GAN Inversion: A Survey

Weihao Xia, Yulun Zhang, Yujiu Yang*, Jing-Hao Xue, Bolei Zhou*, Ming-Hsuan Yang*

Abstract—GAN inversion aims to invert a given image back into the latent space of a pretrained GAN model so that the image can be faithfully reconstructed from the inverted code by the generator. As an emerging technique to bridge the real and fake image domains, GAN inversion plays an essential role in enabling pretrained GAN models, such as StyleGAN and BigGAN, for use for real image editing applications. Moreover, GAN inversion also provides insights into the interpretation of the latent space of GANs and how realistic images can be generated. In this paper, we provide an overview of GAN inversion with a focus on recent algorithms and applications. We cover important techniques of GAN inversion and their applications in image restoration and image manipulation. We further elaborate on some trends and challenges for future research. A curated list of GAN inversion methods, datasets, and other related information can be found at `github.com/weihaox/awesome-gan-inversion`.

Index Terms—Generative Adversarial Networks, Interpretable Machine Learning, Image Reconstruction, Image Manipulation

1 Introduction

The generative adversarial network (GAN) framework is a deep learning architecture that estimates how data points are generated in a probabilistic framework [1], [2]. It consists of two interacting neural networks: a generator, G, and a discriminator, D, which are trained jointly through an adversarial process. The objective of G is to synthesize fake data that resemble real data, while the objective of D is to distinguish between real and fake data. Through an adversarial training process, the generator G can generate fake data that match the real data distribution. In recent years, GANs have been applied to numerous tasks ranging from image translation [3], [4], [5] and image manipulation [6], [7], [8] to image restoration [9], [10], [11], [12], [13].

Numerous GAN models, *e.g.*, PGGAN [14], BigGAN [15] and StyleGAN [16], [17], have been developed to synthesize images with high quality and diversity from random noise input. Recent studies have shown that GANs effectively encode rich semantic information in the intermediate features [18] and latent space [19], [20], [21] as a result of image generation. These methods can synthesize images with various attributes, such as aging, expression, and light direction, by varying the latent code. However, such manipulations in the latent space are only applicable to images generated from GANs rather than to any given real images due to the lack of inference functionality or the encoder in GANs.

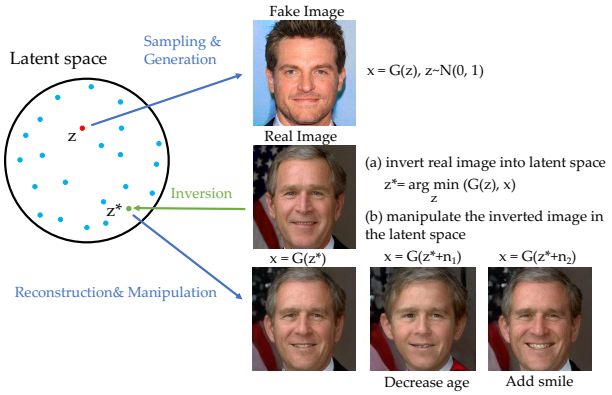

Fig. 1. **Illustration of GAN inversion.** Different from the conventional sampling and generation process using trained generator G, GAN inversion maps a given real image x to the latent space and obtains the latent code $\mathbf{z}^*$. The reconstructed image x^* is then obtained by $x^* = G(\mathbf{z}^*)$. By varying the latent code $\mathbf{z}^*$ in different interpretable directions *e.g.*, $\mathbf{z}^* + \mathbf{n_1}$ and $\mathbf{z}^* + \mathbf{n_2}$ where $\mathbf{n_1}$ and $\mathbf{n_2}$ model the age and smile in the latent space respectively, we can edit the corresponding attribute of the real image. The reconstructed results are from [22].

In contrast, GAN inversion aims to invert a given image back into the latent space of a pretrained GAN model. The image can then be faithfully reconstructed from the inverted code by the generator. Since GAN inversion plays an essential role in bridging real and fake image domains, significant advances have been made [17], [20], [21], [23], [24], [25], [26], [27], [28]. GAN inversion enables the controllable directions found in latent spaces of the existing trained GANs to be applicable to real image editing, without requiring ad-hoc supervision or expensive optimization. As shown in Figure 1, after the real image is inverted into the latent space, we can vary its code along one specific direction to edit the corresponding attribute of the image. As a rapidly growing field that combines the generative adversarial network with interpretable machine learning techniques, GAN inversion not only provides an alternative flexible image editing framework but also helps reveal the

**Corresponding authors*
W. Xia and Y. Yang are with Tsinghua Shenzhen International Graduate School, Tsinghua University, China. Email: weihaox@outlook.com, yang.yujiu@sz.tsinghua.edu.cn
Y. Zhang is with Department of Electrical and Computer Engineering, Northeastern University, USA. Email: yulun100@gmail.com
J.-H. Xue is with the Department of Statistical Science, University College London, UK. Email: jinghao.xue@ucl.ac.uk
B. Zhou is with Department of Information Engineering, The Chinese University of Hong Kong, Shatin, Hong Kong SAR, China. Email: bzhou@ie.cuhk.edu.hk
M.-H. Yang is with Electrical Engineering and Computer Science, University of California at Merced, USA. Email: mhyang@ucmerced.edu

inner mechanism of deep generative models.

In this paper, we present a comprehensive survey of GAN inversion methods with an emphasis on algorithms and applications. To the best of our knowledge, this work is the first survey on the rapidly growing GAN inversion with the following contributions. First, we provide a comprehensive and systematic review and analysis of all aspects of GAN inversion both hierarchically and structurally. Second, we provide a comparative summary of the properties and performance for GAN inversion methods. Third, we discuss the challenges and open issues and identify the trends for future research.

The rest of this article is organized as follows. We give a unified mathematical formulation for the problem of GAN inversion in Section 2. The solution obtained, *i.e.* a latent code for a given image, should have two properties, which are also the two goals of GAN inversion. That is, the latent code should not only 1) reconstruct the input image faithfully and photorealistically but also 2) facilitate downstream tasks. Then, in subsequent sections, we take the two goals as the mainline position and introduce what efforts have been made by the reviewed methods and applications to meet these two purposes. The pretrained GAN model, $G(\mathbf{z})$, has many choices, which we introduce in Section 3.1. To evaluate the solution, we need to consider two important aspects for GAN inversion: how photorealistic (image quality) and faithful (inversion accuracy) the generated image is, which we introduce in Section 3.2. The first goal depends on how the formulation is solved. It is usually a nonconvex problem due to the nonconvexity of $G(\mathbf{z})$, for which finding accurate solutions is difficult. The second goal is primarily decided by which latent space to use. Section 4.1 introduces, analyses, and compares the characteristics of different latent spaces. In Section 4.2, 4.3, and 4.4, we introduce how existing methods have attempted to provide solutions and discuss some important characteristics of these GAN inversion methods. Applications and future directions of GAN inversion are introduced in Sections 5 and 6, respectively.

2 Problem Definition and Overview of GAN Inversion

It is well known that GANs [1], [14], [16] can generate high-resolution and photorealistic 'fake' images, but a lesser-known aspect is how to apply these GANs to applications of 'real' images. One possible solution, *i.e.* GAN inversion, is to obtain the 'real' images' latent codes and perform some subsequent image processing tasks by manipulating the latent codes in the latent space. As indicated, GAN inversion attempts to find a latent code that can achieve two goals for a given image: 1) reconstruct the input image faithfully and photorealistically and 2) facilitate downstream tasks.

We first define the problem of GAN inversion under a unified mathematical formulation. The generator of an unconditional GAN learns the mapping $G : \mathcal{Z} \rightarrow \mathcal{X}$. When $\mathbf{z}_1, \mathbf{z}_2 \in \mathcal{Z}$ are close in the $\mathcal{Z}$ space, the corresponding images $x_1, x_2 \in \mathcal{X}$ are visually similar. GAN inversion maps data x back to latent representation $\mathbf{z}^*$ or, equivalently, finds an image x^* that can be entirely synthesized by the well-trained generator G and remain close to the real image x. Formally, denoting the signal to be inverted as $x \in \mathbb{R}^n$, the well-trained generative model as $G : \mathbb{R}^{n_0} \rightarrow \mathbb{R}^n$, and the latent vector as $\mathbf{z} \in \mathbb{R}^{n_0}$, we study the following inversion problem:

$$\mathbf{z}^* = \arg\min_{\mathbf{z}} \ell(G(\mathbf{z}), x), \tag{1}$$

where $\ell(\cdot)$ is a distance metric in the image or feature space, and G is assumed to be a feed-forward neural network. Typically, $\ell(\cdot)$ can be based on ℓ_1, ℓ_2, perceptual [29] or LPIPS [30] metrics. Some other constraints on latent codes [22] or face identity [31] could also be included in practice. With the obtained $\mathbf{z}^*$, we can obtain the original and manipulated images.

The second goal of facilitating downstream tasks is primarily decided by which latent space to use (see Section 4.1). The first goal, however, depends on how to solve Equation (1), which is usually a nonconvex problem due to the nonconvexity of $G(\mathbf{z})$ and is not easily amenable to finding accurate solutions. Numerous methods [23], [24], [31] have been developed to solve Equation (1) with formulation based on learning, optimization, or both. Generally, learning-based GAN inversion methods cannot faithfully reconstruct the image content. Taking facial images as an example, they have been known to sometimes fail in preserving identity as well as some other details [22], [31]. While optimization-based techniques have achieved superior image reconstruction, their inevitable drawback is the significantly higher computational cost [24], [25]. Thus, recent improvements of learning-based GAN inversion methods mainly focus on how to faithfully reconstruct images, *e.g.* integrating an additional facial identity loss during training [31], [32] or proposing an iterative feedback mechanism [33]. Recent improvements of optimization-based methods place more emphasis on how to find the desired latent code more quickly and propose several strategies on initialization [24], [25] and optimizers [23], [27]. Reconstruction quality and inference time cannot be simultaneously achieved for existing inversion approaches, resulting in a 'quality-time tradeoff'. Although some hybrid approaches are additionally proposed to balance this tradeoff, it remains a challenge to quickly find the accurate latent code.

Similar to GAN inversion, some tasks also aim to learn the inverse mapping of GAN models. Some methods [34], [35], [36] use additive encoder networks to learn the inverse mapping of GANs, but their goals are to jointly train the encoder with both the generator and the discriminator, instead of using *a trained GAN model*. Some other methods, *e.g.* PULSE [37], ILO [38], or PICGM [39], also rely on a pretrained generator to solve inverse problems such as inpainting, superresolution, or denoising. They design different optimization mechanisms to search for latent codes that satisfy the given degraded observations. Since they aim to search for accurate and reliable estimation (*e.g.* denoised image) from a degraded observation (*e.g.* noisy image) instead of *faithful reconstruction of the given image*, we do not categorize them as GAN inversion methods in our survey paper. But it would be beneficial to pay attention to this direction as they share the same idea of finding desired latent code in the latent space of pretrained GAN models.

TABLE 1
Characteristics of GAN inversion methods. 'Type' includes Learning-based (L.), Optimization-based (O.), and Hybrid (H.) GAN inversion. S.-A., L.-W., and S.-R denote Semantic awareness, Layerwise, and Supported Resolution, respectively. GAN model and Dataset indicate which Pretrained Models are trained on which Dataset that a method is inverting, which can be found in Section 3.1.

Method	Publication	Type	S.-A.	L.-W.	S.-R	Space	GAN Model	Dataset
Zhu *et al.* [23]	2016, ECCV	H.	✗	✗	64	$\mathcal{Z}$	[40]	[41], [42], [43]
Creswell *et al.* [44]	2018, TNNLS	O.	✗	✗	128	$\mathcal{Z}$	[40], [45]	[41], [46]
GAN Paint [47]	2019, TOG	H.	✓	✓	256	$\mathcal{Z}$	[14]	[43]
GANSeeing [26]	2019, ICCV	H.	✓	✓	256	$\mathcal{Z}$, $\mathcal{W}$	[14], [16], [45]	[43]
Raj *et al.* [48]	2019, ICCV	O.	✗	✗	64	$\mathcal{Z}$	[40], [49]	[43], [46], [50]
Image2StyleGAN [24]	2019, ICCV	O.	✓	✗	1024	$\mathcal{W}^+$	[16]	[16]
Image2StyleGAN++ [25]	2020, CVPR	O.	✓	✗	1024	$\mathcal{W}^+$	[14], [16]	[14], [16]
mGANPrior [51]	2020, CVPR	O.	✓	✓	256	$\mathcal{Z}$	[14], [16]	[14], [16], [43]
Editing in Style [52]	2020, CVPR	O.	✓	✗	1024	$\mathcal{W}$	[14], [16], [17]	[16], [43]
StyleRig [53]	2020, CVPR	L.	✓	✗	1024	$\mathcal{W}^+$	[16]	[16]
YLG [54]	2020, CVPR	O.	✗	✗	128	$\mathcal{Z}$	[49]	[55]
DGP [28]	2020, ECCV	O.	✗	✓	256	$\mathcal{Z}$	[15]	[42], [55]
Huh *et al.* [27]	2020, ECCV	O.	✓	✗	1024	$\mathcal{Z}$	[15], [17]	[16], [43], [55]
IDInvert [22]	2020, ECCV	H.	✓	✗	256	$\mathcal{W}^+$	[16]	[16], [43]
StyleGAN2 Distillation [56]	2020, ECCV	O.	✓	✗	1024	$\mathcal{W}^+$	[17]	[16]
MimicGAN [57]	2020, IJCV	O.	✗	✗	64	$\mathcal{Z}$	[46]	[40]
PIE [58]	2020, TOG	O.	✓	✗	1024	$\mathcal{W}^+$	[16]	[46]
Nitzan *et al.* [59]	2020, TOG	L.	✓	✗	1024	$\mathcal{W}$	[16]	[14], [16]
Aberdam *et al.* [60]	2020, arxiv	O.	✗	✓	-	$\mathcal{Z}$	a two-layer model	[50]
StyleGAN-Encoder [61]	2020, arxiv	L.	✓	✗	256	$\mathcal{W}^+$	[16]	[14], [16], [62]
Style Intervention [63]	2020, arxiv	O.	✓	✗	1024	$\mathcal{S}$	[17]	[63]
Cherepkov *et al.* [64]	2021, CVPR	O.	✓	✗	256	$\mathcal{W}^+$	[17]	[16], [43]
Zhuang *et al.* [65]	2021, ICLR	O.	✓	✗	1024	$\mathcal{Z}$	[14], [17]	[14], [16]
Chai *et al.* [66]	2021, ICLR	L.	✓	✗	1024	$\mathcal{Z}$, $\mathcal{W}^+$	[14], [17]	[14], [16], [43]
pSp [31]	2021, CVPR	L.	✓	✓	1024	$\mathcal{W}^+$	[17]	[14]
StyleSpace [67]	2021, CVPR	O.	✓	✗	1024	$\mathcal{S}$	[17]	[16], [43]
GH-Feat [68]	2021, CVPR	L.	✓	✗	256	$\mathcal{S}$	[16]	[16], [43], [50]
Hijack-GAN [69]	2021, CVPR	O.	✓	✗	1024	$\mathcal{Z}$	[14], [16]	[14]
GANEnsembling [70]	2021, CVPR	H.	✓	✗	1024	$\mathcal{W}^+$	[17]	[16], [43]
StyleFlow [71]	2021, TOG	O.	✓	✗	1024	$\mathcal{W}^+$	[16], [17]	[16], [43]
SAM [72]	2021, TOG	L.	✓	✗	1024	$\mathcal{W}^+$	[16]	[14], [16]
e4e [73]	2021, TOG	L.	✓	✓	1024	$\mathcal{W}^+$	[17]	[14], [16], [43]
Xu *et al.* [74]	2021, ICCV	O.	✓	✗	1024	$\mathcal{W}^+$	[16]	[16], [75]
Zhu *et al.* [76]	2021, arxiv	O.	✓	✗	1024	$\mathcal{P}$	[16], [17]	[16]
ReStyle [33]	2021, arxiv	L.	✓	✗	1024	$\mathcal{W}^+$	[17]	[14], [16], [43], [77]
Wei *et al.* [32]	2021, arxiv	L.	✓	✗	1024	$\mathcal{W}^+$	[17]	[14], [16]

3 PRELIMINARIES

3.1 Trained GAN Models and Datasets

Deep generative models such as GANs [1] have been used to model natural image distributions and synthesize photorealistic images. Recent advances in GANs, such as DCGAN [40], WGAN [45], PGGAN [14], BigGAN [15], StyleGAN [16], StyleGAN2 [17], and StyleGAN2-Ada [78], have developed better architectures, losses and training schemes. These models are trained on diverse datasets, including faces (CelebA-HQ [14], FFHQ [16], [17], AnimeFaces [79] and AnimalFace [80]), scenes (LSUN [43]), and objects (LSUN [43] and ImageNet [55]).

3.1.1 GAN Models

DCGAN [40] uses convolutions in the discriminator and fractional-strided convolutions in the generator.

WGAN [45] minimizes the Wasserstein distance between the generated and real data distributions, which offers more model stability and makes the training process easier.

BigGAN [15] generates high-resolution and high-quality images, with modifications for scaling up, architectural changes and orthogonal regularization to improve the scalability, robustness and stability of large-scale GANs. BigGAN can be trained on ImageNet at 256×256 and 512×512.

PGGAN [14], also denoted as ProGAN or progressive GAN, uses a growing strategy for the training process. The key idea is to start with a low resolution for both the generator and the discriminator and then add new layers that model increasingly fine-grained details as the training progresses. This approach improves both the training speed and the stabilization, thereby facilitating image synthesis at higher resolution, *e.g.*, CelebA images at 1024×1024 pixels.

StyleGAN [16] implicitly learns hierarchical latent styles for image generation. This model manipulates the per-channel mean and variance to control the style of an image [81] effectively. The StyleGAN generator takes per-block incorporation of style vectors (defined by a mapping network) and stochastic variation (provided by the noise layers) as inputs, instead of samples from the latent space, to generate a synthetic image. This offers control over the style of generated images at different levels of detail. The StyleGAN2 model [17] further improves the image quality by proposing weight demodulation, path length regularization, generator redesign, and removal of progressive growing. The StyleGAN2-Ada [78] proposes an adaptive discriminator augmentation mechanism to stabilize training with limited data. A recent method [82] observes an 'texture sticking' problem (aliasing) in GANs and proposes the Alias-Free GAN by considering aliasing effect in the continuous domain and appropriately low-pass filtering the

results, which is better suited for video and animation. The StyleGAN-based architectures have been used in numerous applications [83], [84], [85]. For style-based generators, the image size R is determined by its number of layers L: $L = 2\log_2 \mathrm{R} - 2$; it also has a maximum resolution of 1024 with 18 layers.

3.1.2 Datasets

ImageNet [55] is a large-scale hand-annotated dataset for visual object recognition research and contains more than 14 million images with more than 20,000 categories.

CelebA [46] is a large-scale face attribute dataset consisting of 200K celebrity images with 40 attribute annotations each. CelebA, together with its succeeding CelebA-HQ [14], CelebAMask-HQ [86], and CelebA-Spoof [87], are widely used in face image generation and manipulation.

Flickr-Faces-HQ (FFHQ) [16], [17] is a high-quality image dataset of human faces crawled from Flickr, which consists of 70,000 high-quality human face images of 1024×1024 pixels and contains considerable variation in terms of age, ethnicity, and image background.

LSUN [43] contains approximately one million labeled images for each of 10 scene categories (*e.g.*, bedroom, church, or tower) and 20 object classes (*e.g.*, bird, cat, or bus). The church and bedroom scene images and car and bird object images are commonly used in the GAN inversion methods.

In addition, some GAN inversion studies also use other datasets [50], [62], [77], [88], [89], [90] in their experiments, such as **DeepFashion** [91], [92], [93], **AnimeFaces** [79], and **StreetScapes** [94].

3.2 Evaluation Metrics

For evaluation, there are two important aspects for GAN inversion: how photorealistic (image quality) and faithful (inversion accuracy) the generated image is. IS, FID, and LPIPS are widely used measurements to assess the quality of GAN-generated images; recent studies have also used SWD. IS and FID are metrics for image diversity, while LPIPS is a metric for similarity. For inversion accuracy, most methods use the reconstruction distance, *e.g.* PSNR or SSIM. Some other methods [59] use cosine or Euclidean distance to evaluate different attributes between the input and output, while other approaches [95] use classification accuracy for assessment.

3.2.1 Image Quality

The mean opinion score (MOS) and **difference mean opinion score** (DMOS) have been used for subjective image quality assessment, where human raters are asked to assign perceptual quality scores to images. Typically, the scores range from 1 (bad) to 5 (good), and the final MOS is calculated as the arithmetic mean over all ratings. However, there are drawbacks with this metric, *e.g.*, nonlinearly perceived scale, bias and variance in rating criteria.

The inception score (IS) [96] is a widely used metric to measure the quality and diversity of images generated from GAN models. It calculates the statistics of the Inception-v3 Network [97] pretrained on ImageNet [98] when applied to generated images:

$$\mathrm{IS} = \exp\left(\mathbb{E}_{x\sim p_g} D_{KL}(p(y|x) \parallel p(y))\right), \tag{2}$$

where $x \sim p_g$ indicates that x is an image sampled from p_g, $D_{KL}(p\|q)$ is the KL-divergence between the distributions p and q, $p(y|x)$ is the conditional class distribution, and $p(y) = \int_x p(y|x)p_g(x)$ is the marginal class distribution.

The Fréchet inception distance [99] (FID) is defined using the Fréchet distance between two multivariate Gaussians:

$$\mathrm{FID} = \|\mu_r - \mu_g\|^2 + \mathrm{Tr}(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{\frac{1}{2}}), \tag{3}$$

where $X_r \sim \mathcal{N}(\mu_r, \Sigma_r)$ and $X_g \sim \mathcal{N}(\mu_g, \Sigma_g)$ are the 2048-dimensional activations of the Inception-v3 [97] pool3 layer for real and generated samples, respectively. The lowest FID indicates the most perceptual results.

The Fréchet segmentation distance (FSD) [26] is an interpretable counterpart to the FID metric:

$$\mathrm{FSD} = \|\mu_g - \mu_t\|^2 + \mathrm{Tr}(\Sigma_g + \Sigma_t - 2(\Sigma_g\Sigma_t)^{\frac{1}{2}}), \tag{4}$$

where μ_t is the mean pixel count for each object class over a sample of training images, and Σ_t is the covariance.

Sliced Wasserstein discrepancy (SWD) [100] is designed to capture the dissimilarity between the outputs of task-specific classifiers and can be obtained by computing 1D Wasserstein distances of the projected point clouds:

$$\tilde{W}(X,Y)^2 = \int_{\theta\in\Omega} W\left(X_\theta, Y_\theta\right)^2 \, \mathrm{d}\theta \tag{5}$$

where $X_\theta = \{\langle X_i, \theta\rangle\}_{i\in I} \subset \mathbb{R}$, and $\Omega = \left\{\theta \in \mathbb{R}^d \backslash \|\theta\| = 1\right\}$ is the unit sphere. It provides geometrically meaningful guidance to detect target samples that are far from the support of the source and enables efficient distribution alignment in an end-to-end trainable fashion.

Learned perceptual image patch similarity (LPIPS) [30] can measure image perceptual quality while reducing manual intervention. It is computed between two patches using the cosine distance in the channel dimension and the average across spatial dimensions and the given convolutional layers of different networks. To obtain the distance between reference and distorted patches x, x_0 with network $\mathcal{F}$, it first computes deep embeddings for layer l, denoted as $\hat{y}^l, \hat{y}^l_0 \in \mathbb{R}^{H_l\times W_l\times C_l}$; normalizes the activations in the channel dimension; scales each channel by vector $w^l \in \mathbb{R}^{C_l}$; and measures the ℓ_2 distance (using $w_l = 1, \forall l$, which is equivalent to computing the cosine distance):

$$d(x, x_0) = \sum_l \frac{1}{H_l W_l} \sum_{h,w} ||w_l \odot (\hat{y}^l_{hw} - \hat{y}^l_{0hw})||_2^2. \tag{6}$$

3.2.2 Inversion Accuracy

Classification Accuracy. Voynov *et al.* [95] propose **reconstructor classification accuracy** (RCA) to measure model interpretability by predicting the direction in the latent space that a given image transformation is generated. The reconstructor's classification solves a multiclass classification problem, and high RCA values imply that directions are easy to distinguish from each other; *i.e.*, the corresponding image transformations do not "interfere" or influence

different factors of variations. Abdal *et al.* [71] use **face identity** to evaluate the quality of the edits and quantify the identity preserving property of the edits. A face classifier model [101] is used to obtain the embeddings of the images, which can be compared (before and after the edits), *i.e.*, i_1 and i_2, and then, the Euclidean distance and the cosine similarity are calculated between the embeddings.

Reconstruction Distances. To evaluate the reconstruction, the most widely used metrics are **peak signal-to-noise ratio** (PSNR) and **structural similarity** (SSIM) [102]. The PSNR is one of the most widely used criteria to measure the quality of reconstruction. The PSNR between the ground truth image I and the reconstruction $\hat{I}$ is defined by the maximum possible pixel value of the image (denoted as L) and the mean squared error (MSE) between images:

$$\text{PSNR} = 10 \cdot \log_{10}\left(\frac{L^2}{\frac{1}{N}\sum_{i=1}^{N}(I(i) - \hat{I}(i))^2}\right), \tag{7}$$

where L equals $2^n - 1$ if represented using linear pulse-code modulation with n bits, *e.g.*, 255 in general cases using 8-bit representations. SSIM measures the structural similarity between images based on independent comparisons in terms of luminance, contrast, and structures. For two images I and $\hat{I}$ with N pixels, the SSIM is given by

$$\text{SSIM}(I, \hat{I}) = [\mathcal{C}_l(I, \hat{I})]^{\alpha}[\mathcal{C}_c(I, \hat{I})]^{\beta}[\mathcal{C}_s(I, \hat{I})]^{\gamma}, \tag{8}$$

where α, β, and γ are the control parameters for adjusting the relative importance. The details of these terms can be found in [102].

In addition, some inversion methods also introduce other metrics to measure the reconstruction distance. Nitzan *et al.* [59] utilize the cosine similarity metric to compare the accuracy of expression preservation, which is calculated as the Euclidean distance between 2D landmarks of I_{attr} and I_{out} [103]. In contrast, the pose preservation is calculated as the Euclidean distance between Euler angles of I_{attr} and I_{out}. Abdal *et al.* [71] develop the **edit consistency score** to measure the consistency across edited images based on the assumption that different permutations of edits should lead to the same attributes when classified with an attribute classifier. For instance, the pose attribute obtained after editing expression and pose and that obtained after editing the same pose and lighting are expected to be the same, as constrained by the proposed score $|\mathcal{A}_p(E_p(E_e(I)) - \mathcal{A}_p(E_l(E_p(I))|$, where E_x denotes a conditional edit along attribute specification x, and $\mathcal{A}_p$ denotes the pose attribute vector regressed by the attribute classifier.

4 GAN Inversion Methods

4.1 Which Space to Embed - From $\mathcal{Z}$ Space to $\mathcal{P}$ Space

Regardless of the GAN inversion method used, one important functionality that they all have is the choice of a latent space for images to embed. A good latent space should have a simple structure and be easy to embed. The latent code in such a latent space should have the following two properties: it should reconstruct the input image faithfully and photorealistically, and it should facilitate downstream tasks (*e.g.* image editing). In this section, we introduce some

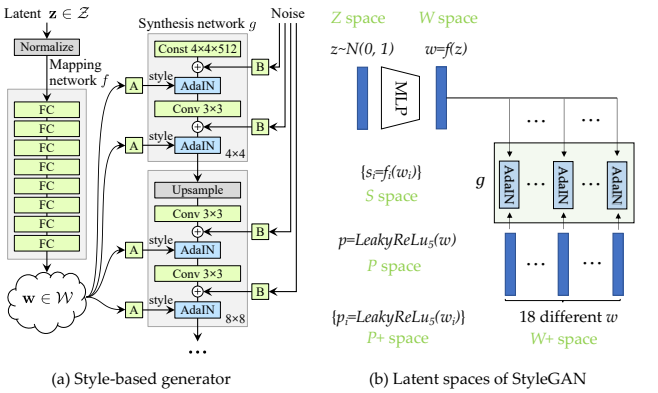

Fig. 2. **Latent spaces that GAN inversion methods choose to embed.** (a) The architecture of Style-based generator; (b) Different latent spaces of Style-based generators. Synthesis network *g* and AdaIN in (b) are the same as in (a).

efforts on latent space analysis and regularization from the original $\mathcal{Z}$ space to the most recent $\mathcal{P}$ space. Figure 2 illustrates the latent spaces that GAN inversion methods choose to embed. These different latent spaces of the style-based generator are derived from the original $\mathcal{Z}$ space.

$\mathcal{Z}$ Space. The generative model in the GAN architecture learns to map the values sampled from a simple distribution, *e.g.*, normal or uniform distribution, to the generated images. These values, sampled *directly* from the distribution, are often called latent codes or latent representations (denoted by $\mathbf{z} \in \mathcal{Z}$), as shown in Figure 2. The structure they form is typically called latent $\mathcal{Z}$ space. Most latent spaces of GANs can be described by the $\mathcal{Z}$ space, including DCGAN [40], PGGAN [14] and BigGAN [15]. However, the constraints of the $\mathcal{Z}$ space subject to a normal distribution limit its representative capacity for the semantic attributes.

$\mathcal{W}$ and $\mathcal{W}^+$ Space. Recent works [16] further convert native $\mathbf{z}$ to the mapped style vectors $\mathbf{w}$ by a nonlinear mapping network f implemented with an 8-layer multilayer perceptron (MLP), which forms another intermediate latent space referred to as $\mathcal{W}$ space. Due to the mapping network and affine transformations, the $\mathcal{W}$ space of StyleGAN contains more untangled features than does the $\mathcal{Z}$ space. Some studies analyze the separability and semantics of both $\mathcal{W}$ and $\mathcal{Z}$ spaces. In [21], Shen *et al.*illustrate that the models using the $\mathcal{W}$ space perform better in terms of separability and representation than those based on the $\mathcal{Z}$ space. The generator G of StyleGAN tends to learn semantic information based on the $\mathcal{W}$ space and performs better than the one using the $\mathcal{Z}$ space. For semantics, the above works evaluate classification accuracy in terms of their latent separation boundaries with respect to different attributes. Because it is not easy to directly embed into $\mathcal{W}$ or $\mathcal{Z}$, some works [24], [25] make use of another latent space, $\mathcal{W}^+$, where a different intermediate latent vector, $\mathbf{w}$, is fed into each of the generator's layers via AdaIN [81]. For a 1024×1024 StyleGAN with 18 layers, $\mathbf{w} \in \mathcal{W}$ has 512 dimensions, and $\mathbf{w} \in \mathcal{W}^+$ has 18×512 dimensions.

$\mathcal{S}$ Space. The style space, abbreviated $\mathcal{S}$, is a space of channelwise style parameters. This $\mathcal{S}$ space is proposed to achieve spatial disentanglement in the spatial dimension

instead of at the semantic level. The spatial entanglement is primarily caused by the intrinsic complexity of style-based generators [16] and the spatial invariance of AdaIN normalization [81]. Xu *et al.* [104] replace the original style codes with disentangled multilevel visual features learned by an encoder. They refer to the space spanned by these style parameters as $\mathcal{Y}$ space, but it is actually a type of $\mathcal{S}$ space. Recent methods [63], [67] have used learned affine transformations to turn $\mathbf{z} \in \mathcal{Z}$ or $\mathbf{w} \in \mathcal{W}$ into channelwise style parameters s for each layer of the generator. By directly intervening the style code $s \in \mathcal{S}$, both methods [63], [67] can achieve fine-grained controls on local translations.

$\mathcal{P}$ Space. A recent method, PULSE [37], has observed a 'soap bubble' effect when searching a generative model's latent space to find the desired points. As indicated by the name, the 'soap bubble' effect is that much of the density of a high-dimensional Gaussian lies close to the surface of a hypersphere. The above authors propose embedding images onto the surface of a hypersphere in $\mathcal{Z}$ space. Based on the observation, Zhu *et al.* [76] propose a $\mathcal{P}$ space. Since the last leaky ReLU uses a slope of 0.2, the transformation from $\mathcal{W}$ space to $\mathcal{P}$ space is $\mathbf{x} = \text{LeakyReLU}_{5.0}(\mathbf{w})$, where $\mathbf{w}$ and $\mathbf{x}$ are latent codes in $\mathcal{W}$ and $\mathcal{P}$ space, respectively. They make the simplest assumption that the joint distribution of latent codes is approximately a multivariate Gaussian distribution and further propose $\mathcal{P}_{\mathcal{N}}$ space to eliminate the dependency and remove redundancy. The transformation from $\mathcal{P}$ space to $\mathcal{P}_{\mathcal{N}}$ space is obtained by PCA whitening: $\hat{\mathbf{v}} = \mathbf{\Lambda}^{-1} \cdot \mathbf{C}^T(\mathbf{x} - \boldsymbol{\mu})$, where $\mathbf{\Lambda}^{-1}$ is a scaling matrix, $\mathbf{C}$ is an orthogonal matrix, and μ is a mean vector. The parameters $\mathbf{C}$, $\mathbf{\Lambda}$, and $\boldsymbol{\mu}$ are obtained from $\text{PCA}(\mathbf{X})$, in which $\mathbf{X} \in \mathbb{R}^{10^6 \times 512}$ consists of 1 million latent samples in $\mathcal{P}$ space. Such transformation normalizes the distribution to be of zero mean and unit variance, leading to the $\mathcal{P}$ space being isotropic in all directions. The P_N^+ space is extended from P_N space: $\mathbf{v} = \left\{\mathbf{\Lambda}^{-1}\mathbf{C}^T(\mathbf{x}_i - \boldsymbol{\mu})\right\}_{i=1}^{18}$. Each of the latent codes is used to demodulate the corresponding StyleGAN feature maps at different layers.

4.2 Inversion Methods

There are three main techniques of GAN inversion, *i.e.*, projecting an image onto the latent space based on learning, optimization, and hybrid formulations, as shown in Figure 3. The learned inverse representations also have other characteristics, *i.e.*, having supported resolution, being semantic-aware, being layerwise, and having out-of-distribution properties. Table 1 lists the characteristics of the existing state-of-the-art GAN inversion methods.

4.2.1 Learning-based GAN Inversion

Learning-based GAN inversion [23], [105], [106] typically involves training an encoding neural network $E(x; \theta_E)$ to map an image, x, onto the latent code $\mathbf{z}$ by

$$\theta_E^* = \arg\min_{\theta_E} \sum_n \mathcal{L}(G(E(x_n; \theta_E)),\, x_n), \tag{9}$$

where x_n denotes the n-th image in the dataset. The objective in (9) is reminiscent of an autoencoder pipeline, with an encoder E and a decoder G. The decoder G is fixed throughout the training. While the optimization problem

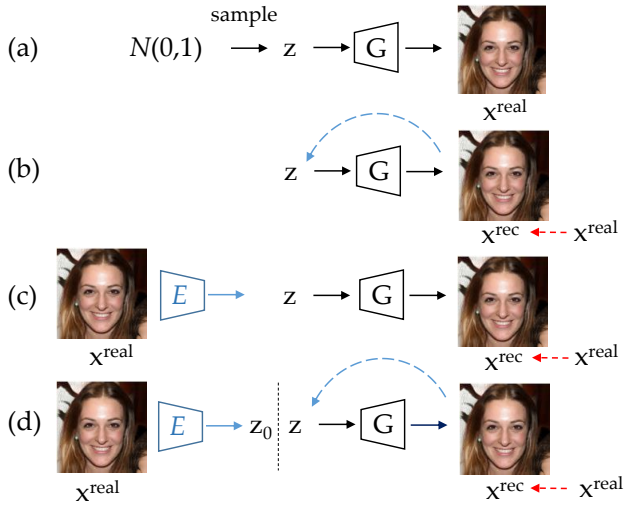

Fig. 3. **Illustration of GAN Inversion Methods.** (a) given a well-trained GAN model, photo-realistic images can be generated from randomly sampled latent vectors. (b) **optimization-based** inversion uses an optimization algorithm to iteratively optimize the latent code to minimize the pixel-wise reconstruction loss. (c) **learning-based** inversion builds an encoder network that maps an image into the latent space. (d) **hybrid** approach uses the encoder to generate an initialization for optimization, *i.e.*, an encoder network is first used to obtain an approximate embedding and then refine it with an optimization algorithm.

described in (1) is the same as the learning objective (9), the learning-based approach often achieves better performance than direct optimization and does not fall into local optima [23], [60].

Perarnau *et al.* [105] propose the invertible conditional GAN (ICGAN) method in which an image, x, is represented by a latent representation, $\mathbf{z}$, and an attribute vector, y, and a modified image x' can be generated by changing y. This approach consists of training an encoder E with a trained CGAN. Different from the method by Zhu *et al.* [23], this encoder E is composed of two subencoders: E_z, which encodes an image to $\mathbf{z}$, and E_y, which encodes an image to y. To train E_z, this method uses the generator to create a dataset of generated images x' and their latent vectors $\mathbf{z}$, minimizes a squared reconstruction loss $\mathcal{L}_{ez}$ between $\mathbf{z}$ and $E_z(G(\mathbf{z}, y'))$ and improves E_y by directly training with $\|y - E_y(x)\|_2^2$. Here, E_y is initially trained by using generated images x' and their conditional information y'. In [61], Guan *et al.*propose the embedding network that consists of two encoders: an identity encoder, E_{id}, to extract identity from the input image x, and an attribute encoder, E_{attr}, to extract attributes from x. Given an input image x in 256×256 resolution, the embedding network generates its latent code $\mathbf{w_e}$, which is then set as the initialization of the iterator. The output of iterator $\mathbf{w_o}$ in turn supervises the training of the embedding network using the MSE loss, LPIPS loss and latent code loss. Tewari *et al.* [53] develop an interpretable model over face semantic parameters of a pretrained StyleGAN $\mathcal{S}$. Given a latent code $\mathbf{w} \in \mathbb{R}^l$ that corresponds to an image I and a vector $\mathbf{p} \in \mathbb{R}^f$ of semantic control parameters, this method learns a function, $\mathcal{R}$, that outputs a modified latent code, $\mathbf{w}' = \mathcal{R}(\mathbf{w}, \mathbf{p})$. The modified latent code $\mathbf{w}'$ is designed to map to a face image $I' = \mathcal{S}(\mathbf{w}')$

that obeys the control parameters $\mathbf{p}$. The encoder $\mathcal{R}$ is trained separately for the different modes of control, *i.e.*, pose, expression and illumination, which is implemented based on a linear two-layer MLP and is trained in a self-supervised manner based on two-way cycle consistency losses and a differentiable face reconstruction network. To improve inversion accuracy, Alaluf *et al.* [33] introduce an iterative refinement mechanism for the encoder. Instead of directly predicting the latent code of a given real image in a single shot, at step t, the encoder operates on an extended input obtained by concatenating the given image $\mathbf{x}$ with the predicted image $y_t = G(\mathbf{w}_t)$: $\Delta_t = E(\mathbf{x}, y_t)$. The latent code at step $t+1$ is then updated as $\mathbf{w}_{t+1} = \Delta_t + \mathbf{w}_t$. The initialization values of $\mathbf{w}_0$ and y_0 are set as the average latent code and its corresponding image, respectively.

To better reuse the layerwise representations learned by the StyleGAN model, Xu *et al.* [68] propose to train a hierarchical encoder based on the feature pyramid network (FPN) by treating the pretrained StyleGAN generator as a learned loss (similar to perceptual loss [29] using learned VGG [107]). The learned disentangled multilevel visual features $\{\mathbf{y}^{(\ell)}\}_{\ell=1}^{L}$ are then fed into per-layer adaptive instance normalization (AdaIN) [81] of the fixed StyleGAN generator to obtain the desired images by replacing the original style code:

$$\texttt{AdaIN}(x_i^{(\ell)}, \mathbf{y}^{(\ell)}) = \mathbf{y}_{s,i}^{(\ell)} \frac{x_i^{(\ell)} - \mu(x_i^{(\ell)})}{\sigma(x_i^{(\ell)})} + \mathbf{y}_{b,i}^{(\ell)}, \tag{10}$$

where L is the number of convolutional layers, $x = G(z)$, $x_i^{(\ell)}$ indicates the i-th channel of the normalized feature map from the ℓ-th layer; $\mu(\cdot)$ and $\sigma(\cdot)$ denote the mean and variance, respectively; and $\mathbf{y}_s^{(\ell)}$ and $\mathbf{y}_b^{(\ell)}$ correspond to the scale and weight parameters in AdaIN, respectively. Richardson *et al.* [31] propose a small mapping network named the map2style module to extract the learned styles from the corresponding feature map. The 18 map2style modules predict 18 single-layer latent codes separately. Wei *et al.* [32] find 18 mapping modules unnecessary and propose a very simple and efficient head, which just consists of an average pooling layer and a fully connected layer. Given three different semantic levels of features obtained by FPN, these three heads produce $\mathbf{w}_{15}, \cdots, \mathbf{w}_{18}$, $\mathbf{w}_{10}, \cdots, \mathbf{w}_{14}$, and $\mathbf{w}_1, \cdots, \mathbf{w}_9$ from the shallow, medium, and deep features, respectively.

Although some methods [34], [35], [36], [108] use additive encoder networks to learn the inverse mapping of GANs, we do not categorize them as GAN inversion since their goals are to *jointly train* the encoder with both the generator and the discriminator, instead of determining the latent space of a trained GAN model.

4.2.2 Optimization-based GAN Inversion

Existing optimization-based GAN inversion methods typically reconstruct a target image by optimizing over the latent vector

$$\mathbf{z}^* = \arg\min_{\mathbf{z}} \ell(x, G(\mathbf{z}; \theta)), \tag{11}$$

where x is the target image, and G is a GAN generator parameterized by θ. Optimization-based GAN inversion typically optimizes the latent code based on either gradient descent [24], [25], [44], [44], [109], [109], [110] or other iterative algorithms [95], [111]. For example, Ramesh *et al.* [111] and Voynov *et al.* [95] use the Jacobian decomposition to analyze the latent space of a pretrained GAN model. Specifically, the left eigenvectors of the Jacobian matrix for the generator are used to indicate the most disentangled directions. In these methods, an interpretable curve is constructed by a latent point, z_0, and a direction from the corresponding k-th left eigenvector. Voynov *et al.* [95] show that once the latent vector moves along that curve, the generated image appears to be transformed smoothly. Nevertheless, while the constructed curves often capture interpretable transformations, the effects are typically entangled (*i.e.*, lighting and geometrical transformations appear simultaneously). This method also requires an expensive (in terms of both memory and runtime) iterative process for computing the Jacobian matrix on each step of the curve construction and has to be applied to each latent code independently. As such, a lightweight approach that identifies a set of directions at once is also developed. We note that the method by Voynov *et al.* [95] can be applied to a larger number of directions, while the approach by Ramesh *et al.* [111] is limited to the maximal number of discovered directions equal to the dimension of latent space.

To address the local minima issue, numerous optimization methods have been developed. Generally, there are two types of optimizers: gradient-based (ADAM [112], L-BFGS [113], Hamiltonian Monte Carlo (HMC) [114]) and gradient-free (covariance matrix adaptation (CMA) [115]) methods. For example, the ADAM optimizer is used in the Image2StyleGAN [24], and the L-BFGS scheme is used in the approach by Zhu *et al.* [23]. Huh *et al.* [27] experiment with various gradient-free optimization methods in the Nevergrad library [116] with the default optimization hyperparameters and find that CMA and its variant BasinCMA perform the best for optimizing the latent vector when inverting images in challenging datasets (*e.g.*, LSUN Cars) to the latent space of StyleGAN2 [17].

Another important issue for optimization-based GAN inversion is initialization. Since (1) is highly nonconvex, the reconstruction quality strongly relies on a good initialization of $\mathbf{z}$ (sometimes $\mathbf{w}$ for StyleGAN [16]). The experiments show that using different initializations leads to a significant perceptual difference in generated images [14], [15], [16], [40]. An intuitive solution is to start from several random initializations and obtain the best result with the minimal cost. Image2StyleGAN [24] analyzes two choices for the initialization $\mathbf{w}^*$ based on random selection and mean latent code $\overline{\mathbf{w}}$ motivated by the observation from [16] that the distance to $\overline{\mathbf{w}}$ can be used to identify low-quality faces. However, a prohibitively large number of random initializations may be required to obtain a stable reconstruction [23], which makes real-time processing impossible. Thus, some [23], [53], [61] instead train a deep neural network to minimize (1) directly, as introduced in Section 4.2.1.

We note that some [23], [61], [106] propose using an encoder to provide better initialization for optimization (which is discussed in Section 4.2.3).

4.2.3 Hybrid GAN Inversion

The hybrid methods [22], [23], [26], [61], [106] exploit the advantages of both approaches discussed above. As one of the pioneering works in this field, Zhu *et al.* [23] propose a framework that first predicts $\mathbf{z}$ of a given real photo x by training a separate encoder $E(x;\theta_E)$, which then uses the obtained $\mathbf{z}$ as the initialization for optimization. The learned predictive model serves as a fast bottom-up initialization for the nonconvex optimization problem (1).

Subsequent studies basically follow this framework and have proposed several variants. For example, to invert G, Bau *et al.* [106] begin by training a network E to obtain a suitable initialization of the latent code $\mathbf{z}_0 = E(x)$ and its intermediate representation $\mathbf{r}_0 = g_n(\cdots(g_1(\mathbf{z}_0)))$, where $g_n(\cdots(g_1(\cdot)))$ in a layerwise representation of $G(\cdot)$. This method then uses $\mathbf{r}_0$ to initialize a search for $\mathbf{r}^*$ to obtain a reconstruction $x' = G(\mathbf{r}^*)$ close to the target x (see Section 4.3.3 for more details). Zhu *et al.* [22] show that in most existing methods, generator G does not provide its domain knowledge to guide the training of encoder E since the gradients from $G(\cdot)$ are not taken into account at all. As such, a domain-specific GAN inversion approach is developed, which both reconstructs the input image and ensures that the inverted code is meaningful for semantic editing (see Section 4.3.2 for more details).

Without using the above framework, Guan *et al.* [61] propose a collaborative learning framework for StyleGAN inversion, where the embedding network gives a reasonable latent code initialization $\mathbf{w_e}$ for the optimization-based iterator, and the updated latent code from the iterator $\mathbf{w_o}$, in turn, supervises the embedding network to produce more accurate latent codes. The objective functions of embedding network $\mathcal{L}_{emb}$ and iterator $\mathcal{L}_{opt}$ are

$$\begin{aligned}\mathcal{L}_{emb} &= \lambda_1 \underbrace{||\mathbf{w_e} - \mathbf{w_o}||_2^2}_{\text{latent loss}} + \lambda_2 \underbrace{||x_e - x_o||_2^2}_{\text{image loss}} + \lambda_3 \underbrace{\Phi(x_e, x_o)}_{\text{feature loss}}, \\ \mathcal{L}_{opt} &= ||G(\mathbf{w}) - x||_2^2 + \alpha\Phi(G(\mathbf{w}), x),\end{aligned} \tag{12}$$

where $\mathbf{w} \in \mathcal{W}^+$ is the latent code to be optimized, G is a frozen generator of StyleGAN pretrained on the FFHQ dataset [16], $x_e = G(\mathbf{w_e})$ and $x_o = G(\mathbf{w_o})$ are generated from $\mathbf{w_e}$ and $\mathbf{w_o}$ by the StyleGAN generator G, $\Phi(\cdot)$ is the LPIPS loss [30], and λ_1, λ_2, and λ_3, α are the loss weights.

4.3 Characteristics of GAN Inversion Methods

In this section, we discuss some important characteristics of GAN inversion methods *i.e. having supported resolution*, *being semantic-aware*, *being layerwise*, and *having out-of-distribution* properties.

4.3.1 Supported Resolution

The image resolution that a GAN inversion method can support is mainly determined by the capacity of generators and inversion mechanisms. Zhu *et al.* [23] use GCGANs trained on several datasets with images of 64×64 pixels, and Bau *et al.* [47], [117] adopt PGGANs [14] trained with images of size 256×256 pixels from Lsun [43]. However, some methods cannot fully leverage the pretrained GAN model. In [22], Zhu *et al.*propose an encoder to map the given images to the latent space of StyleGAN. This method (Figure 4 (a)) performs well for images of 256×256 pixels but does not scale up well to images of 1024×1024 pixels due to the heavy computational cost (where 1/n in the figure means semantic feature maps of 1/n original input resolution). Conversely, the pSp method proposed by Richardson *et al.* [31] (Figure 4 (b)) can synthesize images of 1024×1024 pixels, regardless of input image size, since the 18 map2style modules they proposed are used to predict 18 single-layer latent codes separately. Wei *et al.* [32] propose a similar model but with a lightweight encoder. Similar to [31], features from three semantic levels are used to predict different parts of the latent codes. Nevertheless, this model predicts 9, 5, and 4 layers of latent codes from each semantic level, as shown in Figure 4 (c). Recently, numerous applications such as face swapping on megapixels [118] and infinite-resolution image synthesis [119] are developed as image inversion models can support high-resolution images.

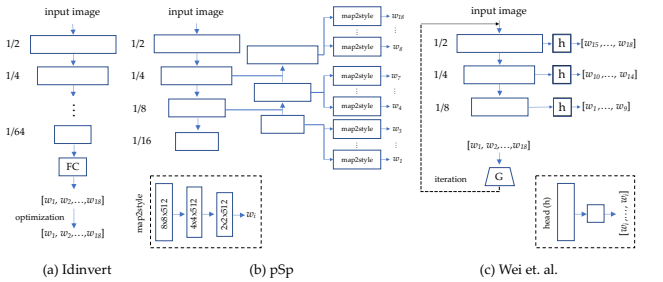

Fig. 4. **Network structure comparison of IDinvert [22], pSp [31], and Wei *et al.* [32].**

4.3.2 Semantic Awareness

GAN inversion methods with semantic-aware properties can perform image reconstruction at the pixel level and align the inverted code with the knowledge that emerged in the latent space. Semantic-aware latent codes can better support image editing by reusing the rich knowledge encoded in the GAN models. As shown on the upper panel of Figure 5, existing approaches typically randomly sample a collection of latent codes $\mathbf{z}$ and feed them into $G(\cdot)$ to obtain the corresponding synthesis x'. The encoder $E(\cdot)$ is then trained by

$$\min_{\Theta_E} \mathcal{L}_E = \|\mathbf{z} - E(G(\mathbf{z}))\|_2, \tag{13}$$

where $\|\cdot\|_2$ denotes the l_2 distance, and Θ_E represents the parameters of the encoder $E(\cdot)$.

Collins *et al.* [52] use a latent object representation to synthesize images with different styles and reduce artifacts. However, the supervision by only reconstructing $\mathbf{z}$ is not sufficient to train an accurate encoder. To alleviate this issue, Zhu *et al.* [22] propose a domain-specific GAN inversion approach to recover the input image at both the pixel and semantic levels. This method first trains a domain-guided encoder to map the image space to the latent space such that all codes produced by the encoder are in-domain latent codes. Then, they perform instance-level domain-regularized optimization by involving the encoder as a regularization term. Such optimization helps better reconstruct

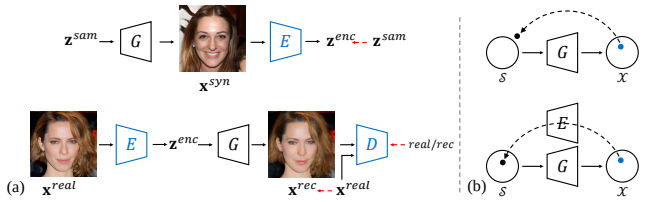

Fig. 5. **Comparisons between the training of (upper) conventional encoder and (lower) domain-guided encoder proposed in [22] for GAN inversion.** Blue blocks represent trainable models and red dashed arrows indicate the supervisions. The domain-guided encoder is trained to recover the real images, instead of being trained with synthesized data to recover the latent code. The generator G is well-trained with fixed weights during training E. (b) The comparison between the conventional optimization and the domain-regularized optimization proposed in [22]. The well-trained domain-guided encoder E is involved as a regularization to fine-tune the latent code in the semantic domain during $\mathbf{z}$ optimization.

the pixel values without affecting the semantic property of the inverted code. The training process is formulated as

$$\min_{\Theta_E} \mathcal{L}_E = \|x - G(E(x))\|_2 + \lambda_1 \|F(x) - F(G(E(x)))\|_2 - \lambda_2 \mathbb{E}[D(G(E(x)))], \tag{14}$$

where $F(\cdot)$ represents the VGG feature extraction, $\mathbb{E}[D(\cdot)]$ is the discriminator loss, and λ_1 and λ_2 are the perceptual and discriminator loss weights, respectively.

The inverted code from the proposed domain-guided encoder can well reconstruct the input image based on the pretrained generator and ensure the code itself to be semantically meaningful. However, the code still needs refinement to better fit the individual target image at the pixel values. Based on the domain-guided encoder, Zhu *et al.*design a domain-regularized optimization with two modules: (i) the output of the domain-guided encoder is used as a starting point to avoid a local minimum and also shorten the optimization process, and (ii) a domain-guided encoder is used to regularize the latent code within the semantic domain of the generator. The objective function is

$$\mathbf{z}^* = \arg\min_{\mathbf{z}} \|x - G(\mathbf{z})\|_2 + \lambda_1' \|F(x) - F(G(\mathbf{z}))\|_2 + \lambda_2' \|\mathbf{z} - E(G(\mathbf{z}))\|_2, \tag{15}$$

where x is the target image to invert, and λ_1' and λ_2' are the loss weights corresponding to the perceptual loss and the encoder regularizer, respectively.

4.3.3 Layerwise

As it is not feasible to determine the generator for the full inversion problem defined by problem (1) when the number of layers is large, a few approaches [26], [60], [120] have been developed to solve a tractable subproblem by decomposing the generator G into layers:

$$G = G_f(g_n(\cdots((g_1(\mathbf{z})))), \tag{16}$$

where $g_1, \ldots, g_n$ are the early layers of G, and G_f constructs all the later layers of G.

The simplest layerwise GAN inversion is based on one layer. We start inverting a single layer to determine if $\min_{\mathbf{z}} \|x - G(\mathbf{z})\|_p = 0$, a specific formulation of (1), holds for any p-norm. Since the problem is nonconvex, additional assumptions are required [121] for gradient descent to find $\arg\min_{\mathbf{z}} \|x - G(\mathbf{z})\|$. When the problem is realizable, however, to find feasible $\mathbf{z}$ such that $x = \text{ReLU}(\mathbf{W}\mathbf{z} + \mathbf{b})$, one could invert the function by solving a linear programming problem:

$$\begin{aligned} \mathbf{w}_i{}^\top \mathbf{z} + b_i &= x_i, \quad \forall i \text{ s.t. } x_i > 0, \\ \mathbf{w}_i{}^\top \mathbf{z} + b_i &\le 0, \quad \forall i \text{ s.t. } x_i = 0. \end{aligned} \tag{17}$$

The solution set of (17) is convex and forms a polytope. However, it also possibly includes uncountable feasible points [120], which makes it unclear how to invert layerwise inversion. Several approaches make additional assumptions to generalize the above result to deeper neural networks. Lei *et al.* [120] assume that the input signal is corrupted by bounded noise in terms of ℓ_1 or ℓ_∞ and propose an inversion scheme for generative models using linear programs layer by layer. The analysis for an assuredly stable inversion is restricted to cases where the following hold: (1) the weights of the network should be Gaussian *i.i.d.*variables; (2) each layer should be expanded by a constant factor; and (3) the last activation function should be ReLU [122] or leaky ReLU [123]. However, these assumptions often do not hold in practice. Aberdam *et al.* [60] relax the expansion assumption of [120] and propose a method that relies on the expansion of the number of nonzero elements. They reformulate problem (1) with ℓ_2 to a layerwise expression:

$$\arg\min_{\mathbf{z}} \left\| \mathbf{y} - \phi((\prod_{i=L}^{0} \mathbf{W}_i^{\hat{\mathcal{S}}_{i+1}})\mathbf{z}) \right\|_2^2, \tag{18}$$

where $\mathbf{y} = G(\mathbf{z}) + \mathbf{e}$ and $\{\mathcal{S}_i\}_{i=1}^L$ are the support sets of each layer, ϕ is an invertible activation function *e.g.*tanh, sigmoid, or piecewise linear, and $\mathbf{W}_i^{\mathcal{S}}$ denote the row-supported matrix according to the support set $\mathcal{S}$. Thus, the sparsity of all the intermediate feature vectors can be used to invert the model by solving sparse coding problems layer by layer. This method does not rely on the distribution of the weights or on the chosen activation function of the last layer. However, this approach can only be applied to invert very shallow networks.

To invert complex state-of-the-art GANs, Bau *et al.* [26] propose solving the easier problem of inverting the final layers G_f:

$$x' = G_f(\mathbf{r}^*), \tag{19}$$

where $\mathbf{r}^* = \arg\min_{\mathbf{r}} \ell(G_f(\mathbf{r}), x)$, $\mathbf{r}$ is an intermediate representation, and ℓ is a distance metric in the image feature space. They solve the inversion problem (1) in a two-step hybrid GAN inversion framework: first constructing a neural network E that approximately inverts the entire G and computes an estimate $\mathbf{z}_0 = E(x)$ and subsequently solving an optimization problem to identify $\mathbf{r}^* \approx \mathbf{r}_0 = g_n(\cdots(g_1(\mathbf{z}_0)))$ that generates a reconstructed image $G_f(\mathbf{r}^*)$ to closely recover x. For each layer $g_i \in \{g_1, ..., g_n, G_f\}$, a small network e_i is first trained to invert g_i. That is, when defining $\mathbf{r}_i = g_i(\mathbf{r}_{i-1})$, the goal is to learn a network, e_i, that approximates the computation $\mathbf{r}_{i-1} \approx e_i(\mathbf{r}_i)$ and ensures the predictions of the network e_i to well preserve the output

Fig. 6. **Illustration of face image manipulation.** These are real image editing results from using StyleFlow [71].

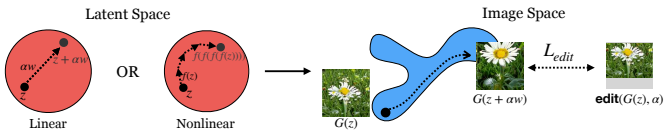

Fig. 7. Illustration of discovering interpretable directions in the latent space [20]. The goal is to find a path in $\mathcal{Z}$ space to transform the generated image $G(\mathbf{z})$ to its edited version $\texttt{edit}(G(\mathbf{z},\alpha))$, *e.g.*, an $\alpha\times$ zoom. The transformation can be represented as $G(\mathbf{z}+\alpha\mathbf{w})$ for a linear walk or $G(f(f(...(\mathbf{z}))))$ for a non-linear walk.

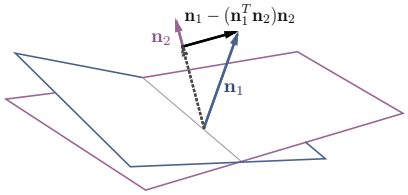

Fig. 8. **Illustration of the non-interference property in subspace.** The projection of $\mathbf{n}_1$ onto $\mathbf{n}_2$ is subtracted from $\mathbf{n}_1$, resulting in a new direction $\mathbf{n}_1-(\mathbf{n}_1^\top\mathbf{n}_2)\mathbf{n}_2$. This figure is from [21].

of layer g_i, *i.e.*, $\mathbf{r_i}\approx g_i(e_i(\mathbf{r}_i))$. As such, e_i is trained to minimize both left- and right-inversion losses:

$$\begin{aligned}\mathcal{L}_{\text{L}}&=\mathbb{E}_{\mathbf{z}}[||\mathbf{r}_{i-1}-e(g_i(\mathbf{r}_{i-1}))||_1],\\ \mathcal{L}_{\text{R}}&=\mathbb{E}_{\mathbf{z}}[||\mathbf{r}_{i}-g_i(e(\mathbf{r}_{i}))||_1],\\ e_i&=\arg\min_{e}\quad \mathcal{L}_{\text{L}}+\lambda_{\text{R}}\,\mathcal{L}_{\text{R}},\end{aligned}\tag{20}$$

where $||\cdot||_1$ denotes an $\mathcal{L}_1$ loss, and λ_{R} is set as 0.01 to emphasize the reconstruction of $\mathbf{r}_{i-1}$. To focus on training near the manifold of representations produced by the generator, this method uses sample $\mathbf{z}$ and layers g_i to compute samples of $\mathbf{r}_{i-1}$ and $\mathbf{r}_i$ such that $\mathbf{r}_{i-1}=g_{i-1}(\cdots g_1(\mathbf{z}))$. Once all the layers are inverted, an inversion network for all of G can be composed as follows:

$$E^*=e_1(e_2(\cdots(e_n(e_f(x))))).\tag{21}$$

The results can be further improved by fine-tuning the composed network E^* to invert G jointly as a whole and obtain the final result E.

4.3.4 Out of Distribution

GAN inversion methods can support inverting the images, especially real images in the wild, that are not generated by the same process of the training data. We refer to this ability as out-of-distribution generalization [124], [125], [126]. This property is a prerequisite for GAN inversion methods to edit real images. The latent code-based editing methods [95], [127], [128], [129] can be combined with inversion methods to discover a desired code with certain attributes. Out-of-distribution generalization has been demonstrated with many GAN inversion methods. In [54], Daras *et al.*show that a local sparse layer (based on local context) can significantly help better invert a GAN model than a dense layer. They demonstrate the generalization ability of the proposed method by manipulating an image of redshank searched via Google that did not appear in the training process. Pan *et al.* [28] propose the deep generative prior (DGP) to embed rich knowledge of natural images. As a generic image prior, the DGP method can be used to restore the missing information of a degraded image by progressively reconstructing it under the discriminator metric. Recently, Abdal *et al.* [71] have introduced the StyleFlow method to the conditional exploration of the StyleGAN latent space. The attribute-conditioned sampling and attribute-controlled editing of the StyleGAN are analyzed by the proposed conditional continuous normalizing flows. As demonstrated in Figure 6, this method can handle extreme pose, asymmetrical expressions, and age diversity well compared to the concurrent techniques. Zhu *et al.* [22] propose a domain-specific GAN inversion approach to recover the input image at both the pixel and semantic levels. Although trained only with the FFHQ dataset, their model can generalize to not only real face images from multiple face datasets [130], [131], [132] but also paintings, caricatures, and black and white photos collected from the Internet. Some methods [25], [66] further exploit how to invert an image into a desired latent code even given a degraded observation. Given a masked image and a style image, the algorithm proposed by Abdal *et al.* [25] can find a latent code that completes the defected area with the style image. In addition to the image, recent methods also show out-of-distribution generalization ability for other modalities, *i.e.* sketch [31], [32] and text [133], [134], [135].

4.4 Latent Space Navigation

Inversion is not the ultimate goal. The reason that we invert a real image into the latent space of a trained GAN model is that it allows us to manipulate the inverted image in the latent space by discovering the desired code with certain attributes. This technique is usually known as latent space navigation [64], [65], GAN steerability [20], [128], and latent code manipulation [21], among others in the literature. Although often regarded as an independent research field, it acts as an indispensable component of GAN inversion for manipulation [72], [135]. Many inversion methods [33], [73] also involve efficient discovery of a desired latent code. In Section 4.1, we have introduced different latent spaces. In this section, we introduce how we discover interpretable and noninterference directions in these latent spaces.

4.4.1 Discovering Interpretable Directions

Some methods support discovering interpretable directions in the latent space, *i.e.*, controlling the generation process by varying the latent codes $\mathbf{z}$ in the desired directions $\mathbf{n}$ with step α, which can often be represented as the vector arithmetic $\mathbf{z}' = \mathbf{z} + \alpha\mathbf{n}$. Such directions are currently discovered in supervised, unsupervised, or self-supervised manners. Recent methods have also been proposed to directly compute the interpretable directions in closed form from the pretrained models without any kind of training or optimization.

Supervised Setting. Existing supervised learning-based approaches typically randomly sample a large amount of latent codes, synthesize a collection of images, and annotate them with some predefined labels by introducing a pretrained classifier (*e.g.*, predicting face attributes or light directions) [19], [20], [21], [71] or extracting statistical image information (*e.g.*, color variations) [136]. For example, to interpret the face representation learned by GANs, Shen *et al.* [21] employ some off-the-shelf classifiers to learn a hyperplane in the latent space serving as the separation boundary and predict semantic scores for synthesized images. Abdal *et al.* [71] learn a semantic mapping between the $\mathcal{Z}$ space and the $\mathcal{W}$ space by using continuous normalizing flows (CNF). Both methods rely on the availability of attributes (typically obtained by a face classifier network), which might be difficult to obtain for new datasets and could require manual labeling effort.

Unsupervised Setting. The supervised setting would introduce bias into the experiment since the sampled codes and synthesized images used as supervision are different in each sampling and may lead to different discoveries of interpretable directions [129]. It also severely restricts a range of directions that existing approaches can discover, especially when the labels are missing. Furthermore, the individual controls discovered by these methods are typically entangled, affecting multiple attributes, and are often nonlocal. Thus, some methods [64], [95], [137], [138] aim to discover interpretable directions in the latent space in an unsupervised manner, *i.e.*, without the requirement of paired data. For example, Härkönen *et al.* [138] create interpretable controls for image synthesis by identifying important latent directions based on PCA applied in the latent or feature space. The obtained principal components correspond to certain attributes, and the selective application of the principal components allows for the control of features. Jahanian *et al.* [20] optimize trajectories (both linear and nonlinear, as shown in Figure 7) in a self-supervised manner. Taking the linear walk $\boldsymbol{w}$ as an example, given an inverted source image $G(\mathbf{z})$, they learn $\boldsymbol{w}$ by minimizing the following function:

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \mathbb{E}_{\mathbf{z},\alpha}[\mathcal{L}(G(\mathbf{z}+\alpha\mathbf{w}), \texttt{edit}(G(\mathbf{z}), \alpha))], \quad (22)$$

where $\mathcal{L}$ measures the distance between the generated image $G(\mathbf{z}+\alpha\boldsymbol{w})$ after taking an α-step in the latent direction and the target $\texttt{edit}(G(\mathbf{z}), \alpha)$ derived from the source image $G(\mathbf{z})$.

Closed-form Solution. Most recently, two methods [128], [129] found that the interpretable directions can be directly computed in *closed form* without any kind of training or optimization. Specifically, Nurit *et al.* [128] observe that the output of the first layer in BigGAN [15] (the first layer maps $\mathbf{z}$ into a tensor with low spatial resolution) already has spatial coordinates and determines the coarse structure of the generated image, which suggests that applying the geometric transformation to the output of the first layer is similar to directly applying a direction $\mathbf{q}$ to the generated image, *i.e.*, $G(\mathbf{z}+\mathbf{q}) \approx \mathcal{T}\{G(\mathbf{z})\}$ for every $\mathbf{z}$, where G is a pretrained generator, $\mathcal{T}$ is the desired transformation in the image, and $\mathbf{q}$ is the target direction in the latent space. The goal is to bring $\mathbf{W}(\mathbf{z}+\mathbf{q})+\mathbf{b}$ as close as possible to $\mathbf{P}(\mathbf{W}\mathbf{z}+\mathbf{b})$. $\mathbf{P}$ denotes the transformation matrix corresponding to $\mathcal{T}$ in the resolution of the first layer's output. $\mathbf{W}$ and $\mathbf{b}$ are the weights and biases of the first layer, respectively. To guarantee that this holds over random draws of $\mathbf{z}$, they formulate the problem as follows:

$$\min_{\mathbf{q}} \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}}[\|\mathbf{D}\Big(\mathbf{W}(\mathbf{z}+\mathbf{q})+\mathbf{b}-\mathbf{P}(\mathbf{W}\mathbf{z}+\mathbf{b})\Big)\|^2], \quad (23)$$

where $p_{\mathbf{z}}$ is the probability density function of $\mathbf{z}$, and $\mathbf{D}$ is a diagonal matrix that can be used to assign different weights to different elements of the tensors. Assuming $\mathbb{E}[\mathbf{z}] = 0$, a closed-form solution, $\mathbf{q}$, can be obtained for the optimal linear direction corresponding to transformation $\mathbf{P}$:

$$\mathbf{q} = (\mathbf{W}^T\mathbf{D}^2\,\mathbf{W})^{-1}\mathbf{W}^T\mathbf{D}^2(\mathbf{P}-\mathbf{I})\,\mathbf{b}. \quad (24)$$

With the linear trajectories $\mathbf{z}+\mathbf{q}$, the generated image inevitably becomes distorted or even meaningless after many steps. Thus, they further propose nonlinear trajectories to remedy the problems. The walks in the latent space take the form $\mathbf{z}_{n+1} = \mathbf{M}\mathbf{z}_n + \mathbf{q}$, where the transformation $\mathbf{P}$ is determined by a vector $\mathbf{q}$ and a diagonal matrix $\mathbf{M}$. Problem (23) is then formulated as follows:

$$\min_{\mathbf{M},\mathbf{q}} \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}}[\|\mathbf{D}\Big(\mathbf{W}(\mathbf{M}\mathbf{z}+\mathbf{q})+\mathbf{b}-\mathbf{P}(\mathbf{W}\mathbf{z}+\mathbf{b})\Big)\|^2]. \quad (25)$$

Assuming again that $\mathbb{E}[\mathbf{z}] = 0$ and making an additional assumption that $\mathbb{E}[\mathbf{z}\mathbf{z}^T] = \sigma_z^2\mathbf{I}$, the solution for $\mathbf{q}^*$ remains the same as in (24), and the solution for $\mathbf{M}$ is

$$\mathbf{M}_{i,i} = \frac{\mathbf{w}_i^T\mathbf{D}^2\mathbf{P}\,\mathbf{w}_i}{\mathbf{w}_i^T\mathbf{D}^2\,\mathbf{w}_i}, \quad (26)$$

where $\mathbf{w}_i$ is the i-th column of $\mathbf{W}$.

Shen *et al.* [129] observe that the semantic transformation of an image, usually denoted by moving the latent code toward a certain direction $\mathbf{n}' = \mathbf{z}+\alpha\mathbf{n}$, is actually determined by the latent direction $\mathbf{n}$, which is independent of the sampled code $\mathbf{z}$. Based on this, they turn to finding the directions $\mathbf{n}$ that can cause a significant change in the output image $\Delta\mathbf{y}$, *i.e.*, $\Delta\mathbf{y} = \mathbf{y}'-\mathbf{y} = (\mathbf{A}(\mathbf{z}+\alpha\mathbf{n})+\mathbf{b})-(\mathbf{A}\mathbf{z}+\mathbf{b}) = \alpha\mathbf{A}\mathbf{n}$, where $\mathbf{A}$ and $\mathbf{b}$ are the weight and bias of certain layers in G, respectively. The obtained formula, $\Delta\mathbf{y} = \alpha\mathbf{A}\mathbf{n}$, suggests that the desired editing with direction $\mathbf{n}$ can be achieved by adding the term $\alpha\mathbf{A}\mathbf{n}$ onto the projected code and indicates that the weight parameter $\mathbf{A}$ should contain the essential knowledge of image variations. The problem of exploring the latent semantics can thus be factorized by solving the following optimization problem:

$$\mathbf{n}^* = \arg\max_{\{\mathbf{n}\in\mathbb{R}^d:\ \mathbf{n}^T\mathbf{n}=1\}} ||\mathbf{A}\mathbf{n}||_2^2. \quad (27)$$

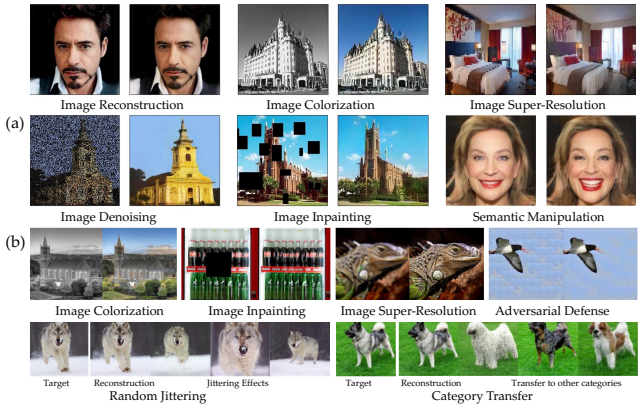

Fig. 9. **Illustration of image processing using GAN inversion.** GAN inversion does not require task-specific dense-labeled datasets and can be applied to many tasks like image reconstruction, image restoration and image manipulation. The upper illustration (a) is from mGANPrior [51] and the lower (b) is from DGP [28].

The desired directions $\mathbf{n}^*$, *i.e.*, a closed-form factorization of latent semantics in GANs, should be the eigenvectors of the matrix $\mathbf{A}^T\mathbf{A}$.

4.4.2 Discovering Noninterference Directions

When several attributes are involved, editing one may affect another since some semantics are not separated. Some methods aim to tackle multi-attribute image manipulation without interference. This characteristic is also named multidimensional [59] or conditional editing [21] in the literature. For example, to edit multiple attributes, Shen *et al.* [21] formulate the inversion-based image manipulation as $x' = G(\mathbf{z}^* + \alpha\mathbf{n})$, where $\mathbf{n}$ is a unit normal vector indicating a hyperplane defined by two latent codes $\mathbf{z}_1$ and $\mathbf{z}_1$. In this method, k attributes $\{\mathbf{z}_1, \cdots, \mathbf{z}_k\}$ can form m (where $m \leq k(k-1)/2$) hyperplanes $\{\mathbf{n}_1, \cdots, \mathbf{n}_m\}$. Noninterference manipulation of multiple attributes means that $\{\mathbf{n}_1, \cdots, \mathbf{n}_m\}$ should be orthogonal. If this condition does not hold, then some semantics will correlate with each other, and $\mathbf{n}_i^\top \mathbf{n}_j$ can be used to measure the entanglement between the i-th and j-th semantics. In particular, this method uses projection to orthogonalize different vectors. As shown in Figure 8, given two hyperplanes with normal vectors $\mathbf{n}_1$ and $\mathbf{n}_2$, the goal is to find a projected direction $\mathbf{n}_1 - (\mathbf{n}_1^\top \mathbf{n}_2)\mathbf{n}_2$ such that moving samples along this new direction can change "attribute one" without affecting "attribute two". For the case where multiple attributes are involved, they subtract the projection from the primal direction onto the plane that is constructed by all conditioned directions. Other GAN inversion methods [56], [61] based on pretrained StyleGAN [16] or StyleGAN2 [17] models can also manipulate multiple attributes due to the stronger separability of $\mathcal{W}$ space than of $\mathcal{Z}$ space. However, as observed by recent methods [63], [67], [133], some attributes remain entangled in the $\mathcal{W}$ space, leading to some unwanted changes when we manipulate a given image. Instead of manipulating in the semantic $\mathcal{W}$ space, Liu *et al.* [63] propose the $\mathcal{S}$ space (style space), where all facial attributes are almost linearly separable. The style code is formed by concatenating the output of all affine layers of the StyleGAN2 [17] generator. Experiments show that the $\mathcal{S}$ space can alleviate *spatially entangled changes* and exert precise local modifications. By intervening the style code $s \in \mathcal{S}$ directly, their method can manipulate different facial attributes along with various semantic directions without affecting others and can achieve fine-grained controls on local translations.

5 Applications

Finding an accurate solution to the inversion problem allows us to match the target image without losing downstream editing capabilities. GAN inversion does not require task-specific dense-labeled datasets and can be applied to many tasks such as image manipulation, image interpolation, image restoration, style transfer, novel-view synthesis and even adversarial defense, as shown in Figure 9. In addition to common applications in image processing, in the last few months, GAN inversion techniques have been widely introduced to many other tasks, such as 3D reconstruction [139], [140], image understanding [141], [142], multimodal learning [133], [134], [135], [143], [144], and medical imaging [145], [146], [147], [148], which shows its versatility for various tasks and increasing attentions from the greater research community.

5.1 Image Manipulation

Given an image x, we want to edit certain regions by manipulating its latent codes $\mathbf{z}$ and obtain the manipulated $\mathbf{z}'$ of the target image x' by linearly transforming the latent representation from a trained GAN model G. This can be formulated in the framework of GAN inversion as the operation of adding a scaled difference vector:

$$x' = G(\mathbf{z}^* + \alpha\mathbf{n}), \tag{28}$$

where $\mathbf{n}$ is the normal direction corresponding to a particular semantic in the latent space, and α is the step for manipulation. In other words, if a latent code is moved in a certain direction, then the semantics contained in the output image should vary accordingly. For example, Xu *et al.* [68] use a hierarchical encoder to obtain the sampled features, matching between the learned GH-Feat with the internal representation of the StyleGAN generator and leading to high-fidelity global and local editing results from multiple levels. Voynov *et al.* [95] gradually determine the direction corresponding to the background removal or background blur without changing the foreground. In [21], Shen *et al.* achieve single and multiple facial attribute manipulation by projecting and orthogonalizing different vectors. Recently, Zhu *et al.* [22] perform semantic manipulation by either decreasing or increasing the semantic degree. Both methods [21], [22] use a projection strategy to search for the semantic direction $\mathbf{n}$.

Some methods can perform region-of-interest editing, which allows for the editing of some desired regions in a given image with user manipulation. Such operations often involve additional tools to select the desired region, as shown in Figure 10. For example, Abdal *et al.* [24], [25] analyze the defective image embedding of StyleGAN trained on FFHQ [16], *i.e.*, the embedding of images with

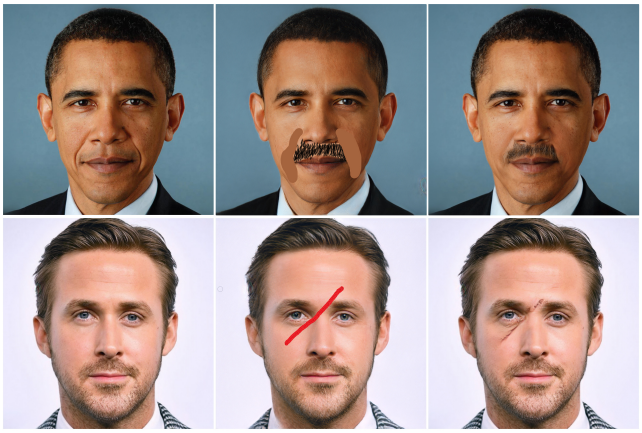

Fig. 10. **Illustration of region-of-interest editing [25].** From left to right: base image; scribbled image; result of local edits.

Fig. 11. **Results of artifacts correction.** First row shows are examples generated by PGGAN [14]. The second row presents the gradually corrected synthesis by moving the latent codes along the positive quality direction. This figure is from [21].

masked regions. The experiments show that the StyleGAN embedding is quite robust to the defects in images, and the embeddings of different facial features are independent of each other [24]. Based on their observation, they develop a mask-based local manipulation method. They find a plausible embedding for regions outside the mask and fill in reasonable semantic content in the masked pixels. The local region-of-interest editing results of their method are shown in Figure 10.

There are also some methods that can manipulate information in images other than semantics, *e.g.*, geometry, texture, and color. For example, some [24], [71] can change pose rotation for face manipulation, while others [95] can manipulate geometry (*e.g.*, zoom/shift/rotation), texture (*e.g.*, background blur/add grass/sharpness), and color (*e.g.*, lighting/saturation).

5.2 Image Generation

Several GAN inversion-based methods are proposed for image generation tasks, such as hairstyle transfer [149], few-shot semantic image synthesis [150], and infinite-resolution image synthesis [119]. Saha *et al.* [149] develop a photorealistic hairstyle transfer method by optimizing the extended latent space and the noise space of StyleGAN2 [17]. In [150], Endo *et al.*assume pixels sharing the same semantics have similar StyleGAN features to generate images and corresponding pseudosemantic masks from random noise in the latent space, and use a nearest-neighbor search for synthesis. This method integrates an encoder with the fixed StyleGAN generator and train the encoder with the pseudolabeled data in a supervised fashion to control the generator. Cheng *et al.* [151] propose a GAN inversion-based method for image inpainting and outpainting. A coordinate-conditioned generator is designed to synthesize patches to be concatenated for a full image. The latent codes, depending on the joint latent codes and their coordinates, synthesize the images overlapping with the input image. The optimal latent code for the available input patches is determined in the latent space of the trained patch-based generator during the outpainting stage. GAN inversion methods can be applied to interactive generation, *i.e.*, starting with strokes drawn by a user and generating natural images that best satisfy the user constraints. As shown in Figure 12, Zhu *et al.* [23] show that users can employ the brush tools to generate an image from scratch and then continually add more scribbles to refine the result. Abdal *et al.* [25] invert the StyleGAN to perform semantic local edits based on user scribbles. With this method, simple scribbles can be converted into photorealistic edits by embedding them into certain layers of StyleGAN. This application is helpful for existing interactive image processing tasks such as sketch-to-image generation [152], [153], [154] and sketch-based image retrieval [155], [156], which usually require densely labeled datasets.

5.3 Image Restoration

Suppose that $\hat{x}$ is obtained via $\hat{x} = \phi(x)$ during acquisition, where x is the distortion-free image, and ϕ is a degradation transform. Numerous image restoration tasks can be regarded as recovering x given $\hat{x}$. A common practice is to learn a mapping from $\hat{x}$ to x, which often requires task-specific training for different ϕ. Alternatively, GAN inversion can employ statistics of x stored in some prior and search in the space of x for an optimal x that best matches $\hat{x}$ by viewing $\hat{x}$ as partial observations of x. For example, Abdal *et al.* [24], [25] observe that StyleGAN embedding is quite robust to the defects in images, *e.g.*, masked regions. Based on that observation, they propose an inversion-based image inpainting method by embedding the source defective image into the early layers of the $\mathcal{W}^+$ space to predict the missing content and into the later layers to maintain color consistency. Pan *et al.* [28] claim that a fixed GAN generator is inevitably limited by the distribution of training data and its inversion cannot faithfully reconstruct unseen and complex images. Thus, they present a relaxed and more practical reconstruction formulation for capturing the statistics of natural images in a trained GAN model as do the prior methods, *i.e.*, the deep generative prior (DGP). Specifically, they reformulate (11) such that it allows the generator parameters to be fine-tuned on the target image on the fly:

$$\theta^*, \mathbf{z}^* = \arg\min_{\theta, \mathbf{z}} \ell(\hat{x}, \phi(G(\mathbf{z}; \theta))). \tag{29}$$

Their method performs visually better than or comparable to state-of-the-art methods in terms of colorization [157], inpainting [158], and superresolution [159]. While artifacts

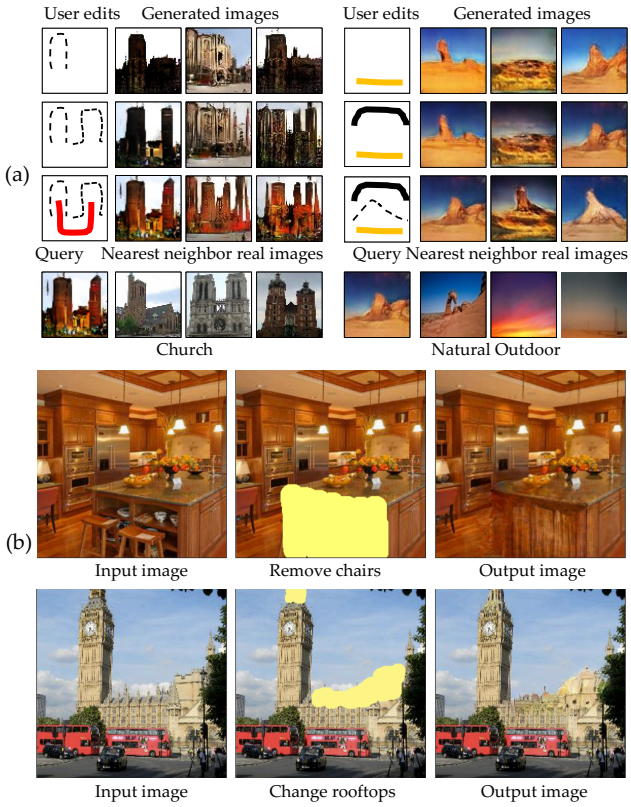

Fig. 12. **Illustration of interactive image generation using GAN inversion.** (a) illustrates results from [23]. The users are allowed to use the brush tools to generate an image from scratch and keep adding more scribbles or sketches for refinement. The last row shows the most similar real images to the generated images. Dashed line represents the sketch tool, and color scribble means the color brush. (b) is from GANPaint [47]. The brushes can draw semantically meaningful units like removing chairs or adding rooftops.

sometimes occur in synthesized face images by GAN models [14], [16], Shen *et al.* [21] show that the quality information encoded in the latent space can be used for restoration. The artifacts generated by PGGAN [14] can be corrected by moving the latent code toward the positive quality direction that is defined by a separating hyperplane using a linear SVM [160] (see Figure 11).

5.4 Image Interpolation

With GAN inversion, new results can be interpolated by morphing between corresponding latent vectors of given images. Given a well-trained GAN generator G and two target images x_A and x_B, morphing between them could naturally be performed by interpolating between their latent vectors $\mathbf{z}_A$ and $\mathbf{z}_B$. Typically, morphing between x_A and x_B can be obtained by applying linear interpolation [7], [28]:

$$\mathbf{z} = \lambda \mathbf{z}_A + (1-\lambda)\mathbf{z}_B, \lambda \in (0,1). \tag{30}$$

Such an operation can be found in [24], [59]. Moreover, in DGP [28], reconstructing two target images x_A and x_B would result in two generators G_{θ_A} and G_{θ_B}, respectively, and the corresponding latent vectors $\mathbf{z}_A$ and $\mathbf{z}_B$ since they also fine-tuned G. In this case, morphing between x_A and x_B can be achieved by linear interpolation of both the latent vectors and the generator parameters:

$$\begin{aligned} \mathbf{z} &= \lambda \mathbf{z}_A + (1-\lambda)\mathbf{z}_B, \\ \theta &= \lambda \theta_A + (1-\lambda)\theta_B, \ \lambda \in (0,1), \end{aligned} \tag{31}$$

and images can be generated with the new $\mathbf{z}$ and θ.

Algorithm 1: Local style transfer [25]

Input: images $x, y \in \mathbb{R}^{n \times m \times 3}$; masks M_b
Output: the embedded code $(\mathbf{w_o}, \mathbf{n_o})$
$(\mathbf{w}^*, \mathbf{n_i}) \leftarrow$ initialize();
$\mathbf{w_o} = W_l(M_b, M_b, 1, \mathbf{w}^*, \mathbf{n_i}, x)$
$\quad + M_{st}(1 - M_b, \mathbf{w}^*, \mathbf{n_i}, y)$;
$\mathbf{n_o} = Mk_n(M_b, \mathbf{w_o}, \mathbf{n_i}, x, G(\mathbf{w_o}))$;

Fig. 13. **Illustration of local style transfer [25].** From left to right: base image, masked region, style image, local style transfer result.

5.5 Style Transfer

To transfer the style from one image to another or mix styles of two images, numerous methods based on GAN inversion have been proposed. Given two latent codes, style transfer can be defined as a crossover operation [16], [24]. Abdal *et al.* [24] introduce two style transfer formulations: one is between the embedded stylized image and other face images (*e.g.*, a face photo and a face sketch), and the other is between embedded images from different classes (*e.g.*, a face photo and a nonface painting). The latent codes of the embedded content image are preserved for the first 9 layers (corresponding to a spatial resolution from 4^2 to 64^2), and the latent codes of the style image for the last 9 layers are overwritten (corresponding to a spatial resolution from 64^2 to 1024^2). In [25], Abdal *et al.* present a local style transfer method (see Algorithm 1 and Figure 13). An embedding algorithm as a gradient-based optimization is developed that iteratively updates an image starting from some initial latent code. The embedding is constructed with two spaces: the semantic $\mathbf{w} \in \mathcal{W}^+$ space and a noise $\mathbf{n} \in \mathcal{N}$ space encoding high-frequency details. Local style transfer modifies a region in the input image x to transform it to the style defined by a style reference image y. The first step is to embed the image into the $\mathcal{W}^+$ space to obtain the code $\mathbf{w}^*$ and the initializing noise code as $\mathbf{n_i}$. The second step is to apply the masked $\mathcal{W}^+$ optimization W_l along with the masked style transfer M_{st} using a blurred mask M_b. Finally, they perform the masked noise optimization Mk_n to output

the final image. The W_l function optimizes $\mathbf{w} \in \mathcal{W}^+$ and is given by

$$W_l(M_p, M_m, \mathbf{w_m}, \mathbf{w_i}, \mathbf{n_i}, x) = \arg\min_{\mathbf{w}} L_p(M_p, G(\mathbf{w}, \mathbf{n}), x) + \|M_m \odot (G(\mathbf{w}, \mathbf{n}) - x)\|, \tag{32}$$

where L_p denotes the perceptual loss [29], $\mathbf{w_i}$ and $\mathbf{n_i}$ are initial variable values, $\odot$ denotes the Hadamard product, and G is the StyleGAN generator. $\mathbf{w_m}$ indicates binary masks for $\mathcal{W}^+$ space. It contains 1 s for variables that should be optimized during an optimization process and 0 s for variables that should remain constant. The M_{st} function optimizes $\mathbf{w}$ to achieve a given target style defined by style image y, which is defined as

$$M_{st}(M_s, \mathbf{w_i}, \mathbf{n_i}, y) = \arg\min_{\mathbf{w}} L_s(M_s, G(\mathbf{w}, \mathbf{n}), y), \tag{33}$$

where L_s is the style loss [161]. The Mk_n function optimizes $\mathbf{n} \in N_s$ only, leaving $\mathbf{w}$ constant: $Mk_n(M, \mathbf{w_i}, \mathbf{n_i}, x, y) = \arg\min_{\mathbf{n}} \|M_m \odot (G(\mathbf{w}, \mathbf{n}) - x)\| + \|(1 - M_m) \odot (G(\mathbf{w}, \mathbf{n}) - y)\|$, where the noise space N_s has dimensions $\{\mathbb{R}^{4\times4}, \cdots, \mathbb{R}^{1024\times1024}\}$. Algorithm 1 shows the main steps of this method. The above authors use an alternating optimization strategy,*i.e.*, optimizing $\mathbf{w}$ while keeping $\mathbf{n}$ fixed and subsequently optimizing $\mathbf{n}$ while keeping $\mathbf{w}$ fixed. Aside from local style transfer, this method can also be used for image inpainting and local edits using scribbles by applying different spatial masks (M_s, M_p, M_m).

5.6 Compressive Sensing

Typically, compressive sensing can be formulated as reconstructing an unknown target signal or image $x \in \mathbb{R}^n$ from observations $\mathbf{y} \in \mathbb{R}^m$ of the form $\mathbf{y} = Ax + \mathbf{e}$, where $A \in \mathbb{R}^{m\times n}$ is a measurement matrix, and $\mathbf{e} \in \mathbb{R}^m$ represents stochastic noise. Since the number of measurements is much smaller than the ambient dimension of the signal, *i.e.*, $m \ll n$, the above inverse problem is an ill-posed problem. An alternative solution method is to obtain an estimate of $\hat{x}$ as the solution to the constrained optimization problem:

$$\begin{aligned} \hat{x} &= \arg\min_{x} \ell(y; Ax), \\ \text{s.t.} \quad & x \in \mathcal{S}, \end{aligned} \tag{34}$$

where ℓ is the loss function, and $\mathcal{S} \subseteq \mathcal{R}^n$ acts as *a prior*. To alleviate the ill-posed nature of the inversion problem (34) and make accurate recovery of x^* possible, several assumptions are commonly made, *e.g.*, the signal $x^* \in \mathcal{S}$ is sufficiently sparse and measurement matrix A satisfies certain algebraic conditions, such as the restricted isometry property (RSP) [162] or the restricted eigenvalue condition (REC) [163].

Applying GAN inversion to compressive sensing is accomplished by estimating the signal as $\hat{x} = G(\hat{\mathbf{z}})$, where $\hat{\mathbf{z}}$ is obtained by minimizing the nonconvex cost function:

$$f(\mathbf{z}) = \|\mathbf{y} - AG(\mathbf{z})\|_2^2. \tag{35}$$

Bora *et al.* [164] propose to solve (35) using backpropagation and standard gradient-based optimization. Hussein *et al.* [165] handle the limited representation capabilities of the generators by making them have image-adaptive (IA) properties using internal learning at test time. Instead of recovering the latent signal x as $\hat{x} = G(\hat{\mathbf{z}})$, where $G(\cdot)$ is a well-trained generator, they simultaneously optimize $\mathbf{z}$ and the parameters of the generator, denoted as θ, by minimizing the cost function:

$$f(\theta, \mathbf{z}) = \|y - AG_\theta(\mathbf{z})\|_2^2. \tag{36}$$

In [166], Shah *et al.* present a projected gradient descent (PGD)-based method to solve (35). The first step of this approach is to update the gradient descent at the t-th iteration to obtain w_t: $w_t \leftarrow x_t + \eta A^\top(y - Ax_t)$, where η denotes the learning rate, and the second step is to use G to find the target image that matches the current estimate w_t by defining the projection operator $\mathcal{P}_G$: $\mathcal{P}_G(w_t) = G(\arg\min_{\mathbf{z}} \|w_t - G(\mathbf{z})\|)$. Based on [166], Raj *et al.* [48] replace the iterative scheme in the inner loop with a learning-based approach, as it often performs better and does not fall into local optima.

5.7 Semantic Diffusion

Semantic image diffusion is an image editing task that inserts the target face to the context and makes them compatible, as illustrated in Figure 14. It can be seen as a variant of image harmonization [10], [167], [168]. Zhu *et al.* [22] use their in-domain inversion method for semantic diffusion, which keeps the characteristics of the target image (*e.g.*, face identity) and adapts to the context information at the same time. Moreover, Xu *et al.* [68] copy some patches (*e.g.*, bed and window) onto a bedroom image and feed the stitched image into the proposed encoder for feature extraction. The extracted features are then visualized via the generator for image harmonization.

5.8 Category Transfer

In Section 5.3, we demonstrate that DGP [28], a method proposed by Pan *et al.*, can be used to restore images of different degradations. Their method can also be used to transfer the object category of given images by tweaking the class condition during the reconstruction. The lower right corner of Figure 9 shows an example that transfers the dog to various other categories without changing the pose, shape, or background.

5.9 Adversarial Defense

Adversarial attack methods [169], [170], [171], [172] aim at fooling a CNN classifier by adding a certain perturbation Δx to a target image x. In contrast, adversarial defense [173], [174] aims at preventing the model from being fooled by attackers. When considering the degradation transform of an adversarial attack as $\phi(x) = x + \Delta x$, where Δx is the perturbation generated by the attacker, we can use the inversion methods demonstrated in Section 5.3 for adversarial defense. For example, DGP [28] directly reconstructs the adversarial image $\hat{x}$ and stops the reconstruction when the MSE loss reaches a certain threshold value.

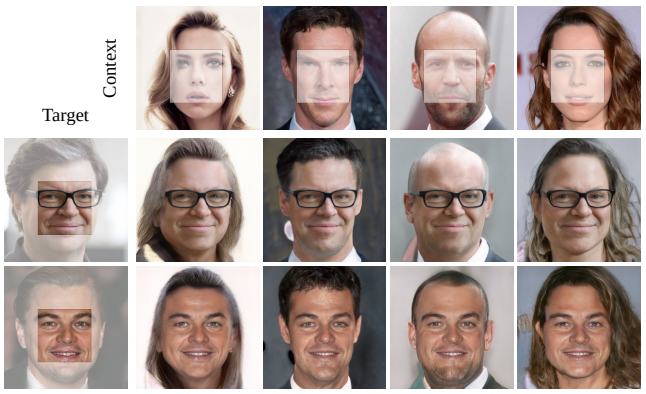

Fig. 14. **Semantic diffusion results using the in-domain GAN inversion method [22]**. Target images in the first column are naturally diffused into context images in the first row with the identify preserved.

5.10 3D Reconstruction

For 3D data, Pan *et al.* [139] and Zhang *et al.* [140] propose 3D shape reconstruction from single images and point cloud completion based on GAN inversion. Given an image generated by GAN, starting with an initial ellipsoid 3D object shape, Pan *et al.* [139] first render a number of unnatural images with various randomly sampled viewpoints and lighting conditions (called pseudosamples). By reconstructing them with the GAN, these pseudosamples could guide the original image toward the sampled viewpoints and lighting conditions in the GAN manifold, producing a number of natural-looking images (called projected samples). These projected samples could be adopted as the ground truth of the differentiable rendering process to refine the prior 3D shape. Instead of using existing 2D GANs trained on images, Zhang *et al.* [140] first train a generator G on 3D shapes in the form of point clouds. Latent codes are used by the pretrained generator to produce complete shapes. Given a partial shape, they look for a target latent vector $\mathbf{z}$ and fine-tune the parameters θ of G that best reconstruct the complete shape via gradient descent.

5.11 Image Understanding

A few methods exploit the representations of trained GAN models and leverage these representations for semantic segmentation and alpha matting [141], [142]. Tritrong *et al.* [141] first embed an image into the latent space for the latent z and feed it into the generator with multiple activation maps. These maps are upsampled and concatenated along the channel dimension to form the desired representation. A segmentation module is trained with a few manually annotated images and extracted representations. During inference, the representation is extracted from a test image and fed into the segmenter to obtain a segmentation map. In [142], two pretrained generators, an alpha network and a discriminator are used for matting. One generator $\mathcal{G}(\mathbf{z})$ is responsible for generating foreground images, and the other generator $\mathcal{G}_{\text{bg}}(\mathbf{z}')$ attends to the background. The alpha network is used to predict a mask $\mathcal{A}(\mathbf{z}) \odot \mathcal{G}(\mathbf{z})$ for image matting. The composite image can be obtained by mixing background and foreground using $\mathcal{A}(\mathbf{z}) \odot \mathcal{G}(\mathbf{z}) + (1 - \mathcal{A}(\mathbf{z})) \odot \mathcal{G}_{\text{bg}}(\mathbf{z}')$ that the discriminator $\mathcal{D}$ cannot distinguish from the real images. During training, the two generators are frozen, and only the alpha network and the discriminator are trained by adversarial learning.

5.12 Multimodal Learning

For multimodal learning, several recent studies have focused on language-driven image generation and manipulation using StyleGAN. Xia *et al.* [133] propose a novel unified framework for both text-to-image generation and text-guided image manipulation tasks by training an encoder to map texts into the latent space of StyleGAN and perform style-mixing to produce diverse results. In [134], Wang *et al.*propose a similar idea but introduce the cycle-consistency training during inversion to learn more robust and consistent inverted latent codes. On the other hand, a few methods [135], [143], [144] first obtain the latent code of a given image and find the target latent code of desired attributes with the guidance of some powerful pretrained language models, *e.g.* CLIP [175] or ALIGN [176]. Logacheva *et al.* [177] present a generative model for landscape animation videos based on StyleGAN inversion.

5.13 Medical Imaging

GAN inversion technique has recently been introduced to medical applications [178]. GAN inversion techniques have recently been introduced to medical applications [178]. These methods [145], [147] are used for data augmentation, where publicly available medical datasets are often outdated, limited, or inadequately annotated. Typically, these methods train the GAN models on domain-specific medical image datasets, *e.g.* Computed Tomography (CT) or Magnetic Resonance (MR), and use existing GAN inversion methods for inversion and manipulation. Fetty *et al.* [147] present a method based on the StyleGAN model [24] in which CT or MR images with desired attributes can be synthesized by traversing points in the latent latent space (see Section 4.4) or style mixing. To synthesize controllable medical images, Ren *et al.* [145] use the domain-specific GAN model by Zhu *et al.* [22] to generate mammograms with desired shape and texture for psychophysical experiments. Overall, these methods based on GAN inversion achieve better interpretability and controllability in synthesizing medical images.

6 Challenges and Future Directions

Theoretical Understanding. Despite its success in application, there is still a lack of theoretical understanding of GAN inversion. GAN inversion can be seen as a nonlinear equivalent to the dimensionality reduction commonly performed by PCA, as proposed by [138]. Nonlinear structure in data can be represented compactly, and the induced geometry necessitates the use of nonlinear statistical tools [179], Riemannian manifolds, and locally linear methods. Well-established theories in related areas can facilitate better theoretical understanding of GAN inversion in terms of the weights (parameters) or latent space of neural networks from different perspectives. For example, we can formulate GAN inversion as decomposing signals into components (matrix factorization problems) and use nonlinear

factor analysis (FA) [180], independent component analysis (ICA) [181], and latent Dirichlet allocation (LDA) [182], [183] to decompose the network weights and to find interpretable directions of latent space.

Inversion Type. In addition to GAN inversion, some methods have been developed to invert generative models based on the encoder-decoder architecture. The IIN method [184] learns invertible disentangled interpretations of variational autoencoders (VAEs) [185]. Zhu *et al.* [186] develop the latently invertible autoencoder method to learn a disentangled representation of face images from which contents can be edited based on attributes. The LaDDer approach [187] uses a meta-embedding based on a generative prior (including an additive VAE and a mixture of hyperpriors) to project the latent space of a well-trained VAE to a lower-dimensional latent space, where multiple VAE models are used to form a hierarchical representation. It is beneficial to explore how to combine GAN inversion and encoder-decoder inversion so that we can exploit the best of both worlds.

Domain Generalization. As discussed in Section 5, GAN inversion has been proven to be effective in cross-domain applications such as style transfer and image restoration, which indicates that pretrained models have learned domain-agnostic features. The images from different domains can be inverted into the same latent space from which effective metrics can be derived. Multitask methods have been developed to collaboratively exploit visual cues, such as image restoration and image segmentation [188] or semantic segmentation and depth estimation [189], [190], within the GAN framework. It is challenging but worthwhile to develop effective and consistent methods to invert the intermediate shared representations so that we can tackle different vision tasks under a unified framework.

Scene Representation. GAN inversion methods [71], [95] can manipulate geometry (*e.g.*, zoom, shift, and rotate), texture (*e.g.*, background blur and sharpness) and color (*e.g.*, lighting and saturation). This ability indicates the GAN models pretrained on large-scale datasets have learned some physical information from real-world scenes. Implicit neural representation learning [191], [192], [193], a recent trend in the 3D community, learns implicit functions for 3D shapes or scenes and enables control of scene properties such as illumination, camera parameters, pose, geometry, appearance, and semantic structure. It has been used for volumetric performance capture [194], [195], [196], novel-view synthesis [197], [197], face shape generation [198], object modeling [199], [200], and human reconstruction [201], [202], [203], [204]. The recent StyleRig method [53] is trained based on the semantic parameters of the 3D morphable model (3DMM) [205] and the input of StyleGAN [16]. It opens an interesting research direction to invert such implicit representations of a pretrained GAN for 3D reconstruction, *e.g.*, using StyleGAN [16] for human face modeling or time-lapse video generation.

Precise Control. GAN inversion can be used to find directions for image manipulation while preserving the identity and other attributes [21], [71]. However, there is also some tuning required to achieve the desired granularity of precise fine-grained control, *e.g.*, gaze redirection [7], [206], relighting [207], [208], and continuous view control [209]. These tasks require fine-grained control, *i.e.*, 1° of camera view or gaze direction. Current GAN inversion methods are incapable of tackling the situation, which indicates that more efforts are necessary to accomplish these tasks with ease, such as creating more disentangled latent spaces and discovering more interpretable directions.

Multimodal Inversion. The existing GAN inversion methods primarily concern images. However, recent advances in generative models are beyond the image domain, such as the GPT-3 language model [210] and WaveNet [211] for audio synthesis. Trained on diverse large-scale datasets, these sophisticated deep neural networks have been proven to be capable of representing an extensive range of different contents, styles, sentiments, and topics. Applying GAN inversion techniques on these different modalities could provide a novel perspective for tasks such as language style transfer. Furthermore, there are also GAN models for multimodality generation or translation [212], [213], [214]. It is a promising direction to invert such GAN models as multimodal representations to create novel kinds of content, behavior, and interaction.

Evaluation Metrics. The perceptual quality metrics, which can better evaluate photorealistic and diverse images or identity consistent with the original image, remain to be explored. Furthermore, the evaluations mostly concentrate on photorealism or judge if the distribution of generated images is consistent with the real images with regard to classification [26] or segmentation [95] accuracy using models trained for real images. However, there is still a lack of effective assessment tools to evaluate the difference between the predicted results and the expected outcome or to measure the inverted latent codes more directly.

7 Conclusions

Deep generative models such as GANs learn to model a rich set of semantic and physical rules about the target distribution by generating the data. GAN inversion reveals the rules encoded in the network or how a rule could be exploited to manipulate images. In this paper, we present a comprehensive overview of GAN inversion methods with an emphasis on algorithms and applications. We summarize the important features of GAN latent space and models and then introduce four kinds of GAN inversion methods and their key characteristics. We then introduce several fascinating applications of GAN inversion, including image manipulation, image restoration, image interpolation, style transfer, and compressive sensing. Ultimately, we discuss some challenges and future directions for GAN inversion, and we hope this paper will inspire future research to solve these challenges.

References

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," in *NeurIPS*, 2014. 1, 2, 3

[2] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT Press, 2016. 1

[3] Q. Mao, H.-Y. Lee, H.-Y. Tseng, S. Ma, and M.-H. Yang, "Mode seeking generative adversarial networks for diverse image synthesis," in *CVPR*, 2019. 1

[4] H.-Y. Lee, H.-Y. Tseng, J.-B. Huang, M. Singh, and M.-H. Yang, "Diverse image-to-image translation via disentangled representations," in *ECCV*, 2018. 1

[5] X. Huang, M. Liu, S. J. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," in *ECCV*, 2018, pp. 179–196. 1

[6] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional GANs," in *CVPR*, 2018. 1

[7] W. Xia, Y. Yang, J.-H. Xue, and W. Feng, "Controllable continuous gaze redirection," in *ACM MM*, 2020. 1, 14, 17

[8] B. Li, X. Qi, T. Lukasiewicz, and P. H. Torr, "Manigan: Text-guided image manipulation," in *CVPR*, 2020. 1

[9] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep CNN for image denoising," *TIP*, 2017. 1

[10] Y.-H. Tsai, X. Shen, Z. Lin, K. Sunkavalli, X. Lu, and M.-H. Yang, "Deep image harmonization," in *CVPR*, 2017. 1, 15

[11] X. Xu, D. Sun, J. Pan, Y. Zhang, H. Pfister, and M.-H. Yang, "Learning to super-resolve blurry face and text images," in *ICCV*, 2017. 1

[12] C. Ma, C.-Y. Yang, X. Yang, and M.-H. Yang, "Learning a no-reference quality metric for single-image super-resolution," *CVIU*, 2017. 1

[13] Y. Li, C. Fang, J. Yang, Z. Wang, X. Lu, and M.-H. Yang, "Flow-grounded spatial-temporal video prediction from still images," in *ECCV*, 2018. 1

[14] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of GANs for improved quality, stability, and variation," in *ICLR*, 2018. 1, 2, 3, 4, 5, 7, 8, 13, 14

[15] A. Brock, J. Donahue, and K. Simonyan, "Large scale GAN training for high fidelity natural image synthesis," in *ICLR*, 2019. 1, 3, 5, 7, 11

[16] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *CVPR*, 2019, pp. 4401–4410. 1, 2, 3, 4, 5, 6, 7, 8, 12, 14, 17

[17] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of StyleGAN," in *CVPR*, 2020. 1, 3, 4, 7, 12, 13

[18] D. Bau, H. Strobelt, W. Peebles, J. Wulff, B. Zhou, J.-Y. Zhu, and A. Torralba, "Semantic photo manipulation with a generative image prior," *TOG*, vol. 38, no. 4, p. 59, 2019. 1

[19] L. Goetschalckx, A. Andonian, A. Oliva, and P. Isola, "Ganalyze: Toward visual definitions of cognitive image properties," in *ICCV*, 2019. 1, 11

[20] A. Jahanian, L. Chai, and P. Isola, "On the "steerability" of generative adversarial networks," in *ICLR*, 2020. 1, 10, 11

[21] Y. Shen, J. Gu, X. Tang, and B. Zhou, "Interpreting the latent space of GANs for semantic face editing," in *CVPR*, 2020. 1, 5, 10, 11, 12, 13, 14, 17

[22] J. Zhu, Y. Shen, D. Zhao, and B. Zhou, "In-domain gan inversion for real image editing," in *ECCV*, 2020. 1, 2, 3, 8, 9, 10, 12, 15, 16

[23] J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros, "Generative visual manipulation on the natural image manifold," in *ECCV*, 2016. 1, 2, 3, 6, 7, 8, 13, 14

[24] R. Abdal, Y. Qin, and P. Wonka, "Image2StyleGAN: How to embed images into the StyleGAN latent space?" in *ICCV*, 2019. 1, 2, 3, 5, 7, 12, 13, 14, 16

[25] ——, "Image2StyleGAN++: How to edit the embedded images?" in *CVPR*, 2020. 1, 2, 3, 5, 7, 10, 12, 13, 14

[26] D. Bau, J.-Y. Zhu, J. Wulff, W. Peebles, H. Strobelt, B. Zhou, and A. Torralba, "Seeing what a gan cannot generate," in *ICCV*, 2019, pp. 4502–4511. 1, 3, 4, 8, 9, 17

[27] M. Huh, R. Zhang, J.-Y. Zhu, S. Paris, and A. Hertzmann, "Transforming and projecting images into class-conditional generative networks," in *ECCV*, 2020. 1, 2, 3, 7

[28] X. Pan, X. Zhan, B. Dai, D. Lin, C. C. Loy, and P. Luo, "Exploiting deep generative prior for versatile image restoration and manipulation," in *ECCV*, 2020. 1, 3, 10, 12, 13, 14, 15

[29] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *ECCV*, 2016, pp. 694–711. 2, 7, 15

[30] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *CVPR*, 2018, pp. 586–595. 2, 4, 8

[31] E. Richardson, Y. Alaluf, O. Patashnik, Y. Nitzan, Y. Azar, S. Shapiro, and D. Cohen-Or, "Encoding in style: a stylegan encoder for image-to-image translation," in *CVPR*, 2021. 2, 3, 7, 8, 10

[32] T. Wei, D. Chen, W. Zhou, J. Liao, W. Zhang, L. Yuan, G. Hua, and N. Yu, "A simple baseline for stylegan inversion," *arXiv preprint arXiv:2104.07661*, 2021. 2, 3, 7, 8, 10

[33] Y. Alaluf, O. Patashnik, and D. Cohen-Or, "Restyle: A residual-based stylegan encoder via iterative refinement," *arXiv preprint arXiv:2104.02699*, 2021. 2, 3, 7, 10

[34] J. Donahue, P. Krähenbühl, and T. Darrell, "Adversarial feature learning," *arXiv preprint arXiv:1605.09782*, 2016. 2, 7

[35] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville, "Adversarially learned inference," *arXiv preprint arXiv:1606.00704*, 2016. 2, 7

[36] O. Lang, Y. Gandelsman, M. Yarom, Y. Wald, G. Elidan, A. Hassidim, W. T. Freeman, P. Isola, A. Globerson, M. Irani *et al.*, "Explaining in style: Training a gan to explain a classifier in stylespace," *arXiv preprint arXiv:2104.13369*, 2021. 2, 7

[37] S. Menon, A. Damian, S. Hu, N. Ravi, and C. Rudin, "PULSE: self-supervised photo upsampling via latent space exploration of generative models," in *CVPR*, 2020, pp. 2434–2442. 2, 6

[38] G. Daras, J. Dean, A. Jalal, and A. G. Dimakis, "Intermediate layer optimization for inverse problems using deep generative models," in *ICML*, 2021. 2

[39] V. A. Kelkar and M. A. Anastasio, "Prior image-constrained reconstruction using style-based generative models," *arXiv preprint arXiv:2102.12525*, 2021. 2

[40] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *ICLR*, 2016. 3, 5, 7

[41] A. Yu and K. Grauman, "Fine-grained visual comparisons with local learning," in *CVPR*, 2014, pp. 192–199. 3

[42] B. Zhou, À. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *NeurIPS*, 2014, pp. 487–495. 3

[43] F. Yu, Y. Zhang, S. Song, A. Seff, and J. Xiao, "Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop," *arXiv preprint arXiv:1506.03365*, 2015. 3, 4, 8

[44] A. Creswell and A. A. Bharath, "Inverting the generator of a generative adversarial network," *TNNLS*, 2018. 3, 7

[45] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *NeurIPS*, 2017, pp. 5767–5777. 3

[46] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *ICCV*, 2015. 3, 4

[47] D. Bau, H. Strobelt, W. Peebles, J. Wulff, B. Zhou, J. Zhu, and A. Torralba, "Semantic photo manipulation with a generative image prior," *TOG*, vol. 38, no. 4, 2019. 3, 8, 14

[48] A. Raj, Y. Li, and Y. Bresler, "Gan-based projector for faster recovery with convergence guarantees in linear inverse problems," in *ICCV*, 2019, pp. 5602–5611. 3, 15

[49] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," in *ICML*, 2019. 3

[50] Y. LeCun, "The mnist database of handwritten digits," *http://yann. lecun. com/exdb/mnist/*, 1998. 3, 4

[51] J. Gu, Y. Shen, and B. Zhou, "Image processing using multi-code gan prior," in *CVPR*, 2020. 3, 12

[52] C. Edo, B. Raja, P. Bob, and S. Sabine, "Editing in style: Uncovering the local semantics of gans," in *CVPR*, 2020. 3, 8

[53] A. Tewari, M. Elgharib, G. Bharaj, F. Bernard, H.-P. Seidel, P. Pérez, M. Zöllhofer, and C. Theobalt, "Stylerig: Rigging stylegan for 3d control over portrait images," in *CVPR*, 2020. 3, 6, 7, 17

[54] G. Daras, A. Odena, H. Zhang, and A. G. Dimakis, "Your local gan: Designing two dimensional local attention mechanisms for generative models," in *CVPR*, 2020, pp. 14 531–14 539. 3, 10

[55] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *IJCV*, vol. 115, no. 3, pp. 211–252, 2015. 3, 4

[56] V. Yuri and K. Vladimir, Ivashkin ang Evgeny, "Stylegan2 distillation for feed-forward image manipulation," in *ECCV*, 2020. 3, 12

[57] R. Anirudh, J. J. Thiagarajan, B. Kailkhura, and P. Bremer, "Mimicgan: Robust projection onto image manifolds with corruption mimicking," *IJCV*, vol. 128, no. 10, pp. 2459–2477, 2020. 3

[58] A. Tewari, M. Elgharib, M. BR, F. Bernard, H.-P. Seidel, P. Pérez, M. Zöllhofer, and C. Theobalt, "Pie: Portrait image embedding for semantic control," *TOG*, vol. 39, no. 6, December 2020. 3

[59] Y. Nitzan, A. Bermano, Y. Li, and D. Cohen-Or, "Face identity disentanglement via latent space mapping," *TOG*, 2020. 3, 4, 5, 12, 14

[60] A. Aberdam, D. Simon, and M. Elad, "When and how can deep generative models be inverted?" *arXiv preprint arXiv:2006.15555*, 2020. 3, 6, 9

[61] S. Guan, Y. Tai, B. Ni, F. Zhu, F. Huang, and X. Yang, "Collaborative learning for faster stylegan embedding," *arXiv preprint arXiv:2007.01758*, 2020. 3, 6, 7, 8, 12

[62] B.-C. Chen, C.-S. Chen, and W. H. Hsu, "Cross-age reference coding for age-invariant face recognition and retrieval," in *ECCV*, 2014, pp. 768–783. 3, 4

[63] Y. Liu, Q. Li, Z. Sun, and T. Tan, "Style intervention: How to achieve spatial disentanglement with style-based generators?" *arXiv preprint arXiv:2011.09699*, 2020. 3, 6, 12

[64] A. Cherepkov, A. Voynov, and A. Babenko, "Navigating the gan parameter space for semantic image editing," in *CVPR*, 2021. 3, 10, 11

[65] P. Zhuang, O. Koyejo, and A. G. Schwing, "Enjoy your editing: Controllable gans for image editing via latent space navigation," in *ICLR*, 2021. 3, 10

[66] L. Chai, J. Wulff, and P. Isola, "Using latent space regression to analyze and leverage compositionality in gans." in *ICLR*, 2021. 3, 10

[67] Z. Wu, D. Lischinski, and E. Shechtman, "Stylespace analysis: Disentangled controls for stylegan image generation," in *CVPR*, 2021. 3, 6, 12

[68] Y. Xu, Y. Shen, J. Zhu, C. Yang, and B. Zhou, "Generative hierarchical features from synthesizing images," in *CVPR*, 2021. 3, 7, 12, 15

[69] H.-P. Wang, N. Yu, and M. Fritz, "Hijack-gan: Unintended-use of pretrained, black-box gans," in *CVPR*, 2021. 3

[70] L. Chai, J.-Y. Zhu, E. Shechtman, P. Isola, and R. Zhang, "Ensembling with deep generative views." in *CVPR*, 2021. 3

[71] A. Rameen, Z. Peihao, M. Niloy, and W. Peter, "Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows," *TOG*, 2021. 3, 5, 10, 11, 13, 17

[72] Y. Alaluf, O. Patashnik, and D. Cohen-Or, "Only a matter of style: Age transformation using a style-based regression model," *arXiv preprint arXiv:2102.02754*, 2021. 3, 10

[73] O. Tov, Y. Alaluf, Y. Nitzan, O. Patashnik, and D. Cohen-Or, "Designing an encoder for stylegan image manipulation," *TOG*, 2021. 3, 10

[74] Y. Xu, Y. Du, W. Xiao, X. Xu, and S. He, "From continuity to editability: Inverting gans with consecutive images," in *ICCV*, 2021. 3

[75] S. R. Livingstone and F. A. Russo, "The ryerson audio-visual database of emotional speech and song (ravdess): A dynamic, multimodal set of facial and vocal expressions in north american english," *PloS one*, vol. 13, no. 5, p. e0196391, 2018. 3

[76] P. Zhu, R. Abdal, Y. Qin, J. Femiani, and P. Wonka, "Improved stylegan embedding: Where are the good latents?" *arXiv preprint arXiv:2012.09036*, 2020. 3, 6

[77] Y. Choi, Y. Uh, J. Yoo, and J.-W. Ha, "Stargan v2: Diverse image synthesis for multiple domains," in *CVPR*, 2020. 3, 4

[78] T. Karras, M. Aittala, J. Hellsten, S. Laine, J. Lehtinen, and T. Aila, "Training generative adversarial networks with limited data," in *NeurIPS*, 2020. 3

[79] J. Yanghua, Z. Jiakai, L. Minjun, T. Yingtao, Z. Huachun, and F. Zhihao, "Towards the high-quality anime characters generation with generative adversarial networks," in *NeurIPS Workshop*, 2017. 3, 4

[80] M.-Y. Liu, X. Huang, A. Mallya, T. Karras, T. Aila, J. Lehtinen, and J. Kautz., "Few-shot unsueprvised image-to-image translation," in *ICCV*, 2019. 3

[81] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *ICCV*, 2017. 3, 5, 6, 7

[82] T. Karras, M. Aittala, S. Laine, E. Härkönen, J. Hellsten, J. Lehtinen, and T. Aila, "Alias-free generative adversarial networks," *arXiv preprint arXiv:2106.12423*, 2021. 3

[83] A. Gabbay and Y. Hoshen, "Style generator inversion for image enhancement and animation," *arXiv preprint arXiv:1906.11880*, 2019. 4

[84] P. Zhu, R. Abdal, Y. Qin, and P. Wonka, "Sean: Image synthesis with semantic region-adaptive normalization," in *CVPR*, 2020, pp. 5104–5113. 4

[85] Z. Zhu, Z. Xu, A. You, and X. Bai, "Semantically multi-modal image synthesis," in *CVPR*, 2020, pp. 5467–5476. 4

[86] C.-H. Lee, Z. Liu, L. Wu, and P. Luo, "Maskgan: Towards diverse and interactive facial image manipulation," in *CVPR*, 2020. 4

[87] Y. Zhang, Z. Yin, Y. Li, G. Yin, J. Yan, J. Shao, and Z. Liu, "Celeba-spoof: Large-scale face anti-spoofing dataset with rich annotations," in *ECCV*, 2020. 4

[88] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017. 4

[89] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," *Technical Report*, 2009. 4

[90] A. Yu and K. Grauman, "Semantic jitter: Dense supervision for visual comparisons via synthetic images," in *ICCV*, 2017. 4

[91] Z. Liu, S. Yan, P. Luo, X. Wang, and X. Tang, "Fashion landmark detection in the wild," in *ECCV*, 2016, pp. 229–245. 4

[92] Z. Liu, P. Luo, S. Qiu, X. Wang, and X. Tang, "Deepfashion: Powering robust clothes recognition and retrieval with rich annotations," in *CVPR*, 2016, pp. 1096–1104. 4

[93] Y. Ge, R. Zhang, L. Wu, X. Wang, X. Tang, and P. Luo, "A versatile benchmark for detection, pose estimation, segmentation and re-identification of clothing images," in *CVPR*, 2019. 4

[94] N. Naik, J. Philipoom, R. Raskar, and C. Hidalgo, "Streetscore-predicting the perceived safety of one million streetscapes," in *CVPR Workshops*, 2014, pp. 779–785. 4

[95] A. Voynov and A. Babenko, "Unsupervised discovery of interpretable directions in the gan latent space," *ICML*, 2020. 4, 7, 10, 11, 12, 13, 17

[96] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training GANs," in *NeurIPS*, 2016. 4

[97] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *CVPR*, 2016, pp. 2818–2826. 4

[98] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*, 2009, pp. 248–255. 4

[99] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs trained by a two time-scale update rule converge to a local Nash equilibrium," in *NeurIPS*, 2017. 4

[100] J. Rabin, G. Peyré, J. Delon, and M. Bernot, "Wasserstein barycenter and its application to texture mixing," in *International Conference on Scale Space and Variational Methods in Computer Vision*, 2011, pp. 435–446. 4

[101] A. Geitgey, "Github - face recognition," 2020. [Online]. Available: https://github.com/ageitgey/face_recognition 5

[102] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *TIP*, vol. 13, 2004. 5

[103] D. E. King, "Dlib-ml: A machine learning toolkit," *JMLR*, 2009. 5

[104] J. Xu, H. Xu, B. Ni, X. Yang, X. Wang, and T. Darrell, "Hierarchical style-based networks for motion synthesis," in *ECCV*, 2020. 6

[105] G. Perarnau, J. Van De Weijer, B. Raducanu, and J. M. Álvarez, "Invertible conditional gans for image editing," *arXiv preprint arXiv:1611.06355*, 2016. 6

[106] D. Bau, J.-Y. Zhu, J. Wulff, W. Peebles, H. Strobelt, B. Zhou, and A. Torralba, "Inverting layers of a large generator," in *ICLR Workshop*, vol. 2, no. 3, 2019, p. 4. 6, 7, 8

[107] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014. 7

[108] S. Pidhorskyi, D. A. Adjeroh, and G. Doretto, "Adversarial latent autoencoders," in *CVPR*, 2020. 7

[109] Z. C. Lipton and S. Tripathi, "Precise recovery of latent vectors from generative adversarial networks," *arXiv preprint arXiv:1702.04782*, 2017. 7

[110] F. Ma, U. Ayaz, and S. Karaman, "Invertibility of convolutional generative networks from partial measurements," in *NeurIPS*, 2018, pp. 9651–9660. 7

[111] A. Ramesh, Y. Choi, and Y. LeCun, "A spectral regularizer for unsupervised disentanglement," *arXiv preprint arXiv:1812.01161*, 2018. 7

[112] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015. 7

[113] D. C. Liu and J. Nocedal, "On the limited memory bfgs method for large scale optimization," *Mathematical programming*, vol. 45, no. 1-3, pp. 503–528, 1989. 7

[114] S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth, "Hybrid monte carlo," *Physics letters B*, 1987. 7

[115] N. Hansen and A. Ostermeier, "Completely derandomized self-adaptation in evolution strategies. evolutionary computation," *Evoluaionary Computation*, 2001. 7

[116] J. Rapin and O. Teytaud, "Nevergrad - A gradient-free optimization platform," https://GitHub.com/FacebookResearch/Nevergrad, 2018. 7

[117] D. Bau, J.-Y. Zhu, H. Strobelt, B. Zhou, J. B. Tenenbaum, W. T. Freeman, and A. Torralba, "Gan dissection: Visualizing and understanding generative adversarial networks," in *ICLR*, 2019. 8

[118] Y. Zhu, Q. Li, J. Wang, C. Xu, and Z. Sun, "One shot face swapping on megapixels," in *CVPR*, 2021. 8

[119] C. H. Lin, Y.-C. Cheng, H.-Y. Lee, S. Tulyakov, and M.-H. Yang, "Infinitygan: Towards infinite-resolution image synthesis," *arXiv preprint arXiv:2104.03963*, 2021. 8, 13

[120] Q. Lei, A. Jalal, I. S. Dhillon, and A. G. Dimakis, "Inverting deep generative models, one layer at a time," in *NeurIPS*, 2019. 9

[121] W. Huang, P. Hand, R. Heckel, and V. Voroninski, "A provably convergent scheme for compressive sensing under random generative priors," *arXiv preprint arXiv:1812.04176*, 2018. 9

[122] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *ICML*, 2010. 9

[123] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *ICML*, 2013. 9

[124] J. Ren, P. J. Liu, E. Fertig, J. Snoek, R. Poplin, M. Depristo, J. Dillon, and B. Lakshminarayanan, "Likelihood ratios for out-of-distribution detection," in *NeurIPS*, 2019, pp. 14 707–14 718. 10

[125] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," *arXiv preprint arXiv:1610.02136*, 2016. 10

[126] K. Lee, K. Lee, H. Lee, and J. Shin, "A simple unified framework for detecting out-of-distribution samples and adversarial attacks," in *NeurIPS*, 2018, pp. 7167–7177. 10

[127] Y. Han, J. Yang, and Y. Fu, "Disentangled face attribute editing via instance-aware latent space search," in *IJCAI*, 2021. 10

[128] S. Nurit, B. Ron, and M. Tomer, "Gan steerability without optimization," in *ICLR*, 2021. 10, 11

[129] Y. Shen and B. Zhou, "Closed-form factorization of latent semantics in gans," in *CVPR*, 2021. 10, 11

[130] O. Chelnokova, B. Laeng, M. Eikemo, J. Riegels, G. Løseth, H. Maurud, F. Willoch, and S. Leknes, "Rewards of beauty: the opioid system mediates social motivation in humans," *Molecular psychiatry*, vol. 19, no. 7, pp. 746–747, 2014. 10

[131] R. Courset, M. Rougier, R. Palluel-Germain, A. Smeding, J. M. Jonte, A. Chauvin, and D. Muller, "The Caucasian and North African French Faces (CaNAFF): A face database," *International Review of Social Psychology*, vol. 31, no. 1, 2018. 10

[132] R. Yi, Y.-J. Liu, Y.-K. Lai, and P. L. Rosin, "ApdrawingGAN: Generating artistic portrait drawings from face photos with hierarchical gans," in *CVPR*, 2019, pp. 10 743–10 752. 10

[133] W. Xia, Y. Yang, J.-H. Xue, and B. Wu, "Tedigan: Text-guided diverse image generation and manipulation," in *CVPR*, 2021. 10, 12, 16

[134] H. Wang, G. Lin, S. C. H. Hoi, and C. Miao, "Cycle-consistent inverse gan for text-to-image synthesis," in *ACM MM*, 2021. 10, 12, 16

[135] O. Patashnik, Z. Wu, E. Shechtman, D. Cohen-Or, and D. Lischinski, "Styleclip: Text-driven manipulation of stylegan imagery," *arXiv preprint arXiv:2103.17249*, 2021. 10, 12, 16

[136] A. Plumerault, H. L. Borgne, and C. Hudelot, "Controlling generative models with continuous factors of variations," *ICLR*, 2020. 11

[137] Y.-D. Lu, H.-Y. Lee, H.-Y. Tseng, and M.-H. Yang, "Unsupervised discovery of disentangled manifolds in gans," *arXiv preprint arXiv:2011.11842*, 2020. 11

[138] H. Erik, H. Aaron, L. Jaakko, and P. Sylvain, "Ganspace: Discovering interpretable gan controls," in *NeurIPS*, 2020. 11, 16

[139] X. Pan, B. Dai, Z. Liu, C. C. Loy, and P. Luo, "Do 2d gans know 3d shape? unsupervised 3d shape reconstruction from 2d image gans," in *ICLR*, 2021. 12, 16

[140] J. Zhang, X. Chen, Z. Cai, L. Pan, H. Zhao, S. Yi, C. K. Yeo, B. Dai, and C. C. Loy, "Unsupervised 3d shape completion through gan inversion," in *CVPR*, 2021. 12, 16

[141] N. Tritrong, P. Rewatbowornwong, and S. Suwajanakorn, "Repurposing gans for one-shot semantic part segmentation," in *CVPR*, 2021. 12, 16

[142] R. Abdal, P. Zhu, N. Mitra, and P. Wonka, "Labels4free: Unsupervised segmentation using stylegan," in *ICCV*, 2021. 12, 16

[143] D. Bau, A. Andonian, A. Cui, Y. Park, A. Jahanian, A. Oliva, and A. Torralba, "Paint by word," *arXiv preprint arXiv:2103.10951*, 2021. 12, 16

[144] W. Xia, Y. Yang, J.-H. Xue, and B. Wu, "Towards open-world text-guided face image generation and manipulation," *arxiv preprint arxiv: 2104.08910*, 2021. 12, 16

[145] Z. Ren, S. X. Yu, and D. Whitney, "Controllable medical image generation via generative adversarial networks," in *Human Vision and Electronic Imaging*, 2021. 12, 16

[146] S. Liu, J. A. Dowling, C. Engstrom, P. B. Greer, S. Crozier, and S. S. Chandra, "Manipulating medical image translation with manifold disentanglement," *arXiv preprint arXiv:2011.13615*, 2020. 12

[147] L. Fetty, M. Bylund, P. Kuess, G. Heilemann, T. Nyholm, D. Georg, and T. Löfstedt, "Latent space manipulation for high-resolution medical image synthesis via the stylegan," *Zeitschrift für Medizinische Physik*, vol. 30, no. 4, pp. 305–314, 2020. 12, 16

[148] G. B. Daroach, J. A. Yoder, K. A. Iczkowski, and P. S. LaViolette, "High-resolution controllable prostatic histology synthesis using stylegan," in *BIOIMAGING*, 2021. 12

[149] R. Saha, B. Duke, F. Shkurti, G. Taylor, and P. Aarabi, "Loho: Latent optimization of hairstyles via orthogonalization," in *CVPR*, 2021. 13

[150] Y. Endo and Y. Kanamori, "Few-shot semantic image synthesis using stylegan prior," *arXiv preprint arXiv:2103.14877*, 2021. 13

[151] Y.-C. Cheng, C. H. Lin, H.-Y. Lee, J. Ren, S. Tulyakov, and M.-H. Yang, "In&out: Diverse image outpainting via gan inversion," *arXiv preprint arXiv:2104.00675*, 2021. 13

[152] W. Xia, Y. Yang, and J.-H. Xue, "Cali-sketch: Stroke calibration and completion for high-quality face image generation from poorly-drawn sketches," *arXiv preprint arXiv:1911.00426*, 2019. 13

[153] A. Ghosh, R. Zhang, P. K. Dokania, O. Wang, A. A. Efros, P. H. S. Torr, and E. Shechtman, "Interactive sketch & fill: Multiclass sketch-to-image translation," in *ICCV*, 2019. 13

[154] S.-Y. Chen, W. Su, L. Gao, S. Xia, and H. Fu, "DeepFaceDrawing: Deep generation of face images from sketches," *TOG*, 2020. 13

[155] M. Eitz, K. Hildebrand, T. Boubekeur, and M. Alexa, "Sketch-based image retrieval: Benchmark and bag-of-features descriptors," *TVCG*, vol. 17, no. 11, pp. 1624–1636, 2010. 13

[156] S. Dey, P. Riba, A. Dutta, J. Llados, and Y.-Z. Song, "Doodle to search: Practical zero-shot sketch-based image retrieval," in *CVPR*, 2019, pp. 2179–2188. 13

[157] G. Larsson, M. Maire, and G. Shakhnarovich, "Learning representations for automatic colorization," in *ECCV*, 2016. 13

[158] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," in *CVPR*, 2018, pp. 9446–9454. 13

[159] T. R. Shaham, T. Dekel, and T. Michaeli, "Singan: Learning a generative model from a single natural image," in *ICCV*, 2019, pp. 4570–4580. 13

[160] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995. 14

[161] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *CVPR*, 2016. 15

[162] E. J. Candès *et al.*, "Compressive sampling," in *ICM*, 2006. 15

[163] D. L. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006. 15

[164] A. Bora, A. Jalal, E. Price, and A. G. Dimakis, "Compressed sensing using generative models," *arXiv preprint arXiv:1703.03208*, 2017. 15

[165] S. A. Hussein, T. Tirer, and R. Giryes, "Image-adaptive gan based reconstruction," *arXiv preprint arXiv:1906.05284*, 2019. 15

[166] V. Shah and C. Hegde, "Solving linear inverse problems using gan priors: An algorithm with provable guarantees," in *ICASSP*, 2018, pp. 4609–4613. 15

[167] D. Cohen-Or, O. Sorkine, R. Gal, T. Leyvand, and Y.-Q. Xu, "Color harmonization," in *ACM SIGGRAPH*, 2006, pp. 624–630. 15

[168] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," *ECCV*, 2018. 15

[169] Y. Fan, B. Wu, T. Li, Y. Zhang, M. Li, Z. Li, and Y. Yang, "Sparse adversarial attack via perturbation factorization," in *ECCV*, 2020. 15

[170] W. Chen, Z. Zhang, X. Hu, and B. Wu, "Boosting decision-based black-box adversarial attacks with random sign flip," in *ECCV*, 2020. 15

[171] Y. Fan, B. Wu, T. Li, Y. Zhang, M. Li, Z. Li, and Y. Yang, "Sparse adversarial attack via perturbation factorization," in *ECCV*, 2020. 15

[172] S. Baluja and I. Fischer, "Adversarial transformation networks: Learning to generate adversarial examples," *arXiv preprint arXiv:1703.09387*, 2017. 15

[173] P. Samangouei, M. Kabkab, and R. Chellappa, "Defense-gan: Protecting classifiers against adversarial attacks using generative models," in *ICLR*, 2018. 15

[174] G. S. Dhillon, K. Azizzadenesheli, Z. C. Lipton, J. Bernstein, J. Kossaifi, A. Khanna, and A. Anandkumar, "Stochastic activation pruning for robust adversarial defense," *arXiv preprint arXiv:1803.01442*, 2018. 15

[175] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," *arXiv preprint arXiv:2103.00020*, 2021. 16

[176] C. Jia, Y. Yang, Y. Xia, Y.-T. Chen, Z. Parekh, H. Pham, Q. V. Le, Y. Sung, Z. Li, and T. Duerig, "Scaling up visual and vision-language representation learning with noisy text supervision," in *ICML*, 2021. 16

[177] E. Logacheva, R. Suvorov, O. Khomenko, A. Mashikhin, and V. Lempitsky, "Deeplandscape: Adversarial modeling of landscape videos," in *ECCV*, 2020. 16

[178] X. Yi, E. Walia, and P. Babyn, "Generative adversarial network in medical imaging: A review," *Medical image analysis*, vol. 58, p. 101552, 2019. 16

[179] L. Kuhnel, T. Fletcher, S. Joshi, and S. Sommer, "Latent space non-linear statistics," *arXiv preprint arXiv:1805.07632*, 2018. 16

[180] H. H. Harman, *Modern factor analysis*. University of Chicago Press, 1976. 17

[181] M. E. Davies and C. J. James, "Source separation using single channel ica," *Signal Processing*, 2007. 17

[182] M. Hoffman, F. R. Bach, and D. M. Blei, "Online learning for latent dirichlet allocation," in *NeurIPS*, 2010. 17

[183] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *JMLR*, vol. 3, no. Jan, pp. 993–1022, 2003. 17

[184] P. Esser, R. Rombach, and B. Ommer, "A disentangling invertible interpretation network for explaining latent representations," in *CVPR*, 2020. 17

[185] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *ICLR*, 2013. 17

[186] J. Zhu, D. Zhao, B. Zhang, and B. Zhou, "Disentangled inference for gans with latently invertible autoencoder," *arXiv preprint arXiv:1906.08090*, 2019. 17

[187] L. Shuyu and C. Ronald, "Ladder: Latent data distribution modelling with a generative prior," in *BMVC*, 2020. 17

[188] W. Xia, Z. Cheng, Y. Yang, and J.-H. Xue, "Cooperative semantic segmentation and image restoration in adverse environmental conditions," *arXiv preprint arXiv:1911.00679*, 2019. 17

[189] V. Nekrasov, T. Dharmasiri, A. Spek, T. Drummond, C. Shen, and I. D. Reid, "Real-time joint semantic segmentation and depth estimation using asymmetric annotations," in *ICRA*, 2019. 17

[190] W. Zhan, X. Ou, Y. Yang, and L. Chen, "Dsnet: Joint learning for scene segmentation and disparity estimation," in *ICRA*, 2019. 17

[191] Z. Chen and H. Zhang, "Learning implicit fields for generative shape modeling," in *CVPR*, 2019, pp. 5939–5948. 17

[192] R. Tucker and N. Snavely, "Single-view view synthesis with multiplane images," in *CVPR*, 2020, pp. 551–560. 17

[193] S. Rajeswar, F. Mannan, F. Golemo, J. Parent-Lévesque, D. Vazquez, D. Nowrouzezahrai, and A. Courville, "Pix2shape: Towards unsupervised learning of 3d scenes from images using a view-based representation," *IJCV*, pp. 1–16, 2020. 17

[194] A. Chen, R. Liu, L. Xie, and J. Yu, "A free viewpoint portrait generator with dynamic styling," *arXiv preprint arXiv:2007.03780*, 2020. 17

[195] L. Liu, W. Xu, M. Habermann, M. Zollhoefer, F. Bernard, H. Kim, W. Wang, and C. Theobalt, "Neural human video rendering by learning dynamic textures and rendering-to-video translation," *TVCG*, 2020. 17

[196] S. Lombardi, T. Simon, J. Saragih, G. Schwartz, A. Lehrmann, and Y. Sheikh, "Neural volumes: Learning dynamic renderable volumes from images," *arXiv preprint arXiv:1906.07751*, 2019. 17

[197] R. Martin-Brualla, N. Radwan, M. S. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. Duckworth, "Nerf in the wild: Neural radiance fields for unconstrained photo collections," in *CVPR*, 2021. 17

[198] S. Wu, C. Rupprecht, and A. Vedaldi, "Unsupervised learning of probably symmetric deformable 3d objects from images in the wild," in *CVPR*, 2020, pp. 1–10. 17

[199] T. Nguyen-Phuoc, C. Richardt, L. Mai, Y.-L. Yang, and N. Mitra, "Blockgan: Learning 3d object-aware scene representations from unlabelled images," *arXiv preprint arXiv:2002.08988*, 2020. 17

[200] H. Kato and T. Harada, "Self-supervised learning of 3d objects from natural images," *arXiv preprint arXiv:1911.08850*, 2019. 17

[201] Z. Zheng, T. Yu, Y. Liu, and Q. Dai, "Pamir: Parametric model-conditioned implicit representation for image-based human reconstruction," *arXiv preprint arXiv:2007.03858*, 2020. 17

[202] B. L. Bhatnagar, C. Sminchisescu, C. Theobalt, and G. Pons-Moll, "Combining implicit function learning and parametric models for 3d human reconstruction," in *ECCV*, 2020. 17

[203] T. He, J. Collomosse, H. Jin, and S. Soatto, "Geo-pifu: Geometry and pixel aligned implicit functions for single-view human reconstruction," in *NeurIPS*, 2020. 17

[204] S. Saito, T. Simon, J. Saragih, and H. Joo, "Pifuhd: Multi-level pixel-aligned implicit function for high-resolution 3d human digitization," in *CVPR*, 2020, pp. 84–93. 17

[205] B. Egger, W. A. Smith, A. Tewari, S. Wuhrer, M. Zollhoefer, T. Beeler, F. Bernard, T. Bolkart, A. Kortylewski, S. Romdhani *et al.*, "3d morphable face models—past, present, and future," *TOG*, vol. 39, no. 5, pp. 1–38, 2020. 17

[206] Z. He, A. Spurr, X. Zhang, and O. Hilliges, "Photo-realistic monocular gaze redirection using generative adversarial networks," *ICCV*, 2019. 17

[207] H. Zhou, S. Hadap, K. Sunkavalli, and D. W. Jacobs, "Deep single-image portrait relighting," in *ICCV*, 2019. 17

[208] X. Zhang, J. T. Barron, Y.-T. Tsai, R. Pandey, X. Zhang, R. Ng, and D. E. Jacobs, "Portrait shadow manipulation," *TOG*, 2020. 17

[209] X. Chen, J. Song, and O. Hilliges, "Monocular neural image based rendering with continuous view control," in *ICCV*, 2019. 17

[210] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," *arXiv preprint arXiv:2005.14165*, 2020. 17

[211] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016. 17

[212] B. Li, X. Qi, T. Lukasiewicz, and P. H. S. Torr, "Controllable text-to-image generation," in *NeurIPS*, 2019. 17

[213] Y. Jia, Y. Zhang, R. J. Weiss, Q. Wang, J. Shen, F. Ren, Z. Chen, P. Nguyen, R. Pang, I. Lopez-Moreno, and Y. Wu, "Transfer learning from speaker verification to multispeaker text-to-speech synthesis," in *NeurIPS*, 2018, pp. 4485–4495. 17

[214] K. R. Prajwal, R. Mukhopadhyay, V. P. Namboodiri, and C. Jawahar, "Learning individual speaking styles for accurate lip to speech synthesis," in *CVPR*, 2020. 17