Variational Approaches for Auto-Encoding Generative Adversarial Networks

Mihaela Rosca* **Balaji Lakshminarayanan*** **David Warde-Farley** **Shakir Mohamed**
DeepMind
{mihaelacr,balajiln,dwf,shakir}@google.com

Abstract

Auto-encoding generative adversarial networks (GANs) combine the standard GAN algorithm, which discriminates between real and model-generated data, with a reconstruction loss given by an auto-encoder. Such models aim to prevent mode collapse in the learned generative model by ensuring that it is grounded in all the available training data. In this paper, we develop a principle upon which auto-encoders can be combined with generative adversarial networks by exploiting the hierarchical structure of the generative model. The underlying principle shows that variational inference can be used a basic tool for learning, but with the intractable likelihood replaced by a synthetic likelihood, and the unknown posterior distribution replaced by an implicit distribution; both synthetic likelihoods and implicit posterior distributions can be learned using discriminators. This allows us to develop a natural fusion of variational auto-encoders and generative adversarial networks, combining the best of both these methods. We describe a unified objective for optimization, discuss the constraints needed to guide learning, connect to the wide range of existing work, and use a battery of tests to systematically and quantitatively assess the performance of our method.

1 Introduction

Generative adversarial networks (GANs) [11] are one of the dominant approaches for learning generative models in contemporary machine learning research, which provide a flexible algorithm for learning in latent variable models. Directed latent variable models describe a data generating process in which a source of noise is transformed into a plausible data sample using a non-linear function, and GANs drive learning by discriminating observed data from model-generated data. GANs allow for training on large datasets, are fast to simulate from, and when trained on image data, produce visually compelling sample images. But this flexibility comes with instabilities in optimization that leads to the problem of mode-collapse, in which generated data does not reflect the diversity of the underlying data distribution. A large class of GAN variants that aim to address this problem are auto-encoder-based GANs (AE-GANs), that use an auto-encoder to encourage the model to better represent *all* the data it is trained with, thus discouraging mode-collapse.

Auto-encoders have been successfully used to improve GAN training. For example, plug and play generative networks (PPGNs) [28] produce state-of-the-art samples by optimizing an objective that combines an auto-encoder loss, a GAN loss, and a classification loss defined using a pre-trained classifier. AE-GANs can be broadly classified into three approaches: (1) those using an auto-encoder as the discriminator, such as energy-based GANs and boundary-equilibrium GANs [3], (2) those using a denoising auto-encoder to derive an auxiliary loss for the generator, such as denoising feature matching GANs [41], and (3) those combining ideas from VAEs and GANs. For example, the variational auto-encoder GAN (VAE-GAN) [22] adds an adversarial loss to the variational evidence lower bound objective. More recent GAN variants, such as mode-regularized GANs (MRGAN) [4] and adversarial generator encoders (AGE) [39] also use a separate encoder in order to stabilize GAN training. Such variants are interesting because they reveal interesting connections to VAEs, however the principles underlying the fusion of auto-encoders and GANs remain unclear.

*Equal contribution.

In this paper, we develop a principled approach for hybrid AE-GANs. By exploiting the hierarchical structure of the latent variable model learned by GANs, we show how another popular approach for learning latent variable models, variational auto-encoders (VAEs), can be combined with GANs. This approach will be advantageous since it allows us to overcome the limitations of each of these methods. Whereas VAEs often produce blurry images when trained on images, they do not suffer from the problem of mode collapse experienced by GANs. GANs allow few distributional assumptions to be made about the model, whereas VAEs allow for inference of the latent variables which is useful for representation learning, visualization and explanation. The approach we will develop will combine the best of these two worlds, provide a unified objective for learning, is purely unsupervised, requires no pre-training or external classifiers, and can easily be extended to other generative modeling tasks.

We begin by exposing the tools that we acquire for dealing with intractable generative models from both GANs and VAEs in section 2, and then make the following contributions:

- We show that variational inference applies equally well to GANs and how discriminators can be used for variational inference with implicit posterior approximations.
- Likelihood-based and likelihood-free models can be combined when learning generative models. In the likelihood-free setting, we develop variational inference with synthetic likelihoods that allows us to learn such models.
- We develop a principled objective function for auto-encoding GANs (α-GAN),[2] and describe considerations needed to make it work in practice.
- Evaluation is one of the major challenges in GAN research and we use a battery of evaluation measures to carefully assess the performance of our approach, comparing to DC-GAN, Wasserstein GAN and adversarial-generator-encoders (AGE), to show that we match the performance of these measures, and to emphasize the continuing challenge of evaluation in implicit generative models.

2 Overcoming Intractability in Generative Models

Latent Variable Models: Latent variable models describe a stochastic process by which modeled data is assumed to be generated (and thereby a process by which synthetic data can be simulated from the model distribution). In their simplest form, an unobserved quantity $\mathbf{z} \sim p(\mathbf{z})$ gives rise to a conditional distribution in the ambient space of the observed data, $\mathbf{x} \sim p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$. In several recently proposed model families, $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$ is specified via a generator (or decoder), $\mathcal{G}_{\boldsymbol{\theta}}(\mathbf{z})$, a non-linear function from $\mathbb{R}^K \to \mathbb{R}^D$ with parameters $\boldsymbol{\theta}$. In this work we consider models with $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, unless otherwise specified.

In *implicit latent variable models*, or likelihood-free models, we do not make any further assumptions about the data generating process and set the observation likelihood $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) = \delta(\mathbf{x} - \mathcal{G}_{\boldsymbol{\theta}}(\mathbf{z}))$, which is the model class considered in many simulation-based models, and especially in generative adversarial networks (GANs) [11]. In *prescribed latent variable models* we make a further assumption of observation noise, and any likelihood function that is appropriate to the data can be used.

In both implicit and prescribed models (such as GANs and VAEs, respectively) an important quantity that describes the quality of the model is the marginal likelihood $p_{\boldsymbol{\theta}}(\mathbf{x})$, in which the latent variables $\mathbf{z}$ have been integrated over. We learn about the parameters $\boldsymbol{\theta}$ of the model by minimizing an f-divergence between the model likelihood and the true data distribution $p^*(\mathbf{x})$, such as the KL-divergence $\mathrm{KL}[p^*(\mathbf{x})||p_{\boldsymbol{\theta}}(\mathbf{x})]$. But in both types of models, the marginal likelihood is intractable, requiring us to find solutions by which we can overcome this intractability in order to learn the model parameters.

Generative Adversarial Networks: One way to overcome the intractability of the marginal likelihood is to never compute it, and instead to learn about the model parameters using a tool that gives us indirect information about it. Generative adversarial networks (GANs) [11] do this by learning a suitably powerful discriminator that learns to distinguish samples from the true distribution $p^*(\mathbf{x})$ and the model $p_{\boldsymbol{\theta}}(\mathbf{x})$. The ability of the discriminator (or lack thereof) to distinguish between real and generated data is the learning signal that drives the optimization of the model parameters: when this discriminator is unable to distinguish between real and simulated data, we have learned all we can about the observed data. This is a principle of learning known under various names, including adversarial training [11], estimation-by-comparison [13, 14], and unsupervised-as-supervised learning [15].

[2] We use the Greek α prefix for α-GAN, as AEGAN and most other Latin prefixes seem to have been taken https://deephunt.in/the-gan-zoo-79597dc8c347.

Let $y = 1$ denote a binary label corresponding to data samples from the real data distribution $\mathbf{x} \sim p^*$ and $y = 0$ for simulated data $\mathbf{x} \sim p_{\boldsymbol{\theta}}$, and a discriminator $\mathcal{D}_{\boldsymbol{\phi}}(\mathbf{x}) = p(y = 1|\mathbf{x})$ that gives the probability that an input $\mathbf{x}$ is from the real distribution, with discriminator parameters $\boldsymbol{\phi}$. At any time point, we update the discriminator by drawing samples from the real data and from the model and minimize the binary cross entropy (1). The generator parameters $\boldsymbol{\theta}$ are then updated by maximizing the probability that samples from $p_{\boldsymbol{\theta}}(\mathbf{x})$ are classified as real. Following Goodfellow et al. [11], an alternative loss in (2) is used since it provides stronger gradients. The optimization is then an alternating minimization w.r.t. $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$.

$$\textbf{Discriminator loss: } \mathbb{E}_{p^*(\mathbf{x})}\big[-\log \mathcal{D}_{\boldsymbol{\phi}}(\mathbf{x})\big] + \mathbb{E}_{p_{\boldsymbol{\theta}}(\mathbf{x})}\big[-\log(1 - \mathcal{D}_{\boldsymbol{\phi}}(\mathbf{x}))\big]. \tag{1}$$

$$\textbf{Generator loss: } \mathbb{E}_{p_{\boldsymbol{\theta}}(\mathbf{x})}[\log(1 - \mathcal{D}_{\boldsymbol{\phi}}(\mathbf{x}))]; \quad \textbf{Alternative loss: } \mathbb{E}_{p_{\boldsymbol{\theta}}(\mathbf{x})}[-\log \mathcal{D}_{\boldsymbol{\phi}}(\mathbf{x})] \tag{2}$$

GANs are especially interesting as a way of learning in latent variable models, since they do not require inference of the latent variables $\mathbf{z}$, and are applicable to both implicit and prescribed models. GANs are based on an underlying principle of density ratio estimation [27, 38] and thus provide us with an important tool for overcoming intractable distributions.

The Density Ratio Trick: By introducing the labels $y = 1$ for real data and $y = 0$ for simulated data in GANs, we re-express the data and model distributions in conditional form, i.e. $p^*(\mathbf{x}) = p(\mathbf{x}|y = 1)$ for the true distribution, and $p_{\boldsymbol{\theta}}(\mathbf{x}) = p(\mathbf{x}|y = 0)$ for the model. The *density ratio* $r_{\boldsymbol{\phi}}(\mathbf{x})$ between the true distribution and model distribution can be computed using these conditional distributions as:

$$r_{\boldsymbol{\phi}}(\mathbf{x}) = \frac{p^*(\mathbf{x})}{p_{\boldsymbol{\theta}}(\mathbf{x})} = \frac{p(\mathbf{x}|y = 1)}{p(\mathbf{x}|y = 0)} = \frac{p(y = 1|\mathbf{x})}{p(y = 0|\mathbf{x})} = \frac{\mathcal{D}_{\boldsymbol{\phi}}(\mathbf{x})}{1 - \mathcal{D}_{\boldsymbol{\phi}}(\mathbf{x})} \tag{3}$$

where we used Bayes' rule in the second last step and assumed that the marginal class probabilities are equal, i.e. $p(y = 0) = p(y = 1)$. This tells us that whenever we wish to compute a density ratio, we can simply draw samples from the two distributions and implement a binary classifier $\mathcal{D}_{\boldsymbol{\phi}}(\mathbf{x})$ of the two sets of samples. By using the density ratio, GANs account for the intractability of the marginal likelihood by looking only at its relative behavior with respect to the true distribution. This trick only requires samples from the two distributions and never access to their analytical forms, making it particularly well-suited for dealing with implicit distributions or likelihood-free models. Since we are required to build a classifier, we can use all the knowledge we have about building state-of-the-art classifiers. This trick is widespread [16, 18, 24, 25, 37], although perhaps not stated as explicitly. While using class probability estimation is amongst the most popular, the density ratio can also be computed in several other ways including by f-divergence minimization and density-ratio matching [27, 34].

Variational Inference: A second approach for dealing with intractable likelihoods is to approximate them. There are several ways to approximate the marginal likelihood, but one of the most popular is to derive a lower bound to it by transforming the marginal likelihood into an expectation over a new variational distribution $q_\eta(\mathbf{z}|\mathbf{x})$, whose variational parameters $\boldsymbol{\eta}$ can be optimized to ensure that a tight bound can be found. The bound obtained is the popular variational lower bound $\mathcal{F}(\boldsymbol{\theta}, \boldsymbol{\eta})$:

$$\log p_{\boldsymbol{\theta}}(\mathbf{x}) = \log \int p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z} \geq \mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})] - \mathrm{KL}[q_\eta(\mathbf{z}|\mathbf{x})||p(\mathbf{z})] = \mathcal{F}(\boldsymbol{\theta}, \boldsymbol{\eta}). \tag{4}$$

Variational auto-encoders (VAEs) [20, 31] provide one way of implementing variational inference in which the variational distribution q is represented as an encoder, and the variational and model parameters are jointly optimized using the pathwise stochastic gradient estimator (also known as the reparameterization trick) [10, 20, 31]. The variational lower bound (4) is a description applicable to both implicit and prescribed models, and gives us a further tool for dealing with intractable distributions, which is to introduce an encoder to invert the generative process and optimize a lower bound on the marginal likelihood.

Synthetic Likelihoods: When the likelihood function is unknown, the variational lower bound (4) cannot directly be used for learning. One further tool with which to overcome this, is to replace the likelihood with a substitute, or *synthetic likelihood* $R(\boldsymbol{\theta})$. The original formulation of the synthetic likelihood [42] is based on a Gaussian assumption, but we use the term here to mean any general substitute for the likelihood that maintains its asymptotic properties. The synthetic likelihood form we use here was proposed by Dutta et al. [9] for approximate Bayesian computation (ABC). The idea is to introduce a synthetic likelihood into the likelihood term of (4) by dividing and multiplying by the true data distribution $p^*(\mathbf{x})$:

$$\mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}\left[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})\right] = \mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}\left[\log \tfrac{p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})}{p^*(\mathbf{x})}\right] + \mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}\left[\log p^*(\mathbf{x})\right] \tag{5}$$

The first term in (5) contains the synthetic likelihood $R(\boldsymbol{\theta}) = \frac{p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})}{p^*(\mathbf{x})}$. Any estimate of the ratio $R(\boldsymbol{\theta})$ is an estimate of the likelihood since they are proportional (and the normalizing constant is independent of $\boldsymbol{\theta}$). Wherever an intractable likelihood appears, we can instead use this ratio. The synthetic likelihood can be estimated using the density ratio trick by training a discriminator to distinguish between samples from the marginal $p^*(\mathbf{x})$ and the conditional $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$ where $\mathbf{z}$ is drawn from $q_\eta(\mathbf{z}|\mathbf{x})$. The second term in (5) is independent of $\boldsymbol{\theta}$ and can be ignored for optimization purposes.

3 A Fusion of Variational and Adversarial Learning

GANs and VAEs have given us useful tools for learning and inference in generative models and we now use these tools to build new hybrid inference methods. The VAE forms our generic starting point, and we will gradually transform it to be more GAN-like.

Implicit Variational Distributions: The major task in variational inference is the choice of the variational distribution $q_\eta(\mathbf{z}|\mathbf{x})$. Common approaches, such as mean-field variational inference, assume simple distributions like a Gaussian, but we would like not to make a restrictive choice of distribution. If we treat this distribution as implicit—we do not know its distribution but are able to generate from it—then we can use the density ratio trick to replace the KL-divergence term in (4).

$$-\mathrm{KL}[q_\eta(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})] = \mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}\left[\log \frac{p(\mathbf{z})}{q_\eta(\mathbf{z}|\mathbf{x})}\right] \approx \mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}\left[\log \frac{\mathcal{C}_{\boldsymbol{\omega}}(\mathbf{z})}{1-\mathcal{C}_{\boldsymbol{\omega}}(\mathbf{z})}\right]. \tag{6}$$

We will thus introduce a latent classifier $\mathcal{C}_{\boldsymbol{\omega}}(\mathbf{z})$ that discriminates between latent variables $\mathbf{z}$ produced by an encoder network and variables sampled from a standard Gaussian distribution. For optimization, the expectation in (6) is evaluated by Monte Carlo integration. Replacing the KL-divergence with a discriminator was first proposed by Makhzani et al. [24], and a similar idea was used by Mescheder et al. [25] for adversarial variational Bayes.

Likelihood Choice: If we make the explicit choice of a likelihood $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$ in the model, the we can substitute our chosen likelihood into (4). We choose a zero-mean Laplace distribution $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z}) \propto \exp(-\lambda||\mathbf{x} - \mathcal{G}_{\boldsymbol{\theta}}(\mathbf{z})||_1)$ with scale parameter λ, which corresponds to using a variational auto-encoder with an L_1 reconstruction loss; this is a highly popular choice and used in many related auto-encoder GAN variants, such as AGE, BEGAN, cycle GAN and PPGN [3, 28, 39, 44].

In GANs the effective likelihood is unknown and intractable. We can again use our tools for intractable inference by replacing the intractable likelihood by its synthetic substitute. Using the synthetic likelihood (5) introduces a new synthetic-likelihood classifier $\mathcal{D}_{\boldsymbol{\phi}}(\mathbf{x})$ that discriminates between data sampled from the conditional and marginal distributions of the model. The reconstruction term $\mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})]$ in (4) can be either:

$$\mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}\left[-\lambda||\mathbf{x} - \mathcal{G}_{\boldsymbol{\theta}}(\mathbf{z})||_1\right] \qquad \text{or} \qquad \mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}\left[\log \frac{\mathcal{D}_{\boldsymbol{\phi}}(\mathcal{G}_{\boldsymbol{\theta}}(\mathbf{z}))}{1-\mathcal{D}_{\boldsymbol{\phi}}(\mathcal{G}_{\boldsymbol{\theta}}(\mathbf{z}))}\right]. \tag{7}$$

These two choices have different behaviors. Using the synthetic discriminator-based likelihood means that this model will have the ability to use the adversarial game to learn the data distribution, although it may still be subject to mode-collapse. This is where an explicit choice of likelihood can be used to ensure that we assign mass to all parts of the output support and prevent collapse. When forming a final loss we can make use of a weighted sum of the two to get the benefits of both types of behavior.

Hybrid Loss Functions: An hybrid objective function that combines all these choices is:

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\eta}) = \mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}\left[-\lambda||\mathbf{x} - \mathcal{G}_{\boldsymbol{\theta}}(\mathbf{z})||_1 + \log \frac{\mathcal{D}_{\boldsymbol{\phi}}(\mathcal{G}_{\boldsymbol{\theta}}(\mathbf{z}))}{1-\mathcal{D}_{\boldsymbol{\phi}}(\mathcal{G}_{\boldsymbol{\theta}}(\mathbf{z}))} + \log \frac{\mathcal{C}_{\boldsymbol{\omega}}(\mathbf{z})}{1-\mathcal{C}_{\boldsymbol{\omega}}(\mathbf{z})}\right] \tag{8}$$

We are required to build four networks: the classifier $\mathcal{D}_{\boldsymbol{\phi}}(\mathbf{x})$ is trained to discriminate between reconstructions from an auto-encoder and real data points; a second classifier is trained to discriminate between latent samples produced by the encoder and samples from a standard Gaussian; we must implement the deep generative model $\mathcal{G}_{\boldsymbol{\theta}}(\mathbf{z})$, and also the encoder network $q_\eta(\mathbf{z}|\mathbf{x})$, which can be implemented using any type of deep network. The density-ratio estimators $\mathcal{D}_{\boldsymbol{\phi}}$ and $\mathcal{C}_{\boldsymbol{\omega}}$ can be trained using any loss for density ratio estimation described in section 2, hence their loss functions are not shown in (8). We refer to training using (8) as α-GAN. Our algorithm alternates between updates

of the parameters of the generator $\boldsymbol{\theta}$, encoder $\boldsymbol{\eta}$, synthetic likelihood discriminator $\boldsymbol{\phi}$, and the latent code discriminator $\boldsymbol{\omega}$; see algorithm 1.

Improved Techniques: Equation (8) provides a principled starting point for optimization based on losses obtained by the combination of insights from VAEs and GANs. To improve the stability of optimization and speed of learning we make two modifications. Firstly, following the insights from GANs, we consider an alternative-loss formulation for both the latent discriminator and the synthetic likelihood discriminator, where we replace $-\log(1-\mathcal{D}_{\boldsymbol{\phi}})$ with $\log \mathcal{D}_{\boldsymbol{\phi}}$ while training the generator as it provides non-saturating gradients. The minimization of the generator parameters becomes:

$$\textbf{Generator Loss: } \mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}\Big[\lambda||\mathbf{x}-\mathcal{G}_{\boldsymbol{\theta}}(\mathbf{z})||_1 - \log \mathcal{D}_{\boldsymbol{\phi}}(\mathcal{G}_{\boldsymbol{\theta}}(\mathbf{z}))\Big], \tag{9}$$

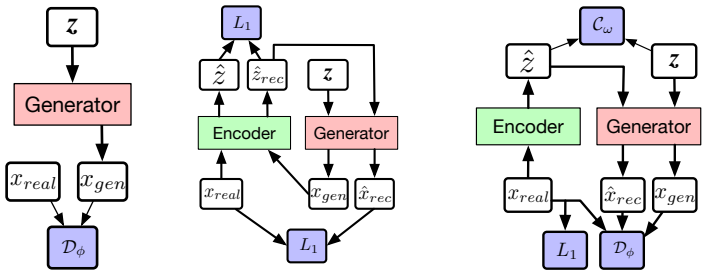
which shows that we have are using the standard GAN updates for the generator (2), with the addition of a reconstruction term, that discourages mode collapse as $\mathcal{G}_{\boldsymbol{\theta}}$ needs to be able to reconstruct every input $\mathbf{x}$.

Secondly, we found that passing the samples to the discriminator as fake samples, in addition to the reconstructions, helps improve performance. One way to justify the use of samples is to apply Jensen's inequality, that is, $\log p_{\boldsymbol{\theta}}(\mathbf{x}) = \log \int p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z} \geq \mathbb{E}_{p(\mathbf{z})}[\log p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})]$, and replace this with a synthetic likelihood, as done for reconstructions. Instead of training two separate discriminators, we train a single discriminator which treats samples and reconstructions as fake, and p^* as real.

4 Related work

Figure 1 summarizes our architecture and the architectures we compare with in the experimental section. Hybrids of VAEs and GANs can be classified by whether the density ratio trick is applied only to likelihood, prior approximation or both. Table 1 reveals the connections to related approaches (see also [16, Table 1]). DCGAN [30] and WGAN-GP [12] are pure GAN variants; they do not use an auto-encoder loss nor do they do inference. WGAN-GP shares the attributes of DCGAN, except that it uses a critic that approximates the Wasserstein distance [1] instead of a density ratio estimator. AGE uses an approximation of KL term, however it does not use a synthetic likelihood, but instead uses observed likelihoods - reconstruction losses - for both latent codes and data. The adversarial component of AGE arises form the opposing goals of the encoder and decoder: the encoder tries to compress data into codes drawn from the prior, while compressing samples into codes which do not match the prior; at the same time the decoder wants to generate samples that when encoded by the encoder will generate codes which match the prior distribution. VAE uses the observation likelihood and an analytic KL term, however it tends to produce blurry images, hence we do not consider it here. To solve the blurriness issue, VAE-GAN change the VAE loss function by replacing the observed likelihood on pixels with an adversarial loss together with a reconstruction metric in discriminator feature space. Unlike our work, VAE-GAN still uses the analytical KL loss to minimize the distance between the prior and the posterior of the latents, and they do not discuss the connection to density ratio estimation. Similar to VAE-GAN, Dosovitskiy and Brox [7] replace the observed likelihood term in the variational lower bound with a weighted sum of a feature matching loss (here the features matched are those of a pre-trained classifier) and an adversarial loss, but instead of using the analytical KL, they use a numerical approximation. We explore the same approximation (also used by AGE) in Section D in the Appendix and show empirically that using the code discriminator is less sensitive to hyperparameters and produces on average better results. By not using a pre-trained classifier or a feature matching loss, α-GAN is trained end-to-end, completely unsupervised and maximizes a lower bound on the true data likelihood.

ALI [8], BiGAN [6] perform inference by creating an adversarial game between the encoder and decoder via a discriminator that operates on $\mathbf{x}, \mathbf{z}$ space. The discriminator learns to distinguish between input-output pairs of the encoder (where $\mathbf{x}$ is a sample from the data distribution and $\mathbf{z}$ is a sample from the conditional posterior $q_\eta(\mathbf{z}|\mathbf{x})$) and decoder (where $\mathbf{z}$ is a sample from the latent prior and $\mathbf{x}$ is a sample from the conditional $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$). Unlike α-GAN , their approach operates jointly, without exploiting the structure of the model. Cycle-GAN [44] was proposed for image-to-image translation, but applying the underlying *cycle consistency* principle to image-to-code translation reveals an interesting connection with α-GAN. Recall that in $\mathbf{x}$ space, we both use a pointwise reconstruction term $\|\mathbf{x}-\hat{\mathbf{x}}\|_1$ term as well as a loss to match the distributions of $\mathbf{x}$ and $\hat{\mathbf{x}}$. In $\mathbf{z}$ space, we only match the distributions of $\mathbf{z}$ and $\hat{\mathbf{z}}$ in α-GAN. Adding pointwise code reconstruction loss $||\mathbf{z}-\hat{\mathbf{z}}||$ would make it similar to CycleGAN. We note however that the CycleGAN authors used the least square GAN loss, while the traditional GAN loss needs to be used to obtain the variational lower bound in (4).



(a) DCGAN (b) AGE (c) α-GAN

Figure 1: Architectures for the three models used for comparison. (WGAN is similar to DCGAN.)

In mode regularized GANs (MRGANs) [4] the generator is part of an auto-encoder, hence it learns how to produce reconstructions from the posterior over latent codes and also independently learns how to produce samples from codes drawn from the prior over latents. MRGANs employ two discriminators, one to distinguish between data and reconstructions and one to distinguish between data and samples. As described in Section 3, in α-GAN we also pass both samples and reconstructions through the discriminator (which learns to distinguish between them and data). However, we only need one discriminator, as we explicitly match the latent prior and the latent posterior given by the model using KL term in (4), which encourages the distributions of reconstructions and sample to be similar.

Algorithm	Likelihood		Prior		
	Observer	Ratio estimator ("synthetic")	KL (analytic)	KL (approximate)	Ratio estimator
VAE	✓		✓		
DCGAN		✓			
VAE-GAN	✓	*	✓		
AGE	✓			✓	
α-GAN (ours)	✓	✓			✓

Table 1: Comparison of different approaches for training generative latent variable models.

5 Evaluation metrics

Evaluating generative models is challenging [36]. In particular, evaluating GANs is difficult due to the lack of likelihood. Multiple proxy metrics have been proposed, and we explore some of them in this work and assess their strengths and weaknesses in the experiments section.

Inception score: The inception score was proposed by Salimans et al. [32] and has been widely adopted since. The inception score uses a pre-trained neural network classifier to capture to two desirable properties of generated samples: highly classifiable and diverse with respect to class labels. It does so by computing the average of the KL divergences between the conditional label distributions of samples (expected to have low entropy for easily classifiable samples) and the marginal distribution obtained from all the samples (expected to have high entropy if all classes are equally represented in the set of samples). As the name suggests, the classifier network used to compute the inception score was originally an Inception network [35] trained on the ImageNet dataset. For comparison to previous work, we report scores using this network. However, when reporting CIFAR-10 results we also report metrics obtained using a VGG style convolutional neural network, trained on the same dataset, which obtained 5.5% error (see section H.5 in the details on this network).

Multi-scale structural similarity (MS-SSIM): The inception score fails to capture mode collapse inside a class: the inception score of a model that generates the same image for a class and the inception score of a model that is able to capture diversity inside a class are the same. To address this issue, Odena et al. [29] assess the similarity between class-conditional generated samples using MS-SSIM [40], an image similarity metric that has been shown to correlate well with human judgement. MS-SSIM ranges between 0.0 (low similarity) and 1.0 (high similarity). By computing the average pairwise MS-SSIM score between images in a given set, we can determine how similar the images are, and in particular, we can compare with the similarity obtained on a reference set (the training set, for example). Since our models are not class conditional, we only used MS-SSIM to evaluate models on CelebA [23], a dataset of faces, since the variability of the data there is smaller. For datasets with very distinct labels, using MS-SSIM would not give us a good metric, since there will be high

variability between classes. We report *sample diversity score* as 1-MSSSIM, hence higher values of sample diversity score are better.

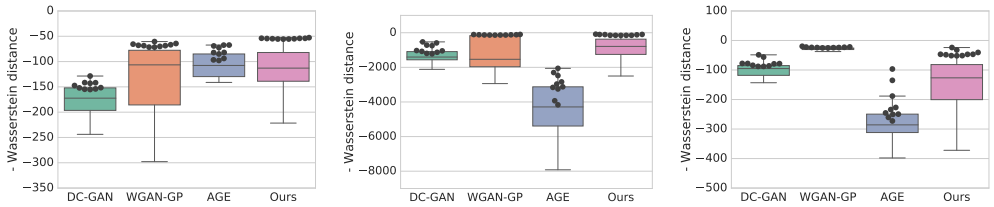
Independent Wasserstein critic: Danihelka et al. [5] proposed training an independent Wasserstein GAN critic to distinguish between held out validation data and generated samples.[3] This metric measures both overfitting and mode collapse: if the generator memorizes the training set, the critic trained on validation data will be able to distinguish between samples and data; if mode collapse occurs, the critic will have an easy task distinguishing between data and samples. The Wasserstein distance does not saturate when the two distributions do not overlap [1], and the magnitude of the distance represents how easy it is for the critic to distinguish between data and samples. To be consistent with the other metrics, we report the negative of the Wasserstein distance between the test set and generator, hence higher values are better. Since the critic is trained independently for evaluation only, and thus does not affect the training of the generator, this evaluation technique can be used irrespective of the training criteria used [5]. To ensure that the independent critic does not overfit to the validation data, we only start training it half way through the training of our model and examined the learning curves during training (see Appendix E in the supplementary material for learning curves).

6 Experiments

To better understand the importance of autoencoder based methods in the GAN landscape, we implemented and compared the proposed α-GAN with another hybrid model, AGE, as well as pure GAN variants such as DCGAN and WGAN-GP, across three datasets: ColorMNIST [26], CelebA [23] and CIFAR-10 [21]. We complement the visual inspection of samples with a battery of numerical test using the metrics above to get an insight of both on the models and on the metrics themselves. For a comprehensive analysis, we report both the best values obtained by each algorithm, as well as the quartiles obtained by each hyperparameter sweep for each model, to assess the sensitivity to hyperparameters. On all metrics, we report box plot for all the hyperparameters we considered with the best 10 jobs indicated by black circles, where higher is better. To the best of our knowledge, we are the first to do such an analysis of the GAN landscape.

For details of the training procedure used in all our experiments, including the hyperparameter sweeps, we refer to Appendix H in the supplementary material. Note that the models considered here are all unconditional and do not make use of label information, hence it is not appropriate to compare our results with those obtained using conditional GANs [29] and semi-supervised GANs [32].

Results on ColorMNIST : We compare the values of an independent Wasserstein critic in Figure 2(a), where higher values are better. While our approach has a broad spread of sensitivity to hyperparameters, on this metric, we are able to achieve many settings in which our performance is the best among the compared methods. This is supported by the generated samples shown in Figure 3.



(a) ColorMNIST (b) CelebA (c) CIFAR-10

Figure 2: Negative Wasserstein distance estimated using an independent Wasserstein critic. The metric captures overfitting to the training data and low quality samples.

Results on CelebA: The CelebA dataset consists of 64×64 pixel images of faces of celebrities. We show samples from the four models in Figure 4. We also compare the models using the independent Wasserstein critic in Figure 2(b) and sample diversity score in Figure 5(a). α-GAN is competitive with WGAN-GP and AGE. On MS-SSIM, where lower values are better, we have a large spread of hyperparameter sensitivity, although α-GAN also achieves the best result on this metric. Unlike WGAN and DCGAN, an advantage of α-GAN and AGE is the ability to reconstruct inputs. Appendix C shows that α-GAN produces better reconstructions than AGE.

[3] Danihelka et al. [5] used the original WGAN [1], whereas we use improved WGAN-GP proposed in [12].

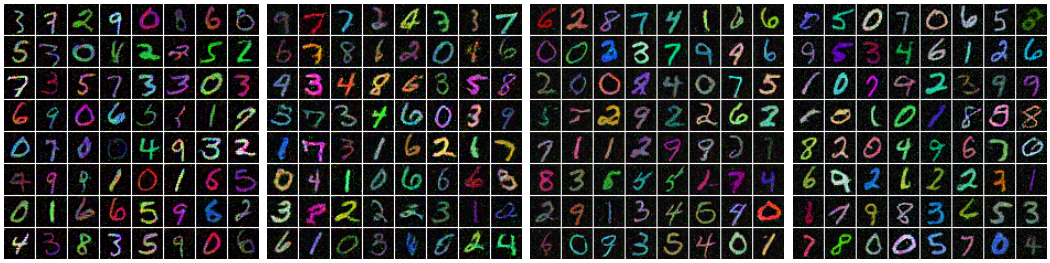


Figure 3: Best samples on ColorMNIST (L-to-R): samples from DCGAN, WGAN-GP, AGE and the proposed variant α-GAN, according to visual inspection.

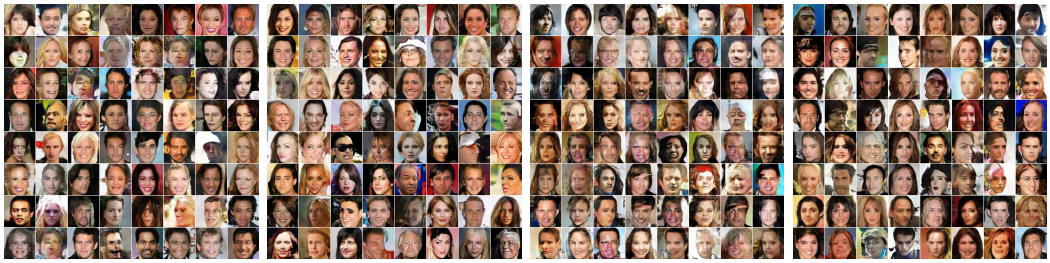


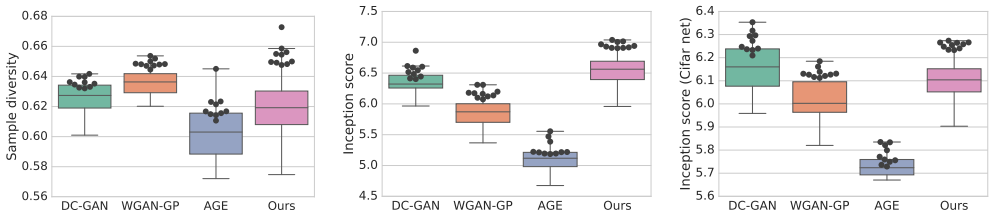
Figure 4: Best samples on CelebA (L-to-R): samples from DCGAN, WGAN-GP, AGE and α-GAN, according to visual inspection. See Figure 7 in Appendix A for a higher resolution version.

Results on CIFAR-10: We show samples from the various models in Figure 6. We evaluate α-GAN using the independent critic, shown in Figure 2(c), where WGAN-GP is the best performing model. We also compare the ImageNet-based inception score in Figures 5(b), where it has the best performance, and with the CIFAR-10 based inception score in Figure 5(c) where it is competitive with DC-GAN, but no single hyperparameter setting is the clear winner. The best reported ImageNet-based inception score on CIFAR for unsupervised models is 7.72 ± 0.13 by DFM-GAN [41], who also report 5.34 ± 0.05 for ALI [8], however these are trained on different architectures and may not be directly comparable. To understand the importance of the model used to evaluate the Inception score, we looked at the relationship between the Inception score measured with the Inception net trained on ImageNet (introduced by [32]) and the VGG style net trained on CIFAR-10, the same dataset on which we train the generative models. We observed that 15% of the jobs in a hyperparameter sweep were ranked as being in the top 50% by the ImageNet Inception score while ranked in the bottom 50% by the CIFAR-10 Inception score. Hence, using the Inception score of a model trained on a different dataset than the generative model is evaluated on can be misleading when ranking models.

Experimental insights: Irrespective of the algorithm used, we found that two factors can contribute significantly to the quality of the results:

- *The network architectures*. We noticed that the most decisive factor in the lies in the architectures chosen for the discriminator and generator. We found that given enough capacity, DCGAN (which uses the traditional GAN [11]) can be very robust, and does not suffer from obvious mode collapse on the datasets we tried. All models reported are sensitive to changes in the architectures, with minor changes resulting in catastrophic mode collapse, regardless of other hyperparameters.
- *The number of updates performed by the individual components of the model*. For DCGAN, we update the generator twice for each discriminator update following https://github.com/carpedm20/DCGAN-tensorflow; we found it stabilizes training and produces significantly better samples, contrary to GAN theory which suggests training discriminator multiple times instead. Our findings are also consistent with the updates performed by the AGE model, where the generator is updated multiple times for each encoder update. Similarly, for α-GAN, we update the encoder (which can be seen as the latent code generator) and the generator twice for each discriminator and code discriminator update. On the other hand, for WGAN-GP, we update the discriminator 5 times for each generator update following [1, 12].

While the independent Wasserstein critic does not directly measure sample diversity, we notice a high correlation between its estimate of the negative Wasserstein distance and sample similarity (see Appendix G). Note however that the measures are not perfectly correlated, and if used to rank the best performing jobs in a hyperparameter sweep they give different results.



(a) CelebA (b) Inception score (ImageNet) (c) Inception score (CIFAR)

Figure 5: Left plot shows sample diversity results on CelebA. Middle plot: Inception score results on CIFAR-10. Right most plot shows Inception score computed using a VGG style network trained on CIFAR-10. As a reference benchmark, we also compute these scores using samples from test data split; diversity: 0.621, inception score: 11.25, inception score (VGG net trained on CIFAR-10): 9.18.

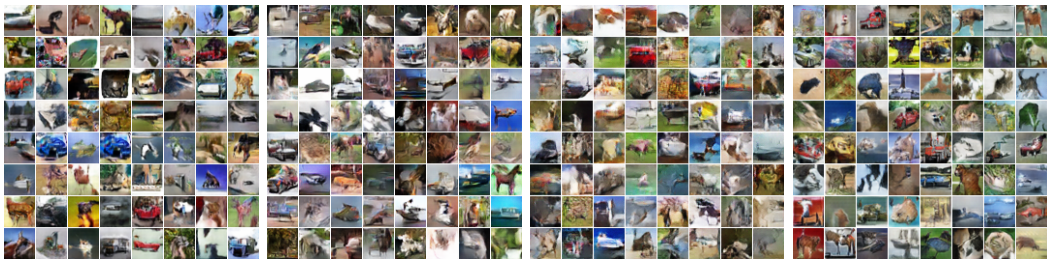


Figure 6: Best samples on CIFAR-10 (L-to-R): DC-GAN, WGAN-GP, AGE and α-GAN, according to visual inspection. For AGE and α-GAN reconstructions on CIFAR-10, see Appendix C.

7 Discussion

In this paper we have combined the variational lower bound on the data likelihood with the density ratio trick, allowing us to better understand the connection between variational auto-encoders and generative adversarial networks. From the newly introduced lower bound on the likelihood we derived a new training criteria for generative models, named α-GAN. α-GAN combines an adversarial loss with a data reconstruction loss. This can be seen in two ways: from the VAE perspective, it can solve the blurriness of samples via the (learned) adversarial loss; from the GAN perspective, it can solve mode collapse by grounding the generator using a perceptual similarity metric on the data - the reconstruction loss. In a quest to understand how α-GAN compares to other GAN models (including auto-encoder based ones), we deployed a set of metrics on 3 datasets as well as compared samples visually. While the picture of evaluating GANs is far from being completed, we show that the metrics employed are complementary and assess different failure modes of GANs (mode collapse, overfitting to the training data and poor learning of the data distribution).

The prospect of marrying the two approaches (VAEs and GANs) comes with multiple benefits: auto-encoder based methods can be used to reconstruct data and thus can be used for inpainting [28] [43]; having an inference network allows our model to be used for representation learning [2], where we can learn disentangled representations by choosing an appropriate latent prior. We thus believe VAE-GAN hybrids such as α-GAN can be used in unsupervised, supervised and reinforcement learning settings, which leads the way to directions of research for future work.

Acknowledgements. We thank Ivo Danihelka and Chris Burgess for helpful feedback and discussions.

References

[1] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein GAN. *arXiv preprint arXiv:1701.07875*, 2017.

[2] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[3] D. Berthelot, T. Schumm, and L. Metz. BEGAN: Boundary equilibrium generative adversarial networks. *arXiv preprint arXiv:1703.10717*, 2017.

[4] T. Che, Y. Li, A. P. Jacob, Y. Bengio, and W. Li. Mode regularized generative adversarial networks. *arXiv preprint arXiv:1612.02136*, 2016.

[5] I. Danihelka, B. Lakshminarayanan, B. Uria, D. Wierstra, and P. Dayan. Comparison of Maximum Likelihood and GAN-based training of Real NVPs. *arXiv preprint arXiv:1705.05263*, 2017.

[6] J. Donahue, P. Krähenbühl, and T. Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016.

[7] A. Dosovitskiy and T. Brox. Generating images with perceptual similarity metrics based on deep networks. In *Advances in Neural Information Processing Systems*, pages 658–666, 2016.

[8] V. Dumoulin, I. Belghazi, B. Poole, A. Lamb, M. Arjovsky, O. Mastropietro, and A. Courville. Adversarially learned inference. *arXiv preprint arXiv:1606.00704*, 2016.

[9] R. Dutta, J. Corander, S. Kaski, and M. U. Gutmann. Likelihood-free inference by penalised logistic regression. *arXiv preprint arXiv:1611.10242*, 2016.

[10] M. C. Fu. Gradient estimation. *Handbooks in operations research and management science*, 13: 575–616, 2006.

[11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

[12] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville. Improved Training of Wasserstein GANs. *arXiv preprint arXiv:1704.00028*, 2017.

[13] M. U. Gutmann and A. Hyvärinen. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, 13(Feb):307–361, 2012.

[14] M. U. Gutmann, R. Dutta, S. Kaski, and J. Corander. Statistical inference of intractable generative models via classification. *arXiv preprint arXiv:1407.4981*, 2014.

[15] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning*. Springer, pp 495–497, 10th printing, 2nd edition, 2013.

[16] F. Huszár. Variational inference using implicit distributions. *arXiv preprint arXiv:1702.08235*, 2017.

[17] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 448–456, 2015.

[18] T. Karaletsos. Adversarial message passing for graphical models. *arXiv preprint arXiv:1612.05048*, 2016.

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

[20] D. P. Kingma and M. Welling. Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[21] A. Krizhevsky. Learning multiple layers of features from tiny images. 2009.

[22] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther. Autoencoding beyond pixels using a learned similarity metric. In *Proceedings of The 33rd International Conference on*

Machine Learning, pages 1558–1566, 2016.

[23] Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.

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

[25] L. Mescheder, S. Nowozin, and A. Geiger. Adversarial Variational Bayes: Unifying Variational Autoencoders and Generative Adversarial Networks. *arXiv preprint arXiv:1701.04722*, 2017.

[26] L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein. Unrolled generative adversarial networks. *arXiv preprint arXiv:1611.02163*, 2016.

[27] S. Mohamed and B. Lakshminarayanan. Learning in implicit generative models. *arXiv preprint arXiv:1610.03483*, 2016.

[28] A. Nguyen, J. Yosinski, Y. Bengio, A. Dosovitskiy, and J. Clune. Plug & play generative networks: Conditional iterative generation of images in latent space. *arXiv preprint arXiv:1612.00005*, 2016.

[29] A. Odena, C. Olah, and J. Shlens. Conditional image synthesis with auxiliary classifier GANs. *arXiv preprint arXiv:1610.09585*, 2016.

[30] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[31] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *The 31st International Conference on Machine Learning (ICML)*, 2014.

[32] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training GANs. *arXiv preprint arXiv:1606.03498*, 2016.

[33] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[34] M. Sugiyama, T. Suzuki, and T. Kanamori. *Density ratio estimation in machine learning*. Cambridge University Press, 2012.

[35] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2818–2826, 2016.

[36] L. Theis, A. v. d. Oord, and M. Bethge. A note on the evaluation of generative models. *arXiv preprint arXiv:1511.01844*, 2015.

[37] D. Tran, R. Ranganath, and D. M. Blei. Deep and hierarchical implicit models. *arXiv preprint arXiv:1702.08896*, 2017.

[38] M. Uehara, I. Sato, M. Suzuki, K. Nakayama, and Y. Matsuo. Generative adversarial nets from a density ratio estimation perspective. *arXiv preprint arXiv:1610.02920*, 2016.

[39] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Adversarial generator-encoder networks. *arXiv preprint arXiv:1704.02304*, 2017.

[40] Z. Wang, E. P. Simoncelli, and A. C. Bovik. Multiscale structural similarity for image quality assessment. In *Signals, Systems and Computers, 2004. Conference Record of the Thirty-Seventh Asilomar Conference on*, volume 2, pages 1398–1402. IEEE, 2003.

[41] D. Warde-Farley and Y. Bengio. Improving generative adversarial networks with denoising feature matching. *ICLR submission*, 2017.

[42] S. N. Wood. Statistical inference for noisy nonlinear ecological dynamic systems. *Nature*, 466 (7310):1102–1104, 2010.

[43] J. Xie, L. Xu, and E. Chen. Image denoising and inpainting with deep neural networks. In *Advances in Neural Information Processing Systems*, pages 341–349, 2012.

[44] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *arXiv preprint arXiv:1703.10593*, 2017.

Supplementary material

A Model Samples

Figure 7 shows larger-sized versions of the samples in Figure 4 in the main text.

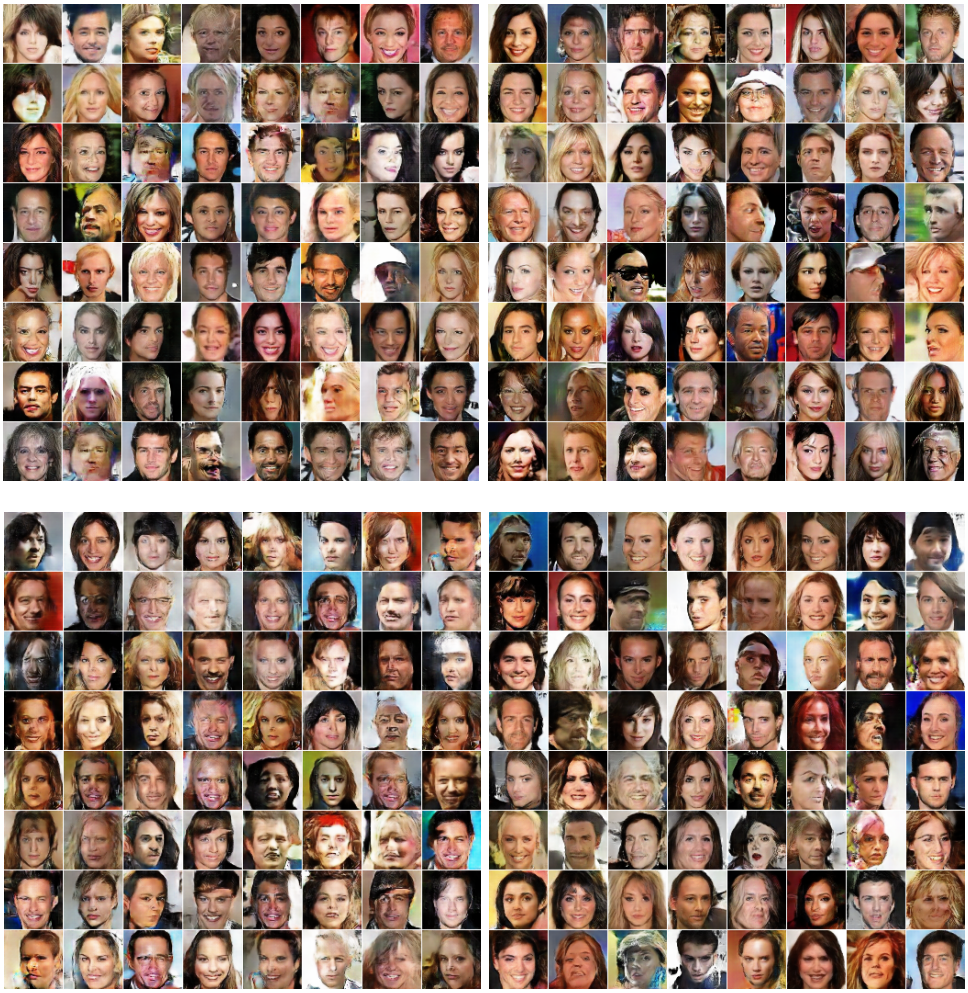


Figure 7: Best samples on CelebA according to visual inspection shown in Figure 4. *Top row*: (left) DCGAN (right) WGAN-GP. *Bottom row*: (left) AGE (right) α-GAN.

B Pseudocode

The overall training procedure is summarized in Algorithm 1.

C Reconstructions

We show reconstructions obtained using α-GAN and AGE for the CelebA dataset in Figure 8 and on CIFAR-10 in Figure 9.

Algorithm 1 Pseudocode for α-GAN

1: Initialize generator $\boldsymbol{\theta}$, encoder (variational distribution) $\boldsymbol{\eta}$ and discriminator $\boldsymbol{\phi}$ randomly.
2: Let $\hat{\mathbf{z}} \sim q_\eta(\mathbf{z}|\mathbf{x})$ denote a sample from the encoding variational distribution $q_\eta(\mathbf{z}|\mathbf{x})$ and $\hat{\mathbf{x}} = \mathcal{G}_{\boldsymbol{\theta}}(\hat{\mathbf{z}})$ denote the 'reconstruction' of $\mathbf{x}$ using $\hat{\mathbf{z}}$.
3: **for** iter $= 1 :$ max_iter **do**
4: Update encoder (variational distribution) $\boldsymbol{\eta}$ by minimizing
▷ *reconstruction and alternative loss from the code discriminator*

$$\mathbb{E}_{p^*(\mathbf{x})}\mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}\Big[\lambda||\mathbf{x} - \mathcal{G}_{\boldsymbol{\theta}}(\mathbf{z})||_1\Big] - \log \mathcal{C}_{\boldsymbol{\omega}}(\mathbf{z})\Big] \tag{10}$$

$$\approx \mathbb{E}_{p^*(\mathbf{x})}\Big[\lambda||\mathbf{x} - \hat{\mathbf{x}}||_1\Big] - \log \mathcal{C}_{\boldsymbol{\omega}}(\hat{\mathbf{z}})\Big] \tag{11}$$

5: Update generator $\boldsymbol{\theta}$ by minimizing ▷ *reconstruction and alternative loss*

$$\mathbb{E}_{p^*(\mathbf{x})}\mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}\big[\lambda||\mathbf{x} - \mathcal{G}_{\boldsymbol{\theta}}(\mathbf{z})||_1 - \log \mathcal{D}_{\boldsymbol{\phi}}(\mathcal{G}_{\boldsymbol{\theta}}(\mathbf{z}))\big] + \mathbb{E}_{p(\mathbf{z})}\big[-\log \mathcal{D}_{\boldsymbol{\phi}}(\mathcal{G}_{\boldsymbol{\theta}}(\mathbf{z}))\big] \tag{12}$$

$$\approx \mathbb{E}_{p^*(\mathbf{x})}\big[\lambda||\mathbf{x} - \hat{\mathbf{x}}||_1 - \log \mathcal{D}_{\boldsymbol{\phi}}(\hat{\mathbf{x}})\big] + \mathbb{E}_{p(\mathbf{z})}\big[-\log \mathcal{D}_{\boldsymbol{\phi}}(\mathcal{G}_{\boldsymbol{\theta}}(\mathbf{z}))\big] \tag{13}$$

6: Update discriminator $\boldsymbol{\phi}$ by minimizing
▷ *treat $p^*(\mathbf{x})$ as real, reconstructions and generated samples as fake*

$$\mathbb{E}_{p^*(\mathbf{x})}\big[-2\log \mathcal{D}_{\boldsymbol{\phi}}(\mathbf{x}) - \mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}\log\big(1 - \mathcal{D}_{\boldsymbol{\phi}}(\mathcal{G}_{\boldsymbol{\theta}}(\mathbf{z}))\big)\big] + \mathbb{E}_{p(\mathbf{z})}\big[-\log\big(1 - \mathcal{D}_{\boldsymbol{\phi}}(\mathcal{G}_{\boldsymbol{\theta}}(\mathbf{z}))\big)\big] \tag{14}$$

$$\approx \mathbb{E}_{p^*(\mathbf{x})}\big[-\log \mathcal{D}_{\boldsymbol{\phi}}(\mathbf{x}) - \log\big(1 - \mathcal{D}_{\boldsymbol{\phi}}(\hat{\mathbf{x}})\big)\big] + \mathbb{E}_{p(\mathbf{z})}\big[-\log\big(1 - \mathcal{D}_{\boldsymbol{\phi}}(\mathcal{G}_{\boldsymbol{\theta}}(\mathbf{z}))\big)\big] \tag{15}$$

7: Update code discriminator $\boldsymbol{\omega}$ by minimizing
▷ *treat $p(\mathbf{z})$ as real and codes from variational distribution as fake*

$$\mathbb{E}_{p^*(\mathbf{x})}\mathbb{E}_{q_\eta(\mathbf{z}|\mathbf{x})}\big[-\log \mathcal{C}_{\boldsymbol{\omega}}(\mathbf{z})\big] + \mathbb{E}_{p(\mathbf{z})}\big[-\log(1 - \mathcal{C}_{\boldsymbol{\omega}}(\mathbf{z}))\big] \tag{16}$$

$$\approx \mathbb{E}_{p^*(\mathbf{x})}\big[-\log \mathcal{C}_{\boldsymbol{\omega}}(\hat{\mathbf{z}})\big] + \mathbb{E}_{p(\mathbf{z})}\big[-\log(1 - \mathcal{C}_{\boldsymbol{\omega}}(\mathbf{z}))\big] \tag{17}$$

8: **end for**

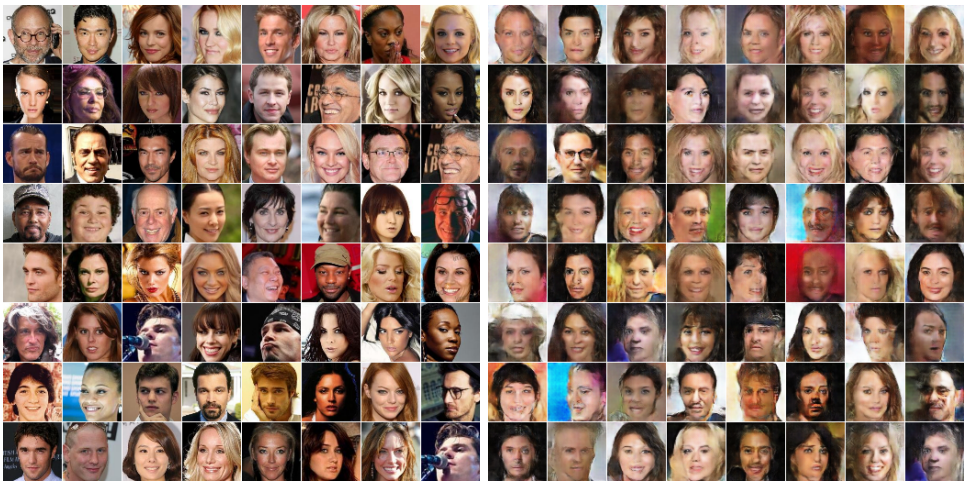
D Ablation experiment: code discriminator and the empirical KL

We have shown that we can estimate the KL term in (4) using the density ratio trick. In the case of a normal prior, another way to estimate the KL divergence on a mini-batch of latents each of dimension n, with per dimension sample mean and variance denoted by m_i and s_i ($i = 1...n$) respectively, is[4]:

$$\mathrm{KL}(q(z|x), N(0, I)) \approx \frac{n}{2} + \sum_{i=1}^{n}\Big(\frac{(s_i)^2 + (m_i)^2}{2} - \log(s_i)\Big) \tag{18}$$

In order to understand how the two different ways of estimating the KL term compare, we replaced the code discriminator in α-GAN with the KL approximation in (18). We then compared the results both by visual inspection (see CelebA and CIFAR-10 samples in Figure 10) and using the Inception score, Independent Wasserstein critic and sample diversity (Figure 13). We used the same hyperparameter sweeps for both methods (see Appendix H for details). In order to avoid be able to use the same hyperparameters for different latent sizes, we divide the approximation in (18) by the latent size. To also understand the effects of the two methods on the resulting autoencoder codes, we plot the means (Figure 11) and the covariance matrix (Figure 12) obtained from the a set of saved latent codes. Our results show that α-GAN using a code discriminator performs better on average than α-GAN using the empirical KL approximation, as it is less sensitive to hyperparamters. However, using the empirical KL we are able to obtain the best Inception score using α-GAN , of 7.15. By assessing the statistics of the final codes obtained by models trained using both approaches, we see that the two models of enforcing the prior have different side effects: the latent codes obtained using the code discriminator are decorrelated, while the ones obtained using the empirical KL are entangled; this is expected, since the correlation of latent dimensions is not modeled by (18), while the code discriminator can pick up that highly correlated codes are not from the same distribution as the prior.

[4]This approximation was also used by Ulyanov et al. [39] in AGE.



(a) AGE

(b) α-GAN

Figure 8: Training reconstructions obtained using AGE and α-GAN on CelebA.

While the code discriminator achieves better disentangling, the means obtained using the empirical KL are closer to 0, the mean of the prior distribution for each latent. We leave investigating these affects and combining the two approaches for future work.

E Monitoring overfitting of the independent Wasserstein critic

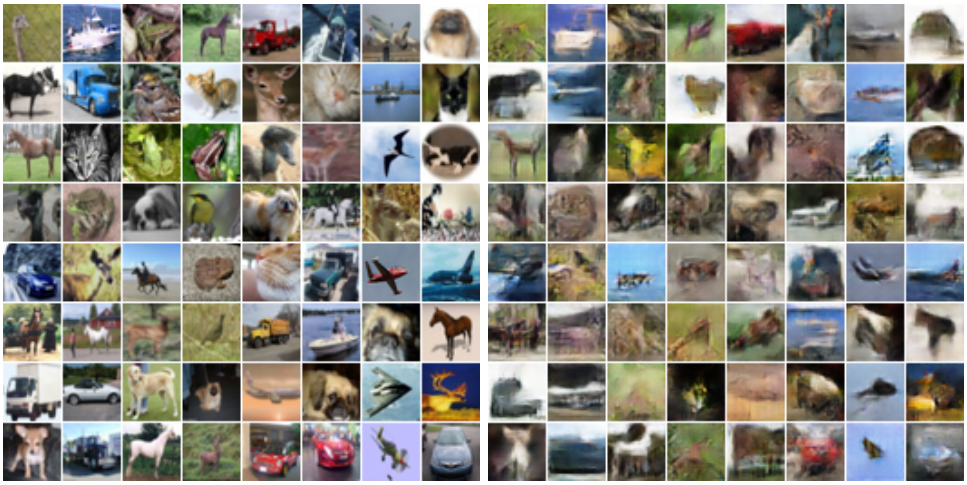
To ensure that the independent Wasserstein critic does overfit during training to the validation data, we monitor the difference in performance between training and test (see Figure 14).

F Best samples according to different metrics

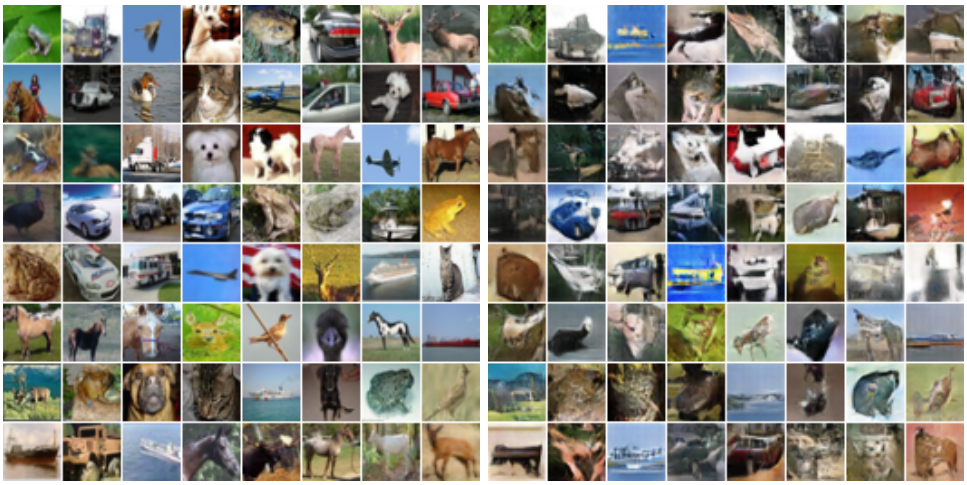
Figure 15 shows the best samples on CelebA according to different metrics.

G Relationships between different metrics

We assess the correlation between sample quality and how good a model is according to a independent Wasserstein critic in Figure 16.



(a) AGE



(b) α-GAN

Figure 9: Training reconstructions obtained using AGE and α-GAN on CIFAR-10. Left is the data and right are reconstructions.

H Training details: hyperparameters and network architectures

For all our models, we kept a fixed learning rate throughout training. We note the difference with AGE, where the authors decayed the learning rate during training, and changed the loss coefficients during training[5].). The exact learning rate sweeps are defined in Table 2. We used the Adam optimizer [19] with $\beta_1 = 0.5$ and $\beta_2 = 0.9$ and a batch size of 64 for all our experiments. We used batch normalization [17] for all our experiments. We trained all ColorMNIST models for 100000 iterations, and CelebA and CIFAR-10 models for 200000 iterations.

	Model			
Network	DCGAN	WGAN-GP	α-GAN	AGE
Generator/Encoder	$0.0001, 0.0002, 0.0003$	$0.0001, 0.0002, 0.0003$	$0.001, 0.0005$	$0.0001, 0.0002, 0.0005$
Discriminator	$0.0001, 0.0002, 0.0003$	$0.0001, 0.0002, 0.0003$	$0.001, 0.0005$	
Code discriminator			$0.001, 0.0005$	

Table 2: Learning rate sweeps performed for each model.

[5] As per advice found here: https://github.com/DmitryUlyanov/AGE/

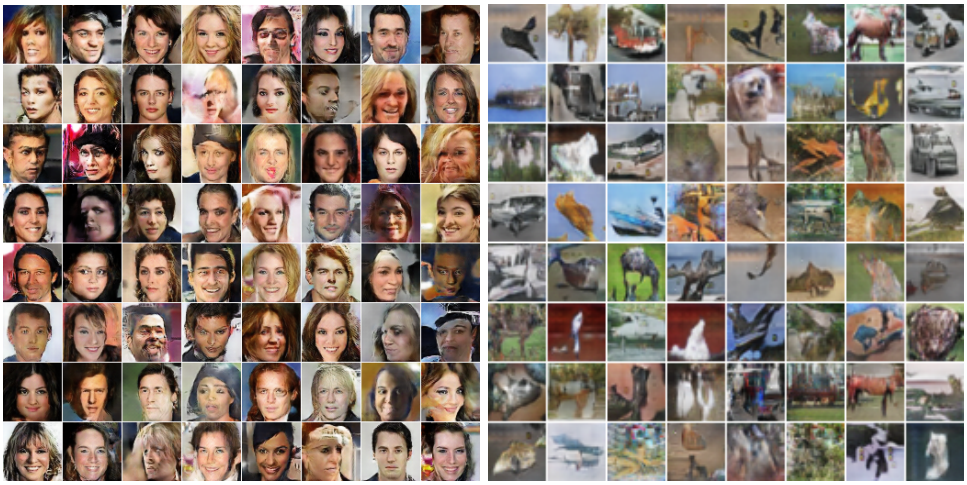


Figure 10: Samples from α-GAN on CelebA and CIFAR-10, trained using the empirical KL approximation (as opposed to a code discriminator) to make the posterior and the prior of the latents match.

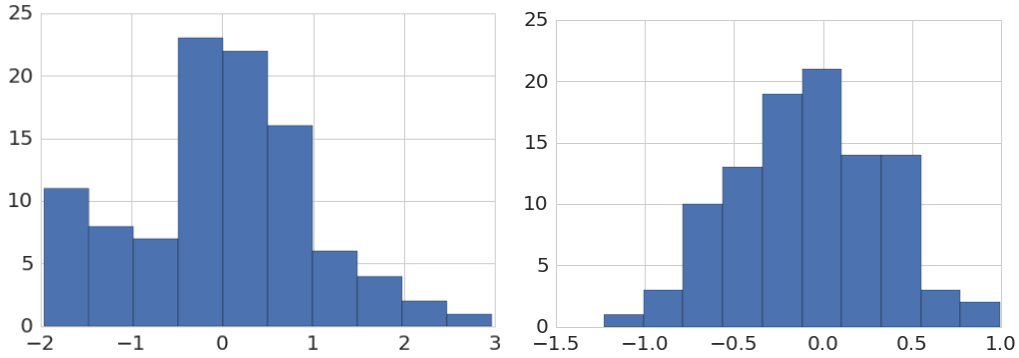


Figure 11: Histogram of latent means obtained on 64000 code representations from α-GAN trained using a code discriminator (left) and the empirical KL approximation (right). The latent size was 100. Since the prior was set to a normal with mean 0, we expect most means to be around 0. We note that the empirical KL seems better at forcing the means to be around 0.

H.1 Scaling coefficients

We used the following sweeps for the models which have combined losses with different coefficients (for all our baselines, we took the sweep ranges from the original papers):

- WGAN-GP
  - The gradient penalty of the discriminator loss function: 10.
- AGE
  - Data reconstruction loss for the encoder: sweep over 100, 500, 1000, 2000.
  - Code reconstruction loss for the generator: 10.
- α-GAN
  - Data reconstruction loss for the encoder: sweep over 1, 5, 10, 50.
  - Data reconstruction loss for the generator: sweep over 1, 5, 10, 50.
  - Adversarial loss for the generator (coming from the data discriminator): 1.0.
  - Adversarial loss for the encoder (coming from the code discriminator): 1.0.

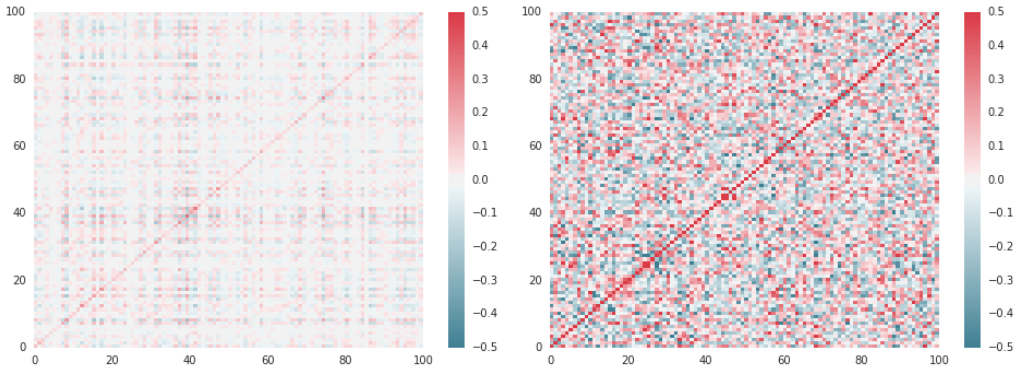
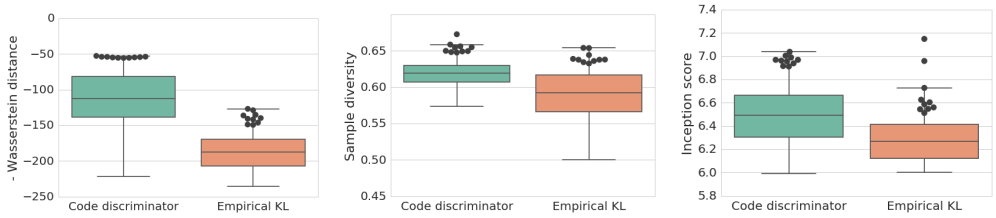


Figure 12: Covariance matrices obtained on 64000 code representations from α-GAN trained using a code discriminator (left) and the empirical KL approximation (right). The latent size was 100. We note that the code discriminator produces latents which have a lot less correlation than the empirical KL (which is what we want in this case, since the prior was a univariate Gaussian).



(a) Independent WGAN-GP critic-ColorMNIST (b) Sample diversity - CelebA (c) Inception-score - CIFAR-10

Figure 13: Comparing α-GAN the code discriminator and the empirical KL approach on different metrics and different datasets.

H.2 Choice of loss functions

For AGE, we used the l_1 loss as the data reconstruction loss, and we used the cosine distance for the code reconstruction loss. For α-GAN , we used l_1 as the data reconstruction loss and the traditional GAN loss for the data and code discriminator.

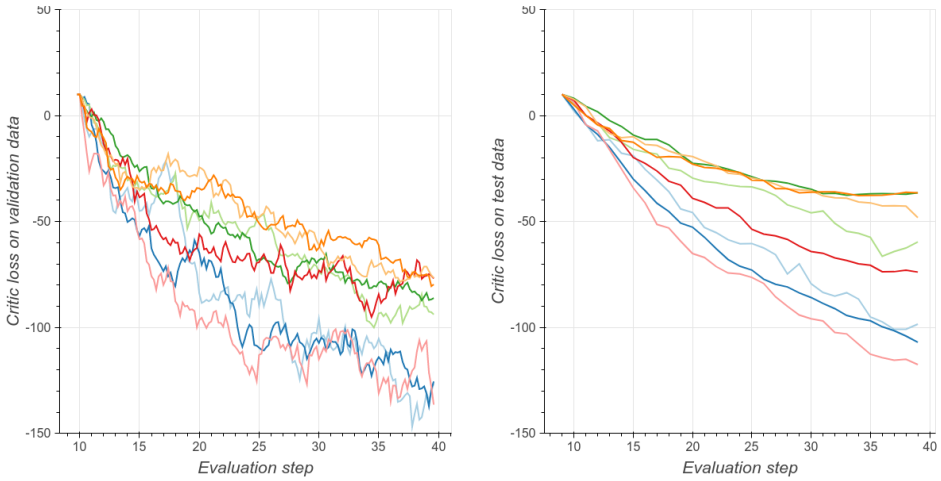


Figure 14: Training curves of the independent Wasserstein critic for different hyperparameter values. The model trained here is α-GAN , trained on CelebA. Left: the loss obtained on a mini-batch from the validation data. Right: the average loss obtained on the entire test set.

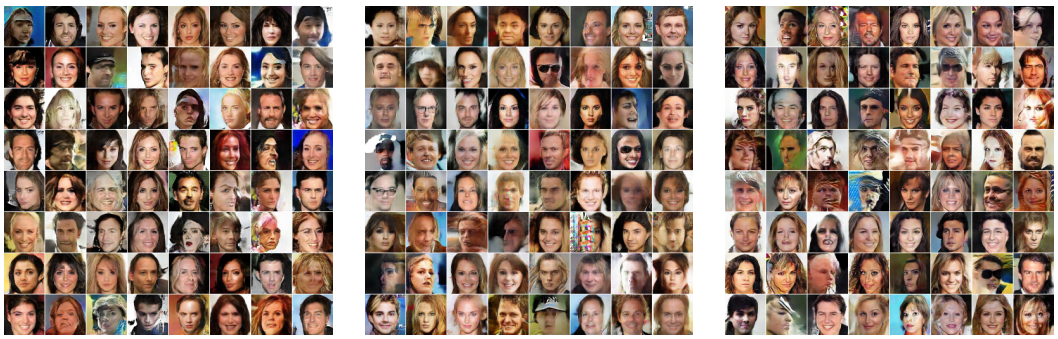


Figure 15: Best samples from α-GAN trained on CelebA according to different metrics: sample quality (left), independent Wasserstein critic (middle), sample diversity (right) given by 1-MSSSIM.

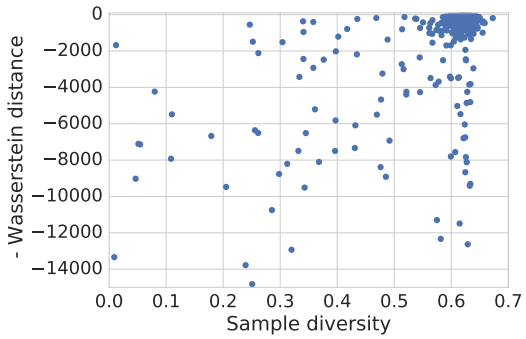


Figure 16: Correlation between sample diversity and the negative Wasserstein distance, obtained from a α-GAN hyperparameter sweep.

H.3 Choice of latent prior

We use a normal prior for all models, apart from AGE [39] which uses a uniform unit ball as the prior, and thus we project the output of the encoder to the unit ball.

H.4 Network architectures

For all our baselines, we used the same discriminator and generator architectures, and we controlled the number of latents for a fair comparison. For AGE we used the encoder architecture suggested by the authors[6], which is very similar to the DCGAN discriminator architecture. For α-GAN , the encoder is always set as a convolutional network, formed by transposing the generator (we do not use any activation function after the encoder). All discriminators and the AGE encoder use leaky units with a slope of 0.2, and all generators used ReLUs. For all our experiments using α-GAN , we used as a code discriminator a 3 layer MLP, each layer containing 750 hidden units. We did not tune the size of this network, and we postulate that since the prior latent distributions are similar (multi variate normals) between datasets, the impact of the architecture of the code discriminator is of less importance than the architecture of the data discriminator, which has to change from dataset to dataset (with the complexity of the data distribution). However, one could improve on our results by carefully tuning this architecture too.

H.4.1 ColorMNIST

For all our models trained on ColorMNIST, we swept over the latent sizes 10, 50 and 75. Tables 3 and 4 describe the discriminator and generator architectures respectively.

[6]Code at: https://github.com/DmitryUlyanov/AGE/

Operation	Kernel	Strides	Feature maps
Convolution	5×5	2×2	8
Convolution	5×5	1×1	16
Convolution	5×5	2×2	32
Convolution	5×5	1×1	64
Convolution	5×5	2×2	64
Linear adv	N/A	N/A	2
Linear class	N/A	N/A	10

Table 3: ColorMNIST discriminator architecture used for DCGAN, WGAN-GP and α-GAN. For DCGAN, we use dropout of 0.8 after the last convolutional layer. No other model uses dropout.

Operation	Kernel	Strides	Feature maps
Linear	N/A	N/A	3136
Transposed Convolution	5×5	2×2	64
Transposed Convolution	5×5	1×1	32
Transposed Convolution	5×5	2×2	3

Table 4: ColorMNIST generator architecture. This architecture was used for all 4 compared models.

H.4.2 CelebA and CIFAR-10

The discriminator and generator architectures used for CelebA and CIFAR-10 were the same as the ones used by Gulrajani et al. [12] for WGAN-GP.[7]

H.5 CIFAR-10 classifier used for Inception score

We used a VGG style [33] convnet trained on CIFAR-10 as the classifier network used to report the inception score in Section 5. The architecture is described in Table 5. We use batch normalization after each convolutional layer. The data is rescaled to be in range $[-1, 1]$, and during training the input images are randomly cropped to size $(24, 24, 3)$. We used a momentum optimizer with learning rate starting at 0.1 and decaying by 0.1 at timesteps 40000 and 60000, with momentum set at 0.9. We used an l_2 regularization penalty of $1e-4$. The network was trained for 80000 epochs, using a batch size of 256 (8 synchronous workers, each having a batch size of 32). The resulting network achieves an accuracy of 5.5% on the official CIFAR-10 test set.

Operation	Kernel	Strides	Feature maps
Convolution	3×3	2×2	64
Convolution	3×3	1×1	64
Convolution	3×3	2×2	128
Convolution	3×3	1×1	128
Convolution	3×3	2×2	128
Convolution	3×3	2×2	256
Convolution	3×3	2×2	256
Convolution	3×3	2×2	256
Convolution	3×3	2×2	512
Convolution	3×3	2×2	512
Convolution	3×3	2×2	512
Average pooling	N/A	N/A	N/A
Linear class	N/A	N/A	10

Table 5: The neural network trained to classify CIFAR-10 data.

[7] Code at: https://github.com/martinarjovsky/WassersteinGAN/blob/master/models/dcgan.py