InvGAN: Invertible GANs

Partha Ghosh[1], Dominik Zietlow[1,2], Michael J. Black[1,2] Larry S. Davis[2], and Xiaochen Hu[2]

[1] MPI for Intelligent Systems, Tübingen pghosh@tuebingen.mpg.de
[2] Amazon.com, Inc. {zietld, mjblack, lrrydav, sonnyh}@amazon.com

Abstract. Generation of photo-realistic images, semantic editing and representation learning are only a few of many applications of high-resolution generative models. Recent progress in GANs have established them as an excellent choice for such tasks. However, since they do not provide an inference model, downstream tasks such as classification cannot be easily applied on real images using the GAN latent space. Despite numerous efforts to train an inference model or design an iterative method to invert a pre-trained generator, previous methods are dataset (e.g. human face images) and architecture (e.g. StyleGAN) specific. These methods are nontrivial to extend to novel datasets or architectures. We propose a general framework that is agnostic to architecture and datasets. Our key insight is that, by training the inference and the generative model together, we allow them to adapt to each other and to converge to a better quality model. Our **InvGAN**, short for Invertible GAN, successfully embeds real images in the latent space of a high quality generative model. This allows us to perform image inpainting, merging, interpolation and online data augmentation. We demonstrate this with extensive qualitative and quantitative experiments.

1 Introduction

The ability to generate photo-realistic images of objects such as human faces or fully clothed bodies has wide applications in computer graphics and computer vision. Traditional computer graphics, based on physical simulation, often fails to produce photo-realistic images of objects with complicated geometry and material properties. In contrast, modern data-driven methods, such as deep learning-based generative models, show great promise for realistic image synthesis [23,24]. Among the four major categories of generative models –generative adversarial networks (GANs), variational auto-encoders (VAEs), normalizing flows and autoregressive models– GANs deliver images with the best visual quality. Although recent efforts in VAEs [13,34] have tremendously improved their generation quality, they still use larger latent space dimensions and deliver lower quality images. Autoregressive models are very slow to sample from and do not provide a latent representation for the trained data. Flow-based methods do not perform dimensionality reduction and hence produce large models and latent representations. On the other hand, GANs offer great generation quality, but do not provide a mechanism to embed real images into the latent space. This limits them as a tool for image editing and manipulation. Specifically, while several

methods exist [6,46,2,4], there is no method that trains the generative and the inference model together[3]. To that end, we propose *InvGAN*, an invertible GAN in which the discriminator acts as an inference module. InvGAN enables a wide range of applications, as described in the following paragraphs.

GANs learn a latent representation of the training data. This representation has been shown to be well-structured [10,23,24], allowing GANs to be employed for a variety of downstream tasks (e.g. classification, regression and other supervised tasks) [28,33]. We extend the GAN framework to include an inference model that embeds real images into the latent space. InvGAN can be used to support representation learning [11,27], data augmentation [10,37] and algorithmic fairness [7,38,39]. Previous methods of inversion rely on computationally expensive optimization of inversion processes [3,4], limiting their scope to offline applications, e.g. data augmentation has to happen before training starts. Efficient, photo-realistic, semantically consistent, and model-based inversion is the key to online and adaptive use-cases.

Recent work shows that even unsupervised GAN training isolates several desirable generative characteristics [29,43]. Prominent examples are correspondences between latent space directions and e.g. hairstyle, skin tone and other visual characteristics. Recent works provide empirical evidence suggesting that one can find paths in the latent space (albeit non-linear) that allow for editing individual semantic aspects. GANs therefore have the potential to become a high-quality graphics editing tool [18,41]. However, without a reliable mechanism for projecting real images into the latent space of the generative model, editing of real data is impossible. InvGAN take a step towards addressing this problem.

2 Related work

The task, GAN inversion, refers to the task of (approximately) inverting the generator network. It has been addressed in two primary ways (1) using an inversion model (often a deep neural network), (2) using an iterative optimization-based method, typically initialized with (1). Although, invertibility of generative models span beyond specific data domains (images, speech, language etc.), we study InvGAN applied to image data only. Its applications of generation of sound, language etc.is left as future work.

Optimization based: iGAN [54] optimizes for a latent code while minimizing the distance between a generated image and a source image. To ensure uniqueness of the preimage of a GAN-generated data point, Zachary et al. [26] employ stochastic clipping. As the complexity of the GAN generators increases, an inversion process based on gradient descent and pixel space MSE is insufficient. Addressing this, Rameen et al. specifically target StyleGAN generators and optimize for perceptual loss [3,4]. However, they invert into the $W+$ space, the so-called extended W space of StyleGAN. This results in high dimensional latent codes and consequently prolongs inversion time. This can also produce out-of-distribution latent representations, which makes them unsuitable for down-

[3] Except for BiGAN [14] and ALI [16]. We discuss the differences in Sec. 2

stream tasks. Contrary to these, InvGAN offers fast inference embedding in the non-extended latent space.

Model based: BiGAN [14] and ALI [16] invert the generator of a GAN during the training process by learning the joint distribution of the latent vector and the data in a completely adversarial setting. However, the quality is limited, partially because of the choice of DCGAN [32] and partially because of the significant dimensionality and distribution diversity between the latent variable and the data domain [15]. More recent models target the StyleGAN architecture [35,44,53] and achieve impressive results. Most leverage StyleGAN peculiarities, i.e., they invert in the $W+$ space, so adaptation to other GAN backbones is non-trivial. Adversarial latent auto-encoders [31] are closest to our current work. Our model and adversarial autoencoders can be made equivalent with a few alterations to the architecture and to the optimization objective. We discuss this more in detail in Section 3.2. Our method, on the other hand, neither uses any data set specific loss nor does it depend upon any specific network architecture.

Hybrid optimization and regression based: Guan et al. [20] train a regressor that is used to initialize an optimization-based method to refine the regressor's guess. However, is specific to human face datasets. Zhu et al. [52] modify the general hybrid approach with an additional criterion that encourages the recovered latent must belong to the semantically meaningful region learned by the generator by assuming that the real image can be reconstructed more faithfully in the immediate neighbourhood of an initial guess given by a model-based inversion mechanism. Yuval et al. [5] replace gradient-based optimization with an iterative encoder that encodes the current generation and target image to the estimated latent code difference. They empirically show that this iterative process converges and that the recovered image improves over iterations. However, this method requires multiple forward passes in order to achieve a suitable latent code. In contrast to the work above, the inference module obtained by our method infers the latent code in one shot. Hence, it is much faster and does not run the risk of finding a non-meaningful latent code.

The inversion mechanisms presented so far do not directly influence the generative process. In most of the cases, they are conducted on a pre-trained frozen generator. Although in the case of ALI [16] and BiGAN [14] the inference model loosely interacts with the generative model at training time, the interaction is only indirect; i.e. through the discriminator. In our work, we tightly couple the inference module with the generative module.

Joint training of generator and inference model: We postulate that jointly training an inference module will help regularize GAN generators towards invertibility. This is inspired by the difficulty of inverting a pre-trained high-performance GAN. For instance, Bau et al. [8] invert PGAN [22], but for best results a two-stage mechanism is needed. Similarly, Image2StyleGAN [2] projects real images into the extended W^+ space of StyleGAN, whereas, arguably, all the generated images can be generated from the more compact z or w space. This is further evident from Wulff et al. [45] who find an intermediate latent space in StyleGAN that is more Gaussian than the assumed prior. However, they too

use an optimization-based method and, hence, it is computationally expensive and at the same time specific to both the StyleGAN backend and the specific data set. Finally, we refer the readers to 'GAN Inversion: A Survey' [47] for a comprehensive review of relate work.

3 Method

Goal: Our goal is to learn an inversion module alongside the generator during GAN training. An inversion module is a mechanism that returns a latent embedding of a given data point. Specifically, we find a generator $G : \mathbb{W} \to \mathbb{X}$ and an inference model $D : \mathbb{X} \to \mathbb{W}$ such that $x \approx G(D(x \sim \mathbb{X}))$, where $\mathbb{X}$ denotes the data domain and $\mathbb{W}$ denotes the latent space. We reuse the GAN discriminator to play the role of this inference model D in practice.

3.1 Architecture

We demonstrate InvGAN using DC-GAN, BigGAN and StyleGAN as the underlying architectures. Figure 1 represents the schematic of our model. We follow the traditional alternate generator-discriminator training mechanism. The generative part consists of three steps 1. sampling latent code: $z \sim \mathcal{N}(0, I)$, 2. mapping the latent code to w space: $w = M(z)$, 3. using mapped code to generate fake data: $x = G(w)$, where M is a mapping network, G is the generator, D is the discriminator, and $\mathcal{N}(0, I)$ is the standard normal distribution. In practice, the discriminator, besides outputting real/fake score, also outputs inferred w parameter, which was found to work better empirically over designs with two different networks for discrimination and inference. From here on, we use $\tilde{w}, c = D(x)$ to denote the inferred latent code ($\tilde{w}$) using the discriminator D and c to denote the real-fake classification decision for the sample $x \in \mathbb{X}$. Wherever obvious, we simply use $D(x)$ to refer to c, the discrimination decision only.

3.2 Objective

GAN Objective: The min-max game between the discriminator network and the generator network of vanilla GAN training is described as

$$\min_{G,M} \max_{D} \mathcal{L}_{\text{GAN}} = \min_{G,M} \max_{D} \left[\mathbb{E}_{x \in \mathbb{X}}[\log D(x)] + \mathbb{E}_{z \in \mathbb{Z}}[\log(1 - D(G(M(z))))]\right]. \tag{1}$$

A naive attempt at an approximately invertible GAN would perform $\min_G \max_D L_{\text{GAN}} + \min_{G,D} \|w - \tilde{w}\|_p$, where $\|\bullet\|_p$ denotes an L_p norm. This loss function can be interpreted as optimal transport cost. We discuss this in more detail at the end of this section. However, this arrangement, coined the "naive model", does not yield satisfactory results, cf. Section 4.4. This can be attributed to three factors: (1) w corresponding to real images are never seen by the generator; (2) no training signal is provided to the discriminator for inferring the latent code corresponding to real images (w_R); (3) the distribution of w_R might differ from prior distribution of w. We address each of these concerns with a specific loss term designed to address the said issues. Our naive model

corresponds to the adversarial autoencoders [31] if the real-fake decision is derived from a common latent representation. However, this forces the encoding of real and generated images to be linearly separable and contributes to degraded inference performance.

Minimizing latent space extrapolation: Since, in the naive version, neither the generator nor the discriminator gets trained with w_R, it relies completely upon its extrapolation characteristics. In order to reduce the distribution mismatch for the generator, we draw half the mini batch of latent codes from the prior and the other half consists of w_R; i.e., $w_{\text{total}} = w +\!\!+ w_R,\ w \sim P(W)$ where $+\!\!+$ denotes a batch concatenation operation. By $w \sim P(W)$, we denote the two stage process given by the following $w = M(z \sim P(Z))$. Together with the naive loss this forms the first three terms of our full objective function given in Equation 3

Pixel space reconstruction loss: Since latent codes for real images are not given, the discriminator cannot be trained directly. However, we recover a self-supervised training signal by allowing the gradients from the generator to flow into the discriminator. Intuitively, the discriminator tries to infer latent codes from real images that help the generator reproduce that image. As shown in Section 4.4, this improves real image inversion tremendously. We enforce further consistency by imposing an image domain reconstruction loss between input and reconstructed real images. However, designing a meaningful distance function for images is a non-trivial task. Ideally, we would like a feature extractor function f that extracts low- and high-level features from the image such that two images can be compared meaningfully. Given such a function, a reconstruction loss can be constructed as

$$\mathcal{L}_{\text{fm}} = \|\mathbb{E}_{x \in \mathbb{X}}(f(x) - f(G(w \sim P(W|x)))\|_p \tag{2}$$

A common practice in the literature is to use a pre-trained VGG [21,51] network as a feature extractor f. However, it is well known that deep neural networks are susceptible to adversarial perturbations [48]. Given this weakness, optimizing for perceptual loss is error-prone. Hence, a combination of a pixel-domain L_2 and feature-space loss is typically used, but this often results in degraded quality. Consequently, we take the discriminator itself as the feature extractor function f. Due to the min-max setting of GAN training, we are guaranteed to avoid the perils of adversarial and fooling samples, if we use the discriminator features, instead of VGG features. The feature loss is shown in the second half of Figure 1. Although this resembles the feature matching described by Salimans et al. [36], it has a crucial difference. As seen in Equation 2 the latent code fed into the generator is drawn from the conditional distribution $P(W|x) := \delta_{D(x)}(w)$ rather than the prior $P(W)$, where $\delta(x)$ represents the Dirac delta function located at x. This forces the distribution of the features to match more precisely as compared to the simple first-moment matching proposed by Salimans et al. in [36].

Addressing mismatch between prior and posterior: Finally, we address the possibility of mismatch between inferred and prior latent distributions (point

(3) described above) by imposing a maximum mean discrepancy (MMD) loss between the sets of samples of the said two distributions. We use an RBF kernel to compute this. The loss improves the random sampling quality by providing a direct learning signal to the mapping network. This forms the last term of our objective function as shown in Equation 3.

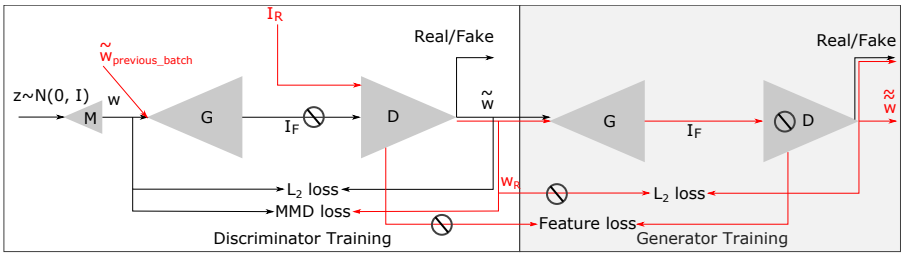


Fig. 1: We train InvGAN following a regular GAN. We use a second output head in the discriminator besides the real fake decision head, to infer the latent-code z of a given image. Here $\oslash$ denotes no gradient propagation during back propagation step. It also denotes 'no training' when it is placed on a model. We use red color to show data flow corresponding to real images.

Putting everything together gives the objective of our complete model. It is as shown in Equation 3. Note that here the expectation operator $\mathbb{E}_{w+w_R}$ acts on several loss terms that are independent of w_R or w. In such cases keeping in mind the identity $c = \mathbb{E}[c]$, where c is a constant, can add clarity. Furthermore, here and in the rest of the paper we use a plus operator $+$, between two optimization process, to indicate that both of them are performed simultaneously.

$$\min_{G,M}\Bigg[\max_{D}\mathcal{L}_{\text{GAN}} + \min_{D}\Bigg[\mathbb{E}_{w+w_R}\big[\,\|M(z)-\tilde{w}\|_2^2 + \big\|(\tilde{w}+w_R)-\tilde{\tilde{w}}\big\|_2^2 + \mathcal{L}_{\text{fm}} + \text{MMD}\{w, w_R\}\big]\Bigg]\Bigg] \tag{3}$$

An optimal transport based interpretation: Neglecting the last three terms described in Equation 3, our method can be interpreted as a Wasserstein autoencoder (the GAN version) (WAE-GAN) [42]. Considering a WAE with its data domain set to our latent space and its latent space assigned to our image domain, if the encoder and the discriminator share weights the analogy is complete. Our model can, hence, be thought of as learning the latent variable model $P(W)$ by randomly sampling a data point $x \sim \mathbb{X}$ from the training set and mapping it to a latent code w via a deterministic transformation. In terms of density, it can be written as in Equation 4.

$$P(W) := \int_{x\in\mathbb{X}} P(w|x)P(x)\mathrm{d}x. \tag{4}$$

As proven by Olivier et al. [9], under this model the optimal transport problem $W_c(P(W), P_D(W)) := \inf_{\Gamma\in P(w_1\sim P(W), w_2\sim P_D(W))}\left[\mathbb{E}_{w_1,w_2\sim\Gamma}\left[c(w_1,w_2)\right]\right]$ can be

solved by finding a generative model $G(X|W)$ such that its X marginal, $P_G(X) = \mathbb{E}_{w\sim P(W)}G(X|w)$ matches the image distribution $P(X)$. We ensure this by considering the Jensen–Shannon divergence $D_{\text{JS}}(P_G(X), P(X))$ using a GAN framework. This leads to the cost function given in Equation 5, when we choose the ground cost function $c(w_1, w_2)$ to be squared L_2 norm.

$$\min_{G,M} \max_{D} \mathcal{L}_{\text{GAN}} + \min_{G,M} \min_{D} \|w - \tilde{w}\|_2^2 \tag{5}$$

Finally, we find that by running the encoding/decoding cycle one more time, we can impose several constraints that improve the quality of the encoder and the decoder network in practice. This leads to our full optimization criterion, as described in Equation 3. Note that our method because of this extra cycle is less efficient computationally as compared to vanilla VAEs or WAEs, but by incurring this computational penalty we successfully avoid having to define a loss function in the image domain. This results in sharper image generation.

3.3 Dealing with resolutions higher than the training resolution

Although StyleGAN [24] and BigGAN [10] have shown that it is possible to generate relatively high-resolution images, in the range of 1024×1024 and 512×512, their training is resource intense and the models are difficult to tune for new data sets. Equipped with invertibility, we explore a tiling strategy to improve the output resolution. First, we train an invertible GAN at a lower resolution ($m \times m$) and simply tile them $n \times n$ times with n^2 latent codes to obtain a higher resolution ($mn \times mn$) final output image. The new latent space containing n^2 latent codes obtained using the inference mechanism of the invertible GAN can now be used for various purposes, as described in Section 4.3 and reconstructions are visualized in Figure 3. This process correlates in spirit somewhat to COCO-GAN [25]. The main difference, however, is that our model at no point learns to assemble neighbouring patches. Indeed, the seams are visible if one squints at the generated images, e.g., in figure 3. However, a detailed study of tiling for generation of higher resolution images than the input domain is beyond the scope of our paper. We simply explore some naive settings and their applications in section 4.3.

4 Experiments

We test InvGAN on several diverse datasets (MNIST, ImageNet, CelebA, FFHQ) and multiple backbone architectures (DC-GAN, BigGAN, StyleGAN). For the mapping network in the generator, we use the standard 8-layer mapping network with StyleGAN and add a 2-layer mapping network to BigGAN and DC-GAN. Our method is evaluated both qualitatively (via style mixing, image inpainting etc.) and quantitatively (via the FID score and the suitability for data augmentation for discriminative tasks such as classification). We found that relative weights of different terms in our objective function do not impact the model's performance significantly. Therefore, keeping simplicity in mind, we avoid tuning and simply set them to be one.

Table 1 shows random sample FIDs, middle point linear interpolation FIDs and test set reconstruction mean absolute errors (MAEs) of our generative model. We note here that interpolation FID and random generation FID are comparable to non-inverting GANs. This leads us to conclude that the inversion mechanism does not adversarially impact the generative properties. We provide a definition, baseline and understanding of inversion of a high-quality generator for uniform comparison of future works on GAN inversion. We highlight model-based inversion, joint training of generative and inference model and its usability in downstream tasks. We demonstrate that InvGAN generalizes across architectures, datasets, and types of downstream task.

Models	**RandFID**	**RandRecFID**	**TsRecFID**	**IntTsFID**	**MAE±1**	**Run Time**
FFHQ [49]	49.65/14.59	56.71/23.93	-/13.73	68.45/38.01	0.129	0.045
FFHQ Enc.[52]	46.82/14.38	-/-	88.48 / -	-/-	0.460	
FFHQ MSE opt.[52]	46.82/14.38	-/-	58.04 / -	-/-	0.106	
FFHQ In-D. Inv.[52]	46.82/14.38	52.02/-	42.64 / -	71.83/-	0.115	99.76
DCGAN, MNIST	17.44/6.10	16.76/4.25	17.77/4.70	26.04/11.44	0.070	$3.3 \cdot 10^{-5}$
StyleGAN, CelebA	26.63/4.81	24.35/3.51	24.37/4.14	32.37/15.60	0.150	$1.0 \cdot 10^{-3}$
StyleGAN, FFHQ	49.14/12.12	44.42/8.85	41.14/7.15	49.52/14.36	0.255	$2.0 \cdot 10^{-3}$

Table 1: Here we report random sample FID (RandFID), FID of reconstructed random samples (RandRecFID), FID of reconstructed test set samples (TsRecFID), FID of the linear middle interpolation of test set images (IntTsFID) and reconstruction per pixel per color channel mean absolute error when images are normalized between ±1, also from test set. All FID scores are here evaluated against train set using 500 and 50000 samples. They are separated by '/'. For the traditional MSE optimization based and In-Domain GAN inversion, the MSE errors are converted to MAE by taking square root and averaging over the color channels and accounting for the re-normalization of pixel values between ±1 (MAE±1). Runtime is given in seconds per image. We ran them on a V100 32GB GPU and measured wall clock time.

We start with a StyleGAN-based architecture on FFHQ and CelebA for image editing. Then we train a BigGAN-based architecture on ImageNet, and show super resolution and video key-framing by tiling in the latent domain to work with images and videos that have higher resolution than training data. We also show ablation studies with a DC-GAN-based architecture on MNIST. In the following sections we evaluate qualitatively by visualizing semantic editing of real images and quantitatively on various downstream tasks including classification fairness, image super resolution, image mixing, etc.

4.1 Semantically consistent inversion using InvGAN

GANs can be used to augment training data and substantially improve learning of downstream tasks, such as improving fairness of classifiers of human-facial attributes [39,38,7,33]. There is an important shortcoming in using existing GAN approaches for such tasks: the labeling of augmented data relies on methods that are trained independently on the original data set, using human annotators or

compute-expensive optimization-based inversion. A typical example is data-set debasing by Ramaswamy et al. [33]. For each training image, an altered example that differs in some attribute (e.g. age, hair color, etc.) has to be generated. This can be done in one of two ways, 1. by finding the latent representation of the ground truth image via optimization and 2. by labeling random samples using pre-trained classifiers on the biased data set. Optimization-based methods are slow and not a viable option for on-demand/adaptive data augmentation. Methods using pre-trained classifiers inherit their flaws, e.g. spurious correlation induced dependencies. However, having access to a high-quality inversion mechanism help us overcome such problems [55].

To verify that InvGAN is indeed suitable for such tasks, we train ResNet50 attribute classifiers on the CelebA dataset. We validate that the encoding and decoding of InvGAN results in a semantically consistent reconstruction by training the classifier only on reconstructions of the full training set. As a baseline, we use the same classifier trained on the original CelebA. We produce two reconstructed training sets by using the tiling-based inversion (trained on ImageNet) and by training InvGAN on CelebA (without tiling). For each attribute, a separate classifier has been trained for 20 epochs. The resulting mean average precisions are reported in Table 2. We see that training on the reconstructions allows for very good domain transfer to real images, indicating that the reconstruction process maintained the semantics of the images.

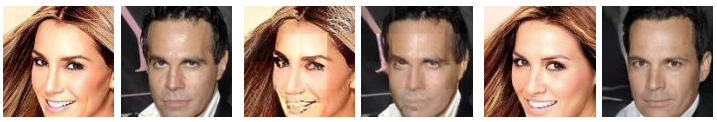
Train on → **Eval. on ↓**	**Original**	**Tile Recon.**	**Full Recon.**
Original	0.81 ± 0.15	0.77 ± 0.16	0.79 ± 0.15
Tile recon.	0.79 ± 0.16	0.80 ± 0.15	0.78 ± 0.16
Full recon.	0.81 ± 0.15	0.78 ± 0.16	0.81 ± 0.14
Recon. Vis.			

Table 2: Mean average precision for a ResNet50 attribute classifier on CelebA, averaged over 20 attributes. We report the performance for training on the original dataset, the reconstructed dataset using the tiling-based method pre-trained on ImageNet and the reconstruction on InvGAN trained on the CelebA training set directly.

4.2 Suitability for image editing

GAN inversion methods have been proposed for machine supported photo editing tasks [52,12,30]. Although there is hardly any quantitative evaluation for the suitability of a specific inversion algorithm or model, a variety of representative operations have been reported [4,3,52]. Among those are in-painting cut out regions of an image, image-merging and improving on amateur manual photo

editing. Figures 2 and 8 in the appendix visualize those operations performed on FFHQ and CelebA images, respectively. We demonstrate in-painting by zeroing out a randomly positioned square patch and then simply reconstructing the image. Image-merging is performed by reconstructing an image which is composed out of two images by simply placing them together. By reconstructing an image that has undergone manual photo editing, higher degrees of photo-realism are achieved. Quantitative metric for such tasks are hard to define and hence is scarcely found in prior art, since they depend upon visual quality of the results. We report reconstruction and interpolation FIDs in Table 1, in an effort to establish a baseline for future research. However, we do acknowledge that a boost in pixel fidelity in our reconstruction will greatly boost the performance of InvGAN on photo editing tasks. The experiments clearly show the general suitability of the learned representations to project out of distribution images to the learned posterior manifold via reconstruction.

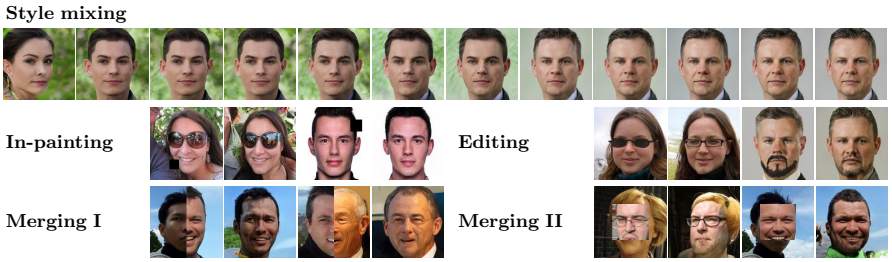
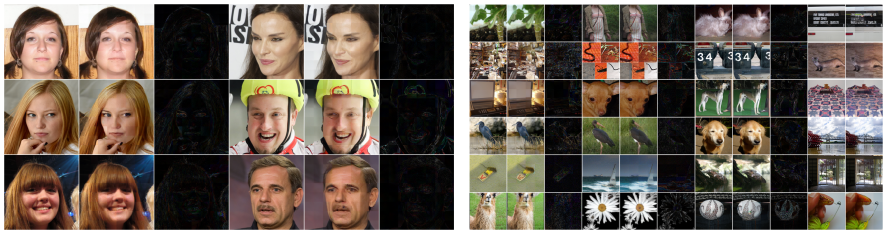


Fig. 2: Benchmark image editing tasks on FFHQ (128 px). Style mixing: We transfer the first $0, 1, 2, \ldots, 11$ style vectors from one image to another. For the other image editing tasks, pairs of images are input image (left) and reconstruction (right).

4.3 Tiling to boost resolution

Limitations in video RAM and instability of high resolution GANs are prominent obstacles in generative model training. One way to bypass such difficulties is to generate the image in parts. Here we train our invertible generative model, a BigGAN architecture, on 32×32 random patches from ImageNet. Once the inversion mechanism and the generator are trained to satisfactory quality, we reconstruct both FFHQ and ImageNet images. We use 256×256 resolution and tile 32 patches in an 8×8 grid for FFHQ images, and 128×128 resolution and tiling 32 patches in a 4×4 grid for ImageNet images. The reconstruction results are shown in Figure 3. Given the successful reconstruction process, we explore the tiled latent space for tasks such as image deblurring and time interpolation of video frames.

Image de-blurring: Here we take a low-resolution image, scale it to the intended resolution using bicubic interpolation, invert it patch by patch, Gaussian blur it, invert it again and linearly extrapolate it in the deblurring direction.



(a) (b)

Fig. 3: Tiled reconstruction of random (a) FFHQ Images and (b) ImageNet images. The left column shows the real images, the second shows the patch by patch reconstructions, and the third shows the absolute pixel-wise differences. Note that interestingly though the patches are reconstructed independent of each other, the errors lie mostly on the edges of the objects in the images, arguably the most information dense region of the images.

The deblurring direction is simply obtained by subtracting the latent code of the given low resolution but bicubic up sampled image from the latent code of the blurred version of it at the same resolution. The exact amount of extrapolation desired is left up to the user. In Figure 4 we show the effect of three different levels of extrapolation. Although our method is not trained for the task of super resolution, by virtue of a meaningful latent space we can enhance image quality.

Fig. 4: Super resolution using extrapolation in the tiled latent space. From left, we visualize the original image, the low-resolution version of it, the reconstruction of the low-resolution version, and progressive extrapolation to achieve deblurring.

Temporal interpolation of video frames: Here we boost the frame rate of a video post capture. We infer the tiled latent space of consecutive frames in a video, and linearly interpolate each tile to generate one or more intermediate frames. Results are shown in Figure 6 in the appendix and in the accompanying videos in supplementary material. We find the latent code of each frame in a video sequence, and then derive intermediate latent codes by weighted averaging neighboring latent codes using a Gaussian window. We use the UCF101 data set [40] for this task. Note however that since there is no temporal constraint and

each patch is independently interpolated, one can notice flicker effect. Further-studies into this matter are left to future work.

As can also be seen, this process produces discontinuities at the boundaries of the patches. This is because neighboring patches are modeled independently of one another in this work. This can be dealt with in a variety of ways, namely, by carefully choosing overlapping patch patterns and explicitly choosing a predetermined or learning-based stitching mechanism, seam detection and correction [1,50]. However, a thorough study in this direction is considered out of scope for the current work.

4.4 Ablation studies

Recall that the naive model defined in Section 3.2 uses the optimization $\min_{G,M}\left[\max_D L_{\text{GAN}} + \min_D \mathbb{E}_{z\sim P(Z)} \left\|M(z) - \tilde{w}\right\|_p\right]$ to train (also given in Equation 5), In Figure 5a we show how our three main components progressively improve the naive model. As is apparent from the method section, the first major improvement comes from exposing the generator to the latent code inferred from real images. This is primarily due to the difference in the prior and the induced posterior distribution. This is especially true during early training, which imparts a lasting impact. The corresponding optimization is $\min_{G,M}\left[\max_D L_{\text{GAN}} + \min_D \mathbb{E}_{w=M(z\sim P(Z))+w_R} \left\|w - \tilde{w}\right\|_p\right]$. This simply reduces the distribution mismatch between prior and posterior by injecting inferred latent codes, improves inversion quality. This is visualized in Figure 5b. We call this model the augmented naive model. However, this modification unlocks the possibility to enforce back propagation of generator loss gradients to the discriminator and real-image, generated-image pairing, as detailed in Section 3.2. This leads to our model and the results are visualized in Figure 5c.

(a) (b) (c)

Fig. 5: Inversion of held out test samples. Columns are in groups of three: the first column holds real images, second their reconstruction and third the absolute pixel-wise difference. (a) Inversion using naive model, i.e. only z reconstruction loss is used, (b) inversion using model that uses latent codes from real samples, i.e., the augmented naive model. (c) our full model. Notice how the imperfections in the reconstructions highlighted with red boxes gradually vanishes as the model improves.

5 Discussion and future work

While InvGAN can reliably invert the generator of a GAN, it still can benefit from an improved reconstruction fidelity for tasks such as image compression, image segmentation, etc. We observe that the reconstruction of rare features, such as microphones, hats or background, tend to have lower quality, as seen in the appendix in Figure 7 bottom row 3rd and 4th columns. This combined with the fact that the reconstruction loss during training tends to saturate even when the weights are sufficiently high indicates that even well-engineered architectures such as StyleGAN and BigGAN lack representative power to provide sufficient data coverage.

Strong inductive biases in the generative model have the potential to improve the quality of the inference module. For instance, GIF [18] and hologan [29] among others introduce strong inductive bias from the underlying 3D geometry and lighting properties of a 2D image. Hence, an inverse module of these generative mechanism has the potential to outperform their counterparts, which are trained fully supervised on the labelled training data alone at estimating 3D face parameters from 2D images.

As was shown by the success of RAEs [17], there is often a mismatch between the induced posterior and the prior of generative models, which can be removed by an ex-post density estimator. InvGAN is also aminable to ex-post density estimation. When applied to the tiled latent codes, it estimates a joint density of the tiles for unseen data. This would recover a generative model without going through the unstable GAN training.

We have shown that our method scales to large datasets such as ImageNet, CelebA, and FFHQ. A future work that is able to improve upon reconstruction fidelity, would be able to explore adversarial robustness by extending [19] to larger datasets.

6 Conclusion

We presented InvGAN, an inference framework for the latent generative parameters used by a GAN generator. InvGAN enjoys several advantages compared to state-of-the-art inversion mechanisms. Since InvGAN is a model-based approach, it enjoys computational efficiency. This enables our mechanism to reconstruct images that are larger than the training images by tiling, with no additional merging step. Furthermore, the inversion mechanism is integrated into the training phase of the generator, this encourages the generator to cover all modes. We further demonstrated that the inferred latent code for a given image is semantically meaningful, i.e., it falls inside the structured part of the latent space learned by the generator.

References

1. seamless color mapping for 3d reconstruction with consumer-grade scanning devices
2. Abdal, R., Qin, Y., Wonka, P.: Image2stylegan: How to embed images into the stylegan latent space? In: Proceedings of the IEEE international conference on computer vision. pp. 4432–4441 (2019)
3. Abdal, R., Qin, Y., Wonka, P.: Image2stylegan: How to embed images into the stylegan latent space? CoRR **abs/1904.03189** (2019), http://arxiv.org/abs/1904.03189
4. Abdal, R., Qin, Y., Wonka, P.: Image2stylegan++: How to edit the embedded images? In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
5. Alaluf, Y., Patashnik, O., Cohen-Or, D.: Restyle: A residual-based stylegan encoder via iterative refinement (2021)
6. Alaluf, Y., Tov, O., Mokady, R., Gal, R., Bermano, A.H.: Hyperstyle: Stylegan inversion with hypernetworks for real image editing (2021). https://doi.org/10.48550/ARXIV.2111.15666, https://arxiv.org/abs/2111.15666
7. Balakrishnan, G., Xiong, Y., Xia, W., Perona, P.: Towards causal benchmarking of bias in face analysis algorithms. In: European Conference on Computer Vision. pp. 547–563. Springer (2020)
8. Bau, D., Strobelt, H., Peebles, W., Zhou, B., Zhu, J.Y., Torralba, A., et al.: Semantic photo manipulation with a generative image prior. arXiv preprint arXiv:2005.07727 (2020)
9. Bousquet, O., Gelly, S., Tolstikhin, I., Simon-Gabriel, C.J., Schoelkopf, B.: From optimal transport to generative modeling: the vegan cookbook. arXiv preprint arXiv:1705.07642 (2017)
10. Brock, A., Donahue, J., Simonyan, K.: Large scale GAN training for high fidelity natural image synthesis. CoRR **abs/1809.11096** (2018), http://arxiv.org/abs/1809.11096
11. Chen, M., Radford, A., Child, R., Wu, J., Jun, H., Luan, D., Sutskever, I.: Generative pretraining from pixels. In: III, H.D., Singh, A. (eds.) Proceedings of the 37th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 119, pp. 1691–1703. PMLR (13–18 Jul 2020), https://proceedings.mlr.press/v119/chen20s.html
12. Cheng, Y., Gan, Z., Li, Y., Liu, J., Gao, J.: Sequential attention gan for interactive image editing. arXiv preprint arXiv:1812.08352 (2020)
13. Child, R.: Very deep {vae}s generalize autoregressive models and can outperform them on images. In: International Conference on Learning Representations (2021), https://openreview.net/forum?id=RLRXCV6DbEJ
14. Donahue, J., Krähenbühl, P., Darrell, T.: Adversarial feature learning. arXiv preprint arXiv:1605.09782 (2016)
15. Donahue, J., Simonyan, K.: Large scale adversarial representation learning. CoRR **abs/1907.02544** (2019), http://arxiv.org/abs/1907.02544
16. Dumoulin, V., Belghazi, I., Poole, B., Mastropietro, O., Lamb, A., Arjovsky, M., Courville, A.: Adversarially learned inference. arXiv preprint arXiv:1606.00704 (2016)
17. Ghosh*, P., Sajjadi*, M.S.M., Vergari, A., Black, M.J., Schölkopf, B.: From variational to deterministic autoencoders. In: 8th International Conference on Learning Representations (ICLR) (Apr 2020), https://openreview.net/forum?id=S1g7tpEYDS, *equal contribution

18. Ghosh, P., Gupta, P.S., Uziel, R., Ranjan, A., Black, M.J., Bolkart, T.: GIF: Generative interpretable faces. In: International Conference on 3D Vision (3DV) (2020), http://gif.is.tue.mpg.de/
19. Ghosh, P., Losalka, A., Black, M.J.: Resisting adversarial attacks using gaussian mixture variational autoencoders. Proceedings of the AAAI Conference on Artificial Intelligence **33**(01), 541–548 (Jul 2019). https://doi.org/10.1609/aaai.v33i01.3301541, https://ojs.aaai.org/index.php/AAAI/article/view/3828
20. Guan, S., Tai, Y., Ni, B., Zhu, F., Huang, F., Yang, X.: Collaborative learning for faster stylegan embedding. CoRR **abs/2007.01758** (2020), https://arxiv.org/abs/2007.01758
21. Johnson, J., Alahi, A., Li, F.: Perceptual losses for real-time style transfer and super-resolution. CoRR **abs/1603.08155** (2016), http://arxiv.org/abs/1603.08155
22. Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of gans for improved quality, stability, and variation. arXiv preprint arXiv:1710.10196 (2017)
23. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 4401–4410 (2019)
24. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of StyleGAN. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8110–8119 (2020)
25. Lin, C.H., Chang, C., Chen, Y., Juan, D., Wei, W., Chen, H.: COCO-GAN: generation by parts via conditional coordinating. CoRR **abs/1904.00284** (2019), http://arxiv.org/abs/1904.00284
26. Lipton, Z.C., Tripathi, S.: Precise recovery of latent vectors from generative adversarial networks. arXiv preprint arXiv:1702.04782 (2017)
27. Locatello, F., Bauer, S., Lucic, M., Raetsch, G., Gelly, S., Schölkopf, B., Bachem, O.: Challenging common assumptions in the unsupervised learning of disentangled representations. In: international conference on machine learning. pp. 4114–4124. PMLR (2019)
28. Marriott, R.T., Madiouni, S., Romdhani, S., Gentric, S., Chen, L.: An assessment of gans for identity-related applications. In: 2020 IEEE International Joint Conference on Biometrics (IJCB). pp. 1–10 (2020). https://doi.org/10.1109/IJCB48548.2020.9304879
29. Nguyen-Phuoc, T., Li, C., Theis, L., Richardt, C., Yang, Y.L.: Hologan: Unsupervised learning of 3d representations from natural images. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV). pp. 7588–7597 (2019)
30. Perarnau, G., Van De Weijer, J., Raducanu, B., Álvarez, J.M.: Invertible conditional gans for image editing. arXiv preprint arXiv:1611.06355 (2016)
31. Pidhorskyi, S., Adjeroh, D.A., Doretto, G.: Adversarial latent autoencoders. CoRR **abs/2004.04467** (2020), https://arxiv.org/abs/2004.04467
32. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434 (2015)
33. Ramaswamy, V.V., Kim, S.S., Russakovsky, O.: Fair attribute classification through latent space de-biasing. arXiv preprint arXiv:2012.01469 (2020)
34. Razavi, A., van den Oord, A., Vinyals, O.: Generating diverse high-fidelity images with vq-vae-2. In: Advances in Neural Information Processing Systems. pp. 14866–14876 (2019)

35. Richardson, E., Alaluf, Y., Patashnik, O., Nitzan, Y., Azar, Y., Shapiro, S., Cohen-Or, D.: Encoding in style: a stylegan encoder for image-to-image translation. CoRR **abs/2008.00951** (2020), https://arxiv.org/abs/2008.00951
36. Salimans, T., Goodfellow, I.J., Zaremba, W., Cheung, V., Radford, A., Chen, X.: Improved techniques for training gans. CoRR **abs/1606.03498** (2016), http://arxiv.org/abs/1606.03498
37. dos Santos Tanaka, F.H.K., Aranha, C.: Data augmentation using gans. CoRR **abs/1904.09135** (2019), http://arxiv.org/abs/1904.09135
38. Sattigeri, P., Hoffman, S.C., Chenthamarakshan, V., Varshney, K.R.: Fairness gan. arXiv preprint arXiv:1805.09910 (2018)
39. Sharmanska, V., Hendricks, L.A., Darrell, T., Quadrianto, N.: Contrastive examples for addressing the tyranny of the majority. arXiv preprint arXiv:2004.06524 (2020)
40. Soomro, K., Zamir, A.R., Shah, M.: UCF101: A dataset of 101 human actions classes from videos in the wild. CoRR **abs/1212.0402** (2012), http://arxiv.org/abs/1212.0402
41. Tewari, A., Elgharib, M., Bharaj, G., Bernard, F., Seidel, H.P., Pérez, P., Zöllhofer, M., Theobalt, C.: Stylerig: Rigging stylegan for 3d control over portrait images, cvpr 2020. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). IEEE (june 2020)
42. Tolstikhin, I., Bousquet, O., Gelly, S., Schoelkopf, B.: Wasserstein auto-encoders. In: International Conference on Learning Representations (2018), https://openreview.net/forum?id=HkL7n1-0b
43. Voynov, A., Babenko, A.: Unsupervised discovery of interpretable directions in the gan latent space. arXiv preprint arXiv:2002.03754 (2020)
44. Wei, T., Chen, D., Zhou, W., Liao, J., Zhang, W., Yuan, L., Hua, G., Yu, N.: A simple baseline for stylegan inversion (2021)
45. Wulff, J., Torralba, A.: Improving inversion and generation diversity in stylegan using a gaussianized latent space. arXiv preprint arXiv:2009.06529 (2020)
46. Xia, W., Zhang, Y., Yang, Y., Xue, J.H., Zhou, B., Yang, M.H.: Gan inversion: A survey. arXiv preprint arXiv:2101.05278 (2021)
47. Xia, W., Zhang, Y., Yang, Y., Xue, J., Zhou, B., Yang, M.: GAN inversion: A survey. CoRR **abs/2101.05278** (2021), https://arxiv.org/abs/2101.05278
48. Xu, H., Ma, Y., Liu, H., Deb, D., Liu, H., Tang, J., Jain, A.K.: Adversarial attacks and defenses in images, graphs and text: A review (2019). https://doi.org/10.48550/ARXIV.1909.08072, https://arxiv.org/abs/1909.08072
49. Xu, Y., Shen, Y., Zhu, J., Yang, C., Zhou, B.: Generative hierarchical features from synthesizing images. In: CVPR (2021)
50. Yu, J., Li, X., Koh, J.Y., Zhang, H., Pang, R., Qin, J., Ku, A., Xu, Y., Baldridge, J., Wu, Y.: Vector-quantized Image Modeling with Improved VQGAN. In: International Conference on Learning Representations (ICLR) (2022)
51. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: CVPR (2018)
52. Zhu, J., Shen, Y., Zhao, D., Zhou, B.: In-domain gan inversion for real image editing. In: Proceedings of European Conference on Computer Vision (ECCV) (2020)
53. Zhu, J., Zhao, D., Zhang, B.: LIA: latently invertible autoencoder with adversarial learning. CoRR **abs/1906.08090** (2019), http://arxiv.org/abs/1906.08090
54. Zhu, J.Y., Krähenbühl, P., Shechtman, E., Efros, A.A.: Generative visual manipulation on the natural image manifold. In: European conference on computer vision. pp. 597–613. Springer (2016)

55. Zietlow, D., Lohaus, M., Balakrishnan, G., Kleindessner, M., Locatello, F., Schölkopf, B., Russell, C.: Leveling down in computer vision: Pareto inefficiencies in fair deep classifiers (2022). https://doi.org/10.48550/ARXIV.2203.04913, https://arxiv.org/abs/2203.04913