IS ↑	FID ↓
Hybrid models						
AGE		5.9				
ALI		5.3				
SVAE		6.8				
α-GAN		6.8				
SVAE-r		7.0				
CQFG (Ours)	3.7	8.1	18.6	4.0	8.6	52.7
CQFG (S2) (Ours)	3.5	6.9	28.9	3.8	8.6	52.1
Adversarial models						
SNGAN		7.4	29.3		8.3	53.1
BatchGAN		7.5	23.7		8.7	51
WGAN-GP		7.9				
SNGAN (Res-Hinge)		8.2	21.7		9.1	40.1
Likelihood-based models						
Real-NVP	3.5					
VAE-IAF	3.1	[3.8]	[73.5]			
PixelCNN++	2.9	[5.4]	[121.3]			

Table 3: Comparison of our models on CIFAR-10 and STL-10 (48×48) with state-of-the-art likelihood based, adversarial and hybrid generative models. [Square brackets] denote that we computed the values using samples obtained with the code and checkpoints released by the authors of the corresponding models. AGE: (Ulyanov et al., 2018), ALI: (Dumoulin et al., 2017a), SVAE: (Chen et al., 2018), SNGAN: (Miyato et al., 2018), BatchGAN: (Lucas et al., 2018), WGAN-GP: (Gulrajani et al., 2017a), NVP: (Dinh et al., 2017), VAE-IAF: (Kingma et al., 2016), PixelCNN++: (Salimans et al., 2017), α-GAN: (Rosca et al., 2017).

of both 178×178 and 96×96 pixels. We also train on STL-10 (100k/8k train/test images) resized to 96×96, on the LSUN-bedrooms dataset (3M/300 train/test images) at 64×64 resolution, and on ImageNet (1.2M/50k train/test images) resized to 64×64 pixels.

We show samples and train images for these datasets in Figure 5, and quantitative evaluation results in Table 4. The fact that our model works without changing the architecture and training hyper-parameters shows the stability of our approach. On the CelebA and LSUN datasets, our CQF generator produces compelling samples despite the high resolution of the images. The samples for STL-10 and ImageNet are less realistic due to the larger variability in these datasets; recall that we do not condition on class labels for generation. On CelebA, all scores improve significantly when using central crops of 96×96, due to the reduced variability in the smaller crop which removes the background from the images.

5.6. Class-conditional results

We also evaluate a class-conditional model and rely on two measures recently proposed by Shmelkov et al. (2018). The first measure, GAN-test, is obtained by training a classifier on natural image data and evaluating it on the samples of a

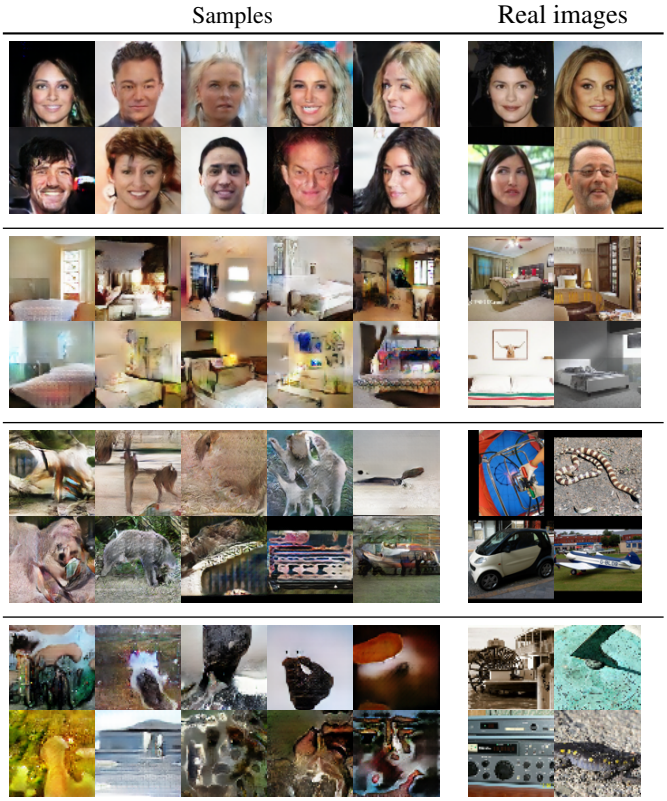

Figure 5: Samples and train images of CelebA (crop 178), LSUN-Bedrooms, STL-10 96×96, and ImageNet.

class-conditional generative model. This measure is sensitive to sample quality only. The second measure, GAN-train, is obtained by training a classifier on generated samples and evaluating it on natural images. This measure requires is sensitive both to quality and coverage. For a given GAN-test level, variations in GAN-train can be attributed to different coverage.

To perform this evaluation we develop a class conditional version of our CQFG model. The discriminator is conditioned using the class conditioning introduced by Miyato & Koyama (2018). GAN generators are typically made class-conditional using conditional batch normalization (De Vries et al., 2017; Dumoulin et al., 2017b), however batch normalization is known to be detrimental in VAEs (Kingma et al., 2016), as we verified in practice. To address this issue, we propose conditional weight normalization (CWN). As in weight normalization (Salimans & Kingma, 2016), we separate the training of the scale and the direction of the weight matrix. Additionally, the scaling factor $g(y)$ of the weight matrix $\mathbf{v}$ is conditioned on the class label y:

$$\mathbf{w} = \frac{g(y)}{\|\mathbf{v}\|}\mathbf{v}, \tag{8}$$

We also make the network biases conditional on the class label. Otherwise, the architecture is the same one used for the experiments in Table 1.

In Table 5 we report the GAN-train and GAN-test measures on CIFAR-10. Our CQFG model obtain a slightly higher GAN-test score than the GAN baseline, which shows that it achieves comparable if not better sample quality, which is inline with the results in terms of IS and FID scores in Section 5.2. Moreover, with CQFG we obtain a substantially better GAN-train score, going from 29.7 to 73.4. Having established similar GAN-test performance, this demonstrates significantly improved sample diversity of the CQFG model as compared to the GAN baseline. This shows that the coverage-driven training improves the coverage of the learned model.

model	GAN-test (%)	GAN-train (%)
GAN	71.8	29.7
CQFG	**76.9**	**73.4**
DCGAN†	58.2	65.0

Table 5: GAN-test and GAN-train measures for class conditional CQFG and GAN models on CIFAR-10. The performance of the DCGAN† model, though not directly comparable, is provided as a reference point.

	Resolution	BPD↓	IS↑	FID↓
CelebA, crop 178	96×96	2.85	—	24.3
CelebA, crop 96	96×96	2.45	—	13.8
STL-10	96×96	3.85	8.8	100.8
ImageNet	64×64	4.90	7.6	69.9
LSUN-Bedrooms	64×64	4.01	—	61.9

Table 4: Quantitative evaluation of our CQFG (large-D) model on additional datasets. IS is not reported on CelebA and LSUN because it is not informative on these datasets.

6. Conclusion

We presented CQGF, a generative model that leverages invertible network layers to relax the conditional pixel independence assumption commonly made in VAE models. Since our model allows for efficient feed-forward sampling, we are able to train our model using a maximum likelihood criterion that ensure coverage of the data generating distribution, as well as an adversarial criterion that ensures high sample quality. We provide quantitative and qualitative experimental results on a collection of five datasets (CIFAR-10, STL-10, CelebA, ImageNet and LSUN-Bedrooms). We obtain IS and FID scores comparable to state-of-the-art GAN models, and held-out likelihood scores that are comparable to recent pure likelihood-based models.

References

Arjovsky, M., Chintala, S., and Bottou, L. Wasserstein generative adversarial networks. In *ICML*, 2017.

Bachman, P. An architecture for deep, hierarchical generative models. In *NIPS*, 2016.

Bishop, C. *Pattern recognition and machine learning*. Spinger-Verlag, 2006.

Brock, A., Donahue, J., and Simonyan, K. Large scale GAN training for high fidelity natural image synthesis. In *ICLR*, 2019.

Chen, L., Dai, S., Pu, Y., Zhou, E., Li, C., Su, Q., Chen, C., and Carin, L. Symmetric variational autoencoder and connections to adversarial learning. In *AISTATS*, 2018.

Chen, X., Kingma, D., Salimans, T., Duan, Y., Dhariwal, P., Schulman, J., Sutskever, I., and Abbeel, P. Variational lossy autoencoder. In *ICLR*, 2017.

De Vries, H., Strub, F., Mary, J., Larochelle, H., Pietquin, O., and Courville, A. C. Modulating early visual processing by language. In *NIPS*, 2017.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using real NVP. In *ICLR*, 2017.

Donahue, J., Krähenbühl, P., and Darrell, T. Adversarial feature learning. In *ICLR*, 2017.

Dorta, G., Vicente, S., Agapito, L., Campbell, N. D. F., and Simpson, I. Structured uncertainty prediction networks. In *CVPR*, 2018.

Dumoulin, V., Belghazi, I., Poole, B., Lamb, A., Arjovsky, M., Mastropietro, O., and Courville, A. C. Adversarially learned inference. In *ICLR*, 2017a.

Dumoulin, V., Shlens, J., and Kudlur, M. A learned representation for artistic style. In *ICLR*, 2017b.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *NIPS*, 2014.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. Improved training of Wasserstein GANs. In *NIPS*, 2017a.

Gulrajani, I., Kumar, K., Ahmed, F., Taiga, A. A., Visin, F., Vazquez, D., and Courville, A. PixelVAE: A latent variable model for natural images. In *ICLR*, 2017b.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. In *NIPS*, 2017.

Karras, T., Aila, T., and abd J. Lehtinen, S. L. Progressive growing of GANSs for improved quality, stability, and variation. In *ICLR*, 2018.

Kingma, D. and Ba, J. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Kingma, D. P. and Welling, M. Auto-encoding variational Bayes. In *ICLR*, 2014.

Kingma, D. P., Salimans, T., Józefowicz, R., Chen, X., Sutskever, I., and Welling, M. Improving variational autoencoders with inverse autoregressive flow. In *NIPS*, 2016.

Larsen, A. B. L., Sønderby, S. K., and Winther, O. Autoencoding beyond pixels using a learned similarity metric. In *ICML*, 2016.

Li, Y., Swersky, K., and Zemel, R. S. Generative moment matching networks. In *ICML*, 2015.

Lin, Z., Khetan, A., Fanti, G., and Oh, S. PacGAN: The power of two samples in generative adversarial networks. In *NIPS*, 2018.

Lucas, T. and Verbeek, J. Auxiliary guided autoregressive variational autoencoders. In *ECML*, 2018.

Lucas, T., Tallec, C., Ollivier, Y., and Verbeek, J. Mixed batches and symmetric discriminators for GAN training. In *ICML*, 2018.

Makhzani, A., Shlens, J., Jaitly, N., and Goodfellow, I. Adversarial autoencoders. In *ICLR*, 2016.

Miyato, T. and Koyama, M. cGANs with projection discriminator. In *ICLR*, 2018.

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral normalization for generative adversarial networks. In *ICLR*, 2018.

Rosca, M., Lakshminarayanan, B., Warde-Farley, D., and Mohamed, S. Variational approaches for auto-encoding generative adversarial networks. *arXiv preprint arXiv:1706.04987*, 2017.

Salimans, T. and Kingma, D. P. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *NIPS*, 2016.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. Improved techniques for training GANs. In *NIPS*, 2016.

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

Shmelkov, K., Schmid, C., and Alahari, K. How good is my GAN? In *ECCV*, 2018.

Sønderby, C. K., Raiko, T., Maaløe, L., Sønderby, S. K., and Winther, O. Ladder variational autoencoders. In *NIPS*, 2016.

Sønderby, C. K., Caballero, J., Theis, L., Shi, W., and Huszár, F. Amortised MAP inference for image super-resolution. In *ICLR*, 2017.

Thanh-Tung, H., Tran, T., and Venkatesh, S. Improving generalization and stability of generative adversarial networks. In *ICLR*, 2019.

Ulyanov, D., Vedaldi, A., and Lempitsky, V. S. It takes (only) two: Adversarial generator-encoder networks. In *AAAI*, 2018.

van den Oord, A., Kalchbrenner, N., and Kavukcuoglu, K. Pixel recurrent neural networks. In *ICML*, 2016.

Zhang, H., Goodfellow, I., Metaxas, D., and Odena, A. Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*, 2018.

A. Model refinements and implementation details

A.1. Top-down sampling of hierarchical latent variables

Flexible priors and posteriors for the variational autoencoder model can be obtained by sampling hierarchical latent variables at different layers in the network. In the generative model p_θ, latent variables $\mathbf{z}$ can be split into L groups, each one at a different layer, and the density over $\mathbf{z}$ split thus:

$$q(\mathbf{z}) = q(\mathbf{z}_L) \prod_{i=1}^{L-1} q(\mathbf{z}_i|\mathbf{z}_{i+1})$$

Additionally, to allow the chain of latent variables to be sampled in the same order when encoding-decoding and when sampling, top-down sampling is used, as proposed in Sønderby et al. (2016); Bachman (2016); Kingma et al. (2016). With top-down sampling, the encoder (symmetric to the decoder) extracts deterministic features h_i at different levels as the image is being encoded, constituting the bottom-up deterministic pass. While decoding the image, these previously extracted deterministic features h_i are used for top-down sampling and help determining the posterior over latent variables at different depths in the decoder. These posteriors are also conditioned on the latent variables sampled at lower feature resolutions, using normal densities:

$$\begin{aligned} q_\phi(\mathbf{z}_1|x) &= \mathcal{N}(\mathbf{z}_1|\mu_1(x,h_1), \sigma_1^2(x), h_1) \\ q_\phi(\mathbf{z}_i|\mathbf{z}_{i-1}) &= \mathcal{N}(\mathbf{z}_i|\mu_i(x,\mathbf{z}_{i-1},h_{i-1}), \sigma_i^2(x,\mathbf{z}_{i-1},h_{i-1})) \end{aligned}$$

This constitutes the stochastic top-down pass.

We refer the reader to Sønderby et al. (2016); Bachman (2016); Kingma et al. (2016) for more detail.

A.2. Inverse autoregressive flow

To increase the flexibility of posteriors used over latent variables in variational inference, Kingma et al. (2016) has proposed a type of normalizing flow called inverse autoregressive flow (IAF). The main appeals of this normalizing flow are its scalability to high dimensionality and its ability to leverage autoregressive neural network (such as those introduced in van den Oord et al. (2016)). First, a latent variable vector is sampled using the reparametrization trick (Kingma & Welling, 2014):

$$\epsilon \sim \mathcal{N}(0, I) z_0 = \mu_0 + \sigma_0 \epsilon.$$

Then mean and variance parameters μ_1 and σ_1 are computed as functions of z_0 using autoregressive models, and a new latent variable z_1 is obtained:

$$z_1 = \mu_1(z_0) + \sigma_1(z_0) z_0.$$

Because σ_1 and μ_1 are implemented by auto-regressive networks, the jacobian $\frac{dz_1}{dz_0}$ is triangular with the values of σ_1 on the diagonal and the density under the new latent variable remains efficient to compute. In theory this transformation can be repeated an arbitrary number of times for increased flexibility, in practice typically a single step is used.

A.3. Gradient penalty

A body of work on Generative Adversarial Networks centers around the idea of regularizing the discriminator by enforcing Lipschitz continuity, for instance in Miyato et al. (2018); Arjovsky et al. (2017); Gulrajani et al. (2017a); Thanh-Tung et al. (2019). In this work we use the approach of Gulrajani et al. (2017a), that enforces the Lipschitz constraint with a gradient penalty term added to the loss:

$$\mathcal{L}_{Grad} = \lambda + \mathbb{E}_{\hat{x}}[(||\Delta_{\hat{x}} D(\hat{x})||_2 - 1)^2],$$

where $\hat{x}$ is obtained by interpolating between real and generated data:

$$\begin{aligned} \epsilon &\sim U_{[0,1]} \\ \hat{x} &= \epsilon x + (1-\epsilon)\tilde{x} \end{aligned}$$

We add this term to the loss used to train the discriminator that yields our quality driven criterion.

A.4. Architecture and training hyper-parameters

We used Adamax (Kingma & Ba, 2015) with learning rate 0.002, $\beta_1 = 0.9$, $\beta_2 = 0.999$ for all experiments. All CIFAR-10 experiments use batch size 64, other experiments in high resolution use batch size 32. To stabilize the adversarial training we use the gradient penalty (Gulrajani et al., 2017a) with coefficient 100, and 1 discriminator update per generator update. We experimented with different weighting coefficient between the two loss components, and found that values in the range 10 to 100 on the adversarial component work best in practice. In this range, no significant influence on the final performance of the model is observed, though the training dynamics in early training are improved with higher values. With values significantly smaller than 10, discriminator collapses was observed in a few isolated cases. All experiments reported here use coefficient 100.

For experiments with hierarchical latent variables we use 32 of them per layer. In the generator we use ELU nonlinearity, in discriminator with residual blocks we use ReLU while in simple convolutional discriminator we use leaky ReLU with slope 0.2.

Unless stated otherwise we use three NVP layers with a single scale and two residual blocks that we train only with the likelihood loss. Regardless of the number of scales, the VAE decoder always outputs a tensor of the same dimension as the target image, which is then fed to the NVP layers.

Just like in reference implementations we use both batch normalization and weight normalization in NVP and only weight normalization in IAF.

We use the reference implementations of IAF and NVP released by the authors.

Discriminator	Generator
	conv 3×3, 16
	IAF block 32
conv 3×3, 16	IAF block down 64
ResBlock 32	IAF block down 128
ResBlock down 64	IAF block down 256
ResBlock down 128	$h \sim \mathcal{N}(0; 1)$
ResBlock down 256	IAF block up 256
Average pooling	IAF block up 128
dense 1	IAF block up 64
	IAF block 32
	conv 3×3, 3

Table 6: Residual architectures for experiments from Table 2 and Table 8

B. On the Inception Score and the Fréchet inception distance

Quantitative evaluation of Generative Adversarial Networks is complicated by the absence of log-likelihood. The Inception Score (IS) and the Fréchet Inception Distance (FID) are metrics that have been proposed by (Salimans et al., 2016) and (Heusel et al., 2017) respectively, to automate the qualitative evaluation of samples. Though imperfect, these metrics have been shown to correlate well with human judgement in practice, and it is standard in the GAN literature to use them for quantitative evaluations.

The Inception Score (IS) (Salimans et al., 2016) is a statistic of the generated images, based on an external deep network trained for classification on ImageNet.

$$IS(p_\theta) = exp(\mathbb{E}_{x\sim p_\theta} D_{KL}(p(y|x)||p(y)))$$

Where $x \sim p_\theta$ is sampled from the generative model, $p(y|x)$ is the conditional class distributions obtained by applying the pretrained classification network to generated images, and $p(y) = \int_x p(y|x)p_\theta(x)$ is the class marginal over generated images.

The Fréchet Inception distance (FID) (Heusel et al., 2017) compares the distributions of Inception embeddings (activations from the penultimate layer of the Inception network) of real ($p_r(\mathbf{x})$) and generated ($p_g(\mathbf{x})$) images. Both of these distributions are modeled as multi-dimensional Gaussians parameterized by their respective mean and covariance. The distance measure is defined between the two Gaussian distributions as:

$$d^2((\mathbf{m}_r, \mathbf{C}_r), (\mathbf{m}_g, \mathbf{C}_g)) = \|\mathbf{m}_r - \mathbf{m}_g\|^2 + \mathrm{Tr}(\mathbf{C}_r + \mathbf{C}_g - 2(\mathbf{C}_r\mathbf{C}_g)^{\frac{1}{2}}), \quad (9)$$

where $(\mathbf{m}_r, \mathbf{C}_r)$, $(\mathbf{m}_g, \mathbf{C}_g)$ denote the mean and covariance of the real and generated image distributions respectively.

Practical Use. In practice, IS and FID correlate predominantly with the quality of samples. In the literature (mostly the generative adversarial networks literature, for instance Miyato et al. (2018)), they are considered to correlate well with human judgement of quality. An empirical indicator of that is that state-of-the art likelihood-based models have very low IS/FID scores despite having good coverage, which shows that the low quality of their samples dominates. Conversely, state-of-the art adversarial models have high IS/FID scores despite suffering from mode dropping (which strongly degrades BPD), so the score is determined mostly by the high quality of their samples. This is especially true when identical architectures and training budget are considered, as in our first experiment in Section 5.2.

Split size	IS	FID
50k (full)	11.3411	0.00
40k	11.3388	0.13
30k	11.3515	0.35
20k	11.3458	0.79
10k	11.3219	2.10
5k	11.2108	4.82
2.5k	11.0446	10.48

Table 7: IS and FID scores obtained by the ground truth when progressively dropping parts of the dataset. The metrics are largely insensitive to removing most of the dataset, unlike BPD. For reference, a reasonable GAN could get around 8 IS and 20 FID.

To obtain a quantitative estimation of how much entropy/coverage impacts the IS and FID measures, we evaluate the scores obtained by random subsamples of the dataset, such that the quality is unchanged but coverage progressively degraded (see details of the scores below). Table 7 shows that when using the full set (50k) images the FID is 0 as the distributions are identical. Notice that as the number of images decreases, IS is very stable (it can even increase, but by very low increments that fall below statistical noise, with a typical std on IS of 0.1). This is because the entropy of the distribution is not strongly impacted by subsampling, even though coverage is. FID is more sensitive, as it behaves more like a measure of coverage (it compares the two distributions). Nonetheless, the variations remain extremely low

even when dropping most of the dataset. For instance, when removing 80% of the dataset (*i.e.*, using $10k$ images), FID is at 2.10, to be compared with typical GAN/CQF values that are around 20. These measurement demonstrate that IS and FID scores are heavily dominated by the quality of images. From this, we conclude that IS and FID can be used as reasonable proxies to asses sample quality, even though they are also slightly influenced by coverage. One should bear in mind, however, that a small increase in these scores may come from better coverage rather than improved sample quality.

C. Qualitative influence of the feature space flexibility

In this section we experiment with different architectures to implement the invertible mapping used to build the feature space as presented in Section 4.1. To assess the impact of the expressiveness of the invertible model on the behavior of our framework, we modify various standard parameters of the architecture. Popular invertible models such as NVP (Dinh et al., 2017) readily offer the possibility of extracting latent representation at several scales, separating global factors of variations from low level detail, thus we experiment with varying number of scales. An other way of increasing the flexibility of the model is to change the number of residual blocks used in each invertible layer. Note that all the models evaluated so far in the main body of the paper are based on a single scale and two residual blocks. In addition to our CQFG models, we also compare with similar models trained with maximum likelihood estimation (MLE). Models are trained first with maximum-likelihood estimation, then with both coverage and quality driven criterions.

The results in Table 8 show that factoring out features at two scales rather than one is helpful in terms of BPD. For the CQFG models, however, the IS and FID deteriorate with more scales, and so a tradeoff between must be struck. For the MLE models, the visual quality of samples also improves when using multiple scales, as reflected in better IS and FID scores. Their quality, however, remains far worse than those produced with the coverage and quality training used for the CQFG models. Samples in the maximum-likelihood setting are provided in Figure 6. With three or more scales, models exhibit symptoms of overfitting: train BPD keeps decreasing while test BPD starts increasing, and IS and FID also degrade.

In Figure 6 we show samples obtained using VAE models trained with MLE. The models include one without invertible decoder layers, and with NVP layers using one, two and three scales. The samples illustrate the dramatic impact of using invertible NVP layers in these autoencoders.

Scales	Blocks	BPD ↓	IS ↑	FID ↓
1	2	3.77	**7.9**	**20.1**
2	2	3.48	6.9	27.7
2	4	**3.46**	6.9	28.9
3	3	3.49	6.5	31.7

(a) CQFG models

Scales	Blocks	BPD ↓	IS ↑	FID ↓
1	2	3.52	3.0	112.0
2	2	**3.41**	**4.5**	85.5
3	2	3.45	4.4	**78.7**
4	1	3.49	4.1	82.4

(b) piVAE models

Table 8: Evaluation on CIFAR-10 of different architectures of the invertible layers of the model.

Figure 6: Samples from MLE models (Table 8b) showing qualitative influence of multi-scale feature space.

D. Visualisations of reconstructions

We display reconstructions obtained by encoding and then decoding ground truth images with our models (CQG and CQFG from Table 1) in Figure 7. As is typical for expressive variational autoencoders, real images and their reconstructions cannot be distinguished visually.

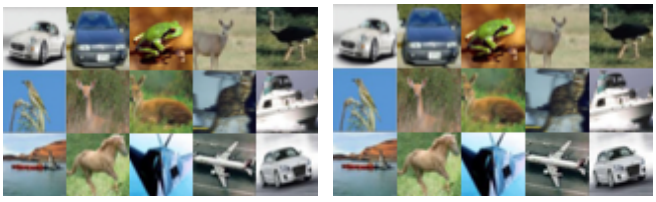

Figure 7: Real images and their reconstructions with the CQFG models.

E. Coverage and Quality driven training algorithm

Algorithm 1 Coverage and Quality driven training for our piVAE model.

for number of training steps **do**

- Sample m real images $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from p^*, approximated by the dataset.
- Map the real images to feature space $\{\boldsymbol{f(x)}^{(1)}, \ldots, \boldsymbol{f(x)}^{(m)}\}$ using the invertible transformation f.
- Encode the feature space vectors using the VAE encoder and get parameters for the posterior $q_\phi(\boldsymbol{z}|\boldsymbol{f(x)})$.
- Sample m latent variable vectors, $\{\hat{\boldsymbol{z}}^{(1)}, \ldots, \hat{\boldsymbol{z}}^{(m)}\}$ from the posterior $q_\phi(z|x)$, and m latent variable vectors $\{\tilde{\boldsymbol{z}}^{(1)}, \ldots, \tilde{\boldsymbol{z}}^{(m)}\}$ from the VAE prior $p_\theta(z)$
- Decode both sets of latent variable vectors using the VAE decoder into the means of conditional Gaussian distributions, $\{\mu(\hat{\boldsymbol{z}})^{(1)}, \ldots, \mu(\hat{\boldsymbol{z}})^{(m)}\}$ and $\{\mu(\tilde{\boldsymbol{z}})^{(1)}, \ldots, \mu(\tilde{\boldsymbol{z}})^{(m)}\}$
- Sample from the Gaussian densities obtained, $\{\mathcal{N}(.|\mu(\hat{\boldsymbol{z}})^{(i)}, \sigma I_n)\}_{i\leq m}$ and $\{\mathcal{N}(.|\mu(\tilde{\boldsymbol{z}})^{(i)}, \sigma I_n)\}_{i\leq m}$, which yields reconstructions in feature space $\{\widehat{\boldsymbol{f(x)}}^{(i)}\}_{i\leq m}$ and samples in feature space $\{\widetilde{\boldsymbol{f(x)}}^{(i)}\}_{i\leq m}$
- Map the samples and reconstructions back to image space using the inverse of the invertible transformation f^{-1} which yields reconstructions $\{\hat{\boldsymbol{x}}^{(i)}\}_{i\leq m}$ and samples $\{\tilde{\boldsymbol{x}}^{(i)}\}_{i\leq m}$
- Compute $L_C(p_\theta)$ using ground truth images $\{\boldsymbol{x}^{(i)}\}_{i\leq m}$ and their reconstructions $\{\hat{\boldsymbol{x}}^{(i)}\}_{i\leq m}$
- Compute $L_Q(p_\theta)$ by feeding the ground truth images $\{\boldsymbol{x}^{(i)}\}_{i\leq m}$ together with the sampled images $\{\tilde{\boldsymbol{x}}^{(i)}\}_{i\leq m}$ to the discriminator
- Optimize the discriminator by gradient descent to bring L_Q closer to L_Q^*
- Optimize the generator by gradient descent to minimize $L_Q + L_C$

end for