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, for the image to 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 the pretrained GAN models such as StyleGAN and BigGAN to be used for real image editing applications. Meanwhile, GAN inversion also provides insights on the interpretation of GAN's latent space and how the realistic images can be generated. In this paper, we provide an overview of GAN inversion with a focus on its recent algorithms and applications. We cover important techniques of GAN inversion and their applications to image restoration and image manipulation. We further elaborate on some trends and challenges for future directions. 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 real data distribution. In recent years, GANs have been applied to numerous tasks ranging from image translation [3], [4], [5], 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 manipulation in the latent space is only applicable to the images generated from GANs rather than any given real images, due to the lack of inference functionality or encoder in GANs.

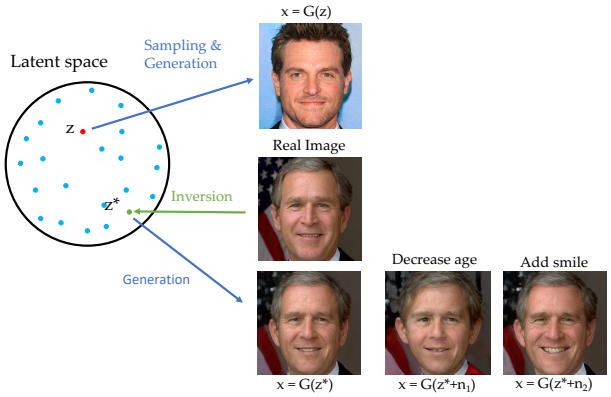

Fig. 1. **Illustration of GAN inversion.** Different from the conventional sampling and generation process using trained generator G, GAN inversion maps a given real image x to the latent space and obtains the latent code $\mathbf{z}^*$. The reconstructed image x^* is then obtained by $x^* = G(\mathbf{z}^*)$. 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 then can be faithfully reconstructed from the inverted code by the generator. Since GAN inversion plays an essential role of 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, as well as an insightful analysis, of all aspects of GAN inversion 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.

2 PRELIMINARIES

2.1 Problem Definition

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 is to map data x back to latent representation $\mathbf{z}^*$, or equivalently, to find an image x^* that can be entirely synthesized by the well-trained generator G and stay close to the real image x. Formally, denoting the signal to invert by $x \in \mathbb{R}^n$, 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], and LPIPS [30] metrics. This is usually a non-convex problem due to the non-convexity of $G(\mathbf{z})$, and GAN inversion methods focus on reconstructing the image content.

2.2 Trained GAN Models and Datasets

Deep generative models such as GANs [1] have been used to model natural image distributions and synthesize photo-realistic images. Recent advances in GANs such as DCGAN [31], WGAN [32], PGGAN [14], BigGAN [15], StyleGAN [16] and StyleGAN2 [17] have developed better architectures, losses and training schemes. These models are trained on diverse datasets, including faces (CelebA-HQ [14], FFHQ [16], [17], AnimeFaces [33] and AnimalFace [34]), scenes (LSUN [35]), and objects (LSUN [35] and ImageNet [36]).

2.2.1 GAN Models

DCGAN [31] uses convolutions in the discriminator and fractional-strided convolutions in the generator.

WGAN [32] minimizes the Wasserstein distance between the generated and real data distributions, which offers more model stability and makes the training process easier.

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 improves both the training speed and the stabilization, thereby facilitating image synthesis at higher resolution, *e.g.*, CelebA images at 1024×1024 pixels.

BigGAN [15] generates high-resolution and high-quality images, with modifications on scaling-up, architectural changes and orthogonal regularization to improve scalability, robustness and stability of large-scale GANs.

StyleGAN [16] implicitly learns hierarchical latent styles for image generation. This model manipulates per-channel mean and variance to control the style of an image [37] 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, redesigning generator, and removing progressive growing. The StyleGAN-based architectures have been applied to numerous applications [38], [39], [40].

2.2.2 Datasets

ImageNet [36] is a large-scale hand-annotated dataset for visual object recognition research, containing more than 14 million images with more than 20,000 categories.

CelebA [41] is a large-scale face attributes dataset consisting of 200K celebrity images with 40 attribute annotations each. CelebA, together with its succeeding CelebA-HQ [14], CelebAMask-HQ [42], and CelebA-Spoof [43], 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 [35] contains around 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.

Besides, some GAN inversion studies also use other datasets [44], [45], [46], [47], [48] in their experiments, such as **DeepFashion** [49], [50], [51], **AnimeFaces** [33], and **StreetScapes** [52].

2.3 Evaluation Metrics

Image synthesis based on GANs is usually evaluated in terms of photorealism, diversity, interpretability, and disentanglability. Photorealism is usually measured by structural similarity (PSNR and SSIM) and perceptual quality (IQA methods). Diversity is especially crucial to GAN generation methods; the most widely used metrics are IS, FID and LPIPS; recent studies also use SWD. Interpretability and disentanglability are the most relative metrics in the GAN inversion task. In the literature, some methods [53] use Cosine or Euclidean distance to evaluate different attributes between input and output while others [54] use the classification accuracy for assessment.

2.3.1 Image Quality Assessment

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 are 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.*, non-linearly perceived scale, bias and variance of rating criteria.

Peak Signal-to-Noise Ratio (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 image:

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

where L equals to $2^n - 1$ if represented using linear pulse-code modulation with n bits, *e.g.*, 255 in general cases using 8-bit representations.

Structural Similarity (SSIM) [55] 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

$$\mathrm{SSIM}(I, \hat{I}) = [\mathcal{C}_l(I, \hat{I})]^{\alpha}[\mathcal{C}_c(I, \hat{I})]^{\beta}[\mathcal{C}_s(I, \hat{I})]^{\gamma}, \tag{3}$$

where α, β, γ are control parameters for adjusting the relative importance. The details of these terms can be found in [55].

2.3.2 Learning-based Perceptual Quality

Inception Score (IS) [56] 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 [57] pretrained on the ImageNet [58] when applied to generated images:

$$\mathrm{IS} = \exp\left(\mathbb{E}_{x \sim p_g} D_{KL}(p(y|x) \parallel p(y))\right), \tag{4}$$

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.

Fréchet Inception Distance [59] (FID) is defined using the Fréchet distance between two multivariate Gaussians:

$$\mathrm{FID} = \|\mu_r - \mu_g\|^2 + \mathrm{Tr}(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{\frac{1}{2}}), \tag{5}$$

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 [57] pool3 layer for real and generated samples, respectively. The lowest FID means the most perceptual results.

Fréchet Segmentation Distance (FSD) [26] is an interpretable counterpart to the FID metric:

$$\mathrm{FSD} = \|\mu_g - \mu_t\|^2 + \mathrm{Tr}(\Sigma_g + \Sigma_t - 2(\Sigma_g\Sigma_t)^{\frac{1}{2}}), \tag{6}$$

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) [60] 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:

$$\tilde{W}(X, Y)^2 = \int_{\theta \in \Omega} W\left(X_\theta, Y_\theta\right)^2 \, \mathrm{d}\theta \tag{7}$$

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 a 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 average across spatial dimensions and 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):

$$d(x, x_0) = \sum_l \frac{1}{H_l W_l} \sum_{h,w} ||w_l \odot (\hat{y}^l_{hw} - \hat{y}^l_{0hw})||_2^2. \tag{8}$$

Perceptual Path Length (PPL) [16] measures the difference of interpolations between two inputs and is calculated by the pairwise image distance between the shifted image and the synthesized image by using the VGG16 [61] embedding. To be specific, if subdividing a latent space interpolation path into linear segments, the total perceptual length of this segmented path can be defined as the sum of perceptual differences over each segment. PPL in latent space $\mathcal{Z}$ is

$$l_{\mathcal{Z}} = \mathbb{E}\Big[\frac{1}{\epsilon^2} d\big(G(\mathrm{slerp}(\mathbf{z}_1, \mathbf{z}_2;\, t)), G(\mathrm{slerp}(\mathbf{z}_1, \mathbf{z}_2;\, t + \epsilon))\big)\Big], \tag{9}$$

where $\mathbf{z}_1, \mathbf{z}_2 \sim P(\mathbf{z}), t \sim U(0, 1)$, ϵ is a small subdivision, G is the generator (*i.e.*, $g \circ f$ for style-based networks), and $d(\cdot, \cdot)$ evaluates the perceptual distance between the resulting images. PPL in $\mathcal{W}$ is carried out in a similar way:

$$\begin{aligned} l_{\mathcal{Z}} = \mathbb{E}\Big[\frac{1}{\epsilon^2} d\big(&g(\mathrm{lerp}(f(\mathbf{z}_1), f(\mathbf{z}_2);\, t)), \\ &g(\mathrm{lerp}(f(\mathbf{z}_1), f(\mathbf{z}_2);\, t + \epsilon))\big)\Big]. \end{aligned} \tag{10}$$

Typically, $\mathrm{slerp}(\cdot)$ in $\mathcal{Z}$ denotes spherical interpolation [62], which is the most appropriate way of interpolating in the normalized input latent space [63]. The linear interpolation $\mathrm{lerp}(\cdot)$ is used in $\mathcal{W}$ since vectors in $\mathcal{W}$ are not normalized in any fashion.

2.3.3 Inversion Accuracy

Classification Accuracy. Voynov *et al.* [54] propose the **Reconstructor Classification Accuracy** (RCA) to measure the model interpretability by predicting the direction in the latent space that a given image transformation is generated. The reconstructor's classification solves a multi-class classification problem and high RCA values imply that directions

are easy to distinguish from each other, *i.e.*, corresponding image transformations do not "interfere" or influence different factors of variations. Abdal *et al.* [64] use the **Face Identity** to evaluate the quality of the edits and quantify the identity preserving property of the edits. A face classifier model [65] 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 calculate the Euclidean distance and the cosine similarity between the embeddings.

Reconstruction Distance. Nitzan *et al.* [53] 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} [66]. In contrast, the pose preservation is calculated as the Euclidean distance between Euler angles of I_{attr} and I_{out}. Abdal *et al.* [64] 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 the pose attribute 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 conditional edit along attribute specification x and $\mathcal{A}_p$ denotes the pose attribute vector regressed by the attribute classifier.

3 Latent Space of GANs

Before introducing GAN inversion methods, we first review interesting features of latent space, *e.g.*, $\mathcal{W}$ and $\mathcal{Z}$ spaces for StyleGAN [16]. We discuss what GAN models can learn in terms of interpretability, disentanglability, and invertibility.

3.1 $\mathcal{Z}$ space and $\mathcal{W}$ space

The generative model in the GAN architecture learns to map values sampled from a simple distribution, *e.g.*, normal or uniform distribution, to generated images. These values, sampled *directly* from the distribution, form a structure typically called latent $\mathcal{Z}$ space, and these values are latent codes or latent representations, denoted by $\mathbf{z} \in \mathcal{Z}$. Most latent spaces of GANs can be described by the $\mathcal{Z}$ space, including DCGAN [31], PGGAN [14] and BigGAN [15]. As discussed in Section 2.2.1, recent work [16] further converts native $\mathbf{z}$ to the mapped style vectors $\mathbf{w}$ by a non-linear mapping network f implemented with an 8-layer multi-layer perceptron (MLP), which forms another intermediate latent space referred as $\mathcal{W}$ space. Examples include the $\mathcal{W}$ space for the StyleGAN [16], and the $\mathcal{W}^+$ space for the StyleGAN2 [17].

Due to the mapping network and affine transformations, the $\mathcal{W}$ space of StyleGAN contains more untangled features than $\mathcal{Z}$ space. The $\mathcal{W}$ space alleviates entanglement [16] such that images can be easily generated. Other studies also analyze the separability and semantics of both $\mathcal{W}$ and $\mathcal{Z}$ spaces. For example, Shen *et al.* [21] 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 the StyleGAN method tends to learn semantic information based on the $\mathcal{W}$ space, and performs better than the one using the $\mathcal{Z}$ space. For semantics, they evaluate classification accuracy on their latent separation boundaries with respect to different attributes. The StyleGAN with the $\mathcal{W}$ space achieves higher accuracy than the PGGAN with the $\mathcal{Z}$ space. The constraints of the $\mathcal{Z}$ space subject to normal distribution limit its representative capacity for the semantic attributes since the assumption may not hold. To tackle spatial entanglement caused by the intrinsic complexity of style-based generators [16] and the spatial invariance of AdaIN normalization [37], very recent methods [67], [68] further propose the $\mathcal{S}$ space to achieve spatial disentanglement in the spatial dimension instead of at the semantic level. By directly intervening the style code $s \in \mathcal{S}$, both methods [67], [68] can achieve fine-grained controls on local translations.

3.2 Interpretability

Numerous methods have been developed to interpret hidden feature vectors or individual neurons of deep models based on three groups of approaches: network modification, feature visualization, and vector arithmetic. The first group of approaches focuses on modifying network architectures or losses for training to analyze how the models perform. Tao *et al.* [69] and Li *et al.* [8], [70] utilize attention mechanisms to learn image-text matching by correlating fine-grained word-level information with the intermediate feature maps, which enhances the detailed attribute modification guided by natural language descriptions. Xia *et al.* [7] introduce a controller with several branches that separately edit head pose, gaze direction and other secondary facial attributes for gaze redirection and interpolation.

Existing methods mainly focus on the second group of approaches that uses feature visualization to interpret model performance. Nguyen *et al.* [71] optimize over input codes of a generator network that was trained to reconstruct images from hidden layers. Recently, Daras *et al.* [72] introduce two-dimensional local attention mechanisms for generative models and show that the newly introduced attention heads indeed capture interesting aspects of the real image by visualizing the attention maps. The third group of approaches utilizes the vector arithmetic property. Radford *et al.* [31] demonstrate a rich linear structure of trained GANs, indicating that algebraic operations in the latent space can lead to semantically meaningful manipulation in the image space. Numerous approaches have been developed to synthesize different results by varying the latent codes. Subsequent studies [21], [22], [73] further demonstrate that the latent space of GANs encodes rich hierarchical semantic information. Such vector arithmetic property is often realized by moving the latent code towards specific directions as $\mathbf{z}' = \mathbf{z} + \alpha\mathbf{n}$, where $\mathbf{n} \in \mathbb{R}^d$ denotes the direction corresponding to a particular attribute and α is the step.

3.3 Disentanglability

As there is no precise definition, we define disentanglability as the property that latent vector $\mathbf{z}$ can be separated into diverse and independent parts of representation $\mathbf{z_k}$. In addition, *diversity* means that each factor $\mathbf{z_k}$ represents a specific interpretable concept, and *independence* indicates

that it would be possible to analyze and modify different factors $\mathbf{z_k}$ independently of each other. This implies a factorization of their joint density $p(\tilde{\mathbf{z}}) = \prod_{k=0}^{K} p(\mathbf{z_k})$. The latent features of well-disentangled features of different classes can be visualized [74], [75] by using the t-SNE [76] method. The disentangled representation learning gives an exploitable structure and makes us able to change only some properties of the underlying state, *e.g.*, hair color of a facial image, while leaving all other properties invariant. Many recent methods on disentanglement aim to learn an interpretable and transferable representation with explicit constraints in training. For example, Liu *et al.* [77] propose a unified model that learns disentangled representation for describing and manipulating data across multiple domains, and Lu *et al.* [78] disentangle the content and blur features from blurred images for image restoration.

Some unconditional image generation models [16], [17] demonstrate well-disentanglement properties. Rather than learning disentangled latent space with explicit constraints, these methods introduce a non-linear mapping network to the generator: $f : \mathcal{Z} \rightarrow \mathcal{W}$. For example, the intermediate latent space $\mathcal{W}$ of StyleGAN [16] is induced from a learned piecewise continuous mapping. It yields less entangled representations in an unsupervised setting, *i.e.*, when the factors of variation are not known in advance. In subsequent sections, we will show that this well-disentangled latent space offers feasible inversion and achieves better results.

3.4 Invertibility

As discussed in Section 3.1, Radford *et al.* [31] present the vector arithmetic property of latent representation $\mathbf{z}$. Creswell *et al.* [79] propose the concept of inversion together with their inverting algorithm. Esser *et al.* [80] show that sampling from the simple distribution, *e.g.*, a Gaussian distribution, benefits the generative models trained for image generation. Due to the convexity of the Gaussian density, linear operations between representations are meaningful, and linear interpolations enable traversing along the latent space of GANs.

Although the above-mentioned methods perform well empirically, less success has been made in theoretical understanding of the model invertibility. Recent methods aim to address this issue from different theoretical perspectives, such as compressed sensing [81], [82], [83] and sparse representation [84]. For example, Hand *et al.* [85] establish that a fully connected generative network with weights following Gaussian distribution can be inverted given only compressive linear observations of its last layer. Gilbert *et al.* [82] study a 1-layer network with a special activation function (concatenated ReLU [86], which is essentially linear) and a strong k-sparsity assumption on the latent code. Ma *et al.* [83] show that the expansive network consisting of two layers of transposed convolutions followed by ReLU [87] activation functions is invertible. Aberdam *et al.* [84] provide invertibility and uniqueness guarantees for general non-statistical weight matrices. In [84], theoretical conditions are derived from the sparse representation theory. It provides the invertibility of deep generative models by ensuring a unique solution for the inversion problem defined in (1). One corollary is that the number of nonzero elements should expand by a factor of at least 2 between layers to ensure a unique global minimum of the inverse problem, which effectively predicts whether the generative process is invertible or not.

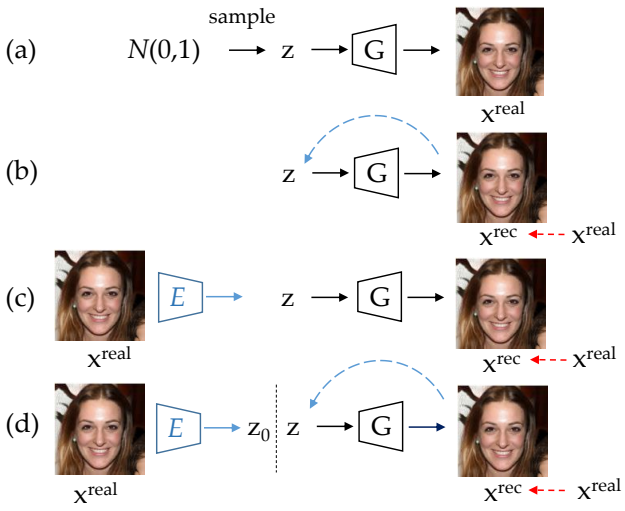

Fig. 2. **Illustration of GAN Inversion Methods.** (a) given a well-trained GAN model, photo-realistic images can be generated from randomly sampled latent vectors. (b) **optimization-based** inversion uses an optimization algorithm to iteratively optimize the latent code to minimize the pixel-wise reconstruction 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.

4 GAN Inversion Methods

4.1 Inversion Methods

There are primarily 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 2. The learned inverse representations also have other characteristics, *i.e.*, interpretable directions, semantic-aware, layer-wise, non-interference, region-of-interesting, and out-of-distribution. Table 1 lists characteristics of existing state-of-the-art GAN inversion methods.

4.1.1 *Learning-based GAN Inversion*

Learning-based GAN inversion [23], [73], [115] typically involves training an encoding neural network $E(x;\theta_E)$ to map an image x to the latent code $\mathbf{z}$ by

$$\theta_E^* = \arg\min_{\theta_E} \sum_n \mathcal{L}(G(E(x_n;\theta_E)),\, x_n), \tag{11}$$

where x_n denotes the n-th image in the dataset. The architecture of the predictive model E is usually equivalent to that of the discriminator D of the adversarial network, and only varies in the final layer. The objective in (11) is reminiscent of an auto-encoder pipeline, with an encoder E and a decoder G. The decoder G is fixed throughout the training. While the optimization problem described in (1) is the same as the learning objective (11), the learning-based approach

TABLE 1
Characteristics of GAN inversion methods. 'Type' includes Learning-based (L.), Optimization-based (O.), Hybrid (H.), and Closed-Form (C.) GAN inversion. I.-D., S.-A., L.-W., N.-I., R.-I. and O.-D. denote discovering the Interpretable Directions in a Supervised (S.) or an Unsupervised (U.) manner, Semantic-Aware, Layer-Wise, Non-Interference, Region-of-Interest, and Out-of-Distribution, 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 2.2.

Method	Publication	Type	I.-D.	S.-A.	L.-W.	N.-I.	R.-I.	O.-D.	GAN Model	Dataset
Zhu *et al.* [23]	2016, ECCV	H.	✗	✗	✗	✗	✗	✗	[31]	[35], [88], [89]
Creswell *et al.* [79]	2018, TNNLS	O.	✗	✗	✗	✗	✗	✗	[31], [32]	[41], [88]
GAN Dissection [90]	2019, ICLR	O.	✗	✓	✓	✗	✓	✗	[14]	[35]
GAN Paint [91]	2019, TOG	H.	✗	✓	✓	✗	✓	✗	[14]	[35]
Raj *et al.* [92]	2019, ICCV	O.	✗	✗	✗	✗	✗	✗	[31], [93]	[35], [41], [44]
GANSeeing [26]	2019, ICCV	H.	✗	✓	✓	✗	✗	✗	[14], [16], [32]	[35]
Image2StyleGAN [24]	2019, ICCV	O.	✗	✓	✗	✗	✗	✗	[16]	[16]
Image2StyleGAN++ [25]	2020, CVPR	O.	✗	✓	✗	✗	✓	✗	[14], [16]	[14], [16]
mGANPrior [94]	2020, CVPR	O.	✗	✓	✓	✓	✗	✓	[14], [16]	[14], [16], [35]
Editing in Style [95]	2020, CVPR	O.	✗	✓	✗	✗	✓	✗	[14], [16], [17]	[16], [35]
StyleRig [96]	2020, CVPR	L.	✗	✓	✗	✗	✗	✗	[16]	[16]
InterFaceGAN [21]	2020, CVPR	L., O.	S.	✓	✓	✓	✗	✓	[14], [16]	[16]
YLG [72]	2020, CVPR	O.	✗	✗	✗	✗	✗	✓	[93]	[36]
GANRewriting [97]	2020, ECCV	O.	✗	✓	✗	✗	✓	✗	[14], [17]	[35]
DGP [28]	2020, ECCV	O.	✗	✗	✓	✗	✗	✗	[15]	[36], [89]
Huh *et al.* [27]	2020, ECCV	O.	✗	✓	✗	✗	✓	✓	[15], [17]	[16], [35], [36]
IDInvert [22]	2020, ECCV	H.	✗	✓	✗	✗	✗	✓	[16]	[16], [35]
StyleGAN2 Distillation [98]	2020, ECCV	O.	S.	✓	✗	✓	✗	✓	[17]	[16]
GANSteering [20]	2020, ICLR	O.	S.	✓	✗	✗	✗	✗	[15], [16], [31]	[16], [35], [36]
GANLatentDiscovery [54]	2020, ICML	O.	U.	✓	✗	✗	✗	✗	[14], [15]	[14], [33], [44]
MimicGAN [99]	2020, IJCV	O.	✗	✗	✗	✗	✗	✗	[31]	
PIE [100]	2020, TOG	O.	S.	✓	✗	✓	✗	✓	[16]	[41]
Nitzan *et al.* [53]	2020, TOG	L.	✗	✓	✗	✓	✗	✗	[16]	[14], [16]
StyleFlow [64]	2021, TOG	O.	S.	✓	✗	✓	✗	✓	[16], [17]	[16], [35]
GANSpace [101]	2020, NeurIPS	O.	U.	✓	✓	✓	✗	✗	[15], [16], [17]	[14], [35]
Aberdam *et al.* [84]	2020, arxiv	O.	✗	✗	✓	✗	✗	✗	a two-layer model	[44]
StyleGAN-Encoder [102]	2020, arxiv	L.	✗	✓	✗	✓	✗	✗	[16]	[14], [16], [47]
pSp [103]	2020, arxiv	L.	✗	✓	✓	✓	✓	✓	[17]	[14]
Style Intervention [67]	2020, arxiv	O.	S.	✓	✗	✓	✓	✓	[17]	[67]
StyleSpace [68]	2020, arxiv	O.	S.	✓	✗	✓	✓	✓	[17]	[16], [35]
Lu *et al.* [104]	2020, arxiv	L.	U.	✓	✗	✓	✗	✗	[14], [17], [105]	[16], [36]
Cherepkov *et al.* [106]	2020, arxiv	O.	U.	✓	✗	✓	✗	✓	[17]	[16], [35]
Spingarn *et al.* [107]	2021, ICLR	C.	U.	✓	✗	✓	✗	✗	[15]	[36]
Zhuang *et al.* [108]	2021, ICLR	O.	✗	✓	✗	✓	✗	✓	[14], [17]	[14], [16]
Chai *et al.* [109]	2021, ICLR	L.	✗	✓	✗	✗	✓	✗	[14], [17]	[14], [16], [35]
SeFa [110]	2021, CVPR	C.	U.	✓	✓	✓	✓	✓	[14], [15], [16], [17]	[14], [16], [35], [36], [52]
GH-Feat [111]	2021, CVPR	L.	✗	✓	✓	✗	✗	✗	[16]	[16], [35], [44]
Hijack-GAN [112]	2021, CVPR	O.	U.	✓	✗	✓	✗	✓	[14], [16]	[14]
e4e [113]	2021, arxiv	L.	✗	✓	✓	✓	✓	✓	[17]	[14], [16], [35]
SAM [114]	2021, arxiv	L.	✗	✓	✗	✓	✗	✓	[16]	[14], [16]

often achieves better performance than direct optimization and does not fall into local optima [23], [84].

For example, Perarnau *et al.* [115] 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 sub-encoders: 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'. Guan *et al.* [102] propose the embedding network, which 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 turns supervises the training of the embedding network, using the MSE loss, LPIPS loss and latent code loss. Tewari *et al.* [96] 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 better reuse the layer-wise representations learned by the StyleGAN model, Xu *et al.* [111] propose to train a hierarchical encoder by treating the pretrained StyleGAN generator as a learned loss (similar to perceptual loss [29] using learned VGG [61]). The learned disentangled multi-level visual features $\{\mathbf{y}^{(\ell)}\}_{\ell=1}^{L}$ are then fed into per-layer adaptive instance normalization (AdaIN) [37] of the fixed StyleGAN generator to obtain the desired images by replac-

ing the original style code:

$$\texttt{AdaIN}(x_i^{(\ell)}, \mathbf{y}^{(\ell)}) = \mathbf{y}_{s,i}^{(\ell)} \frac{x_i^{(\ell)} - \mu(x_i^{(\ell)})}{\sigma(x_i^{(\ell)})} + \mathbf{y}_{b,i}^{(\ell)}, \tag{12}$$

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, $\mathbf{y}_s^{(\ell)}$ and $\mathbf{y}_b^{(\ell)}$ correspond to the scale and weight parameters in AdaIN respectively.

Although some methods [116], [117] 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.

4.1.2 Optimization-based GAN Inversion

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

$$\mathbf{z}^* = \arg\min_{\mathbf{z}} \ell(x, G(\mathbf{z}; \theta)), \tag{13}$$

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], [79], [79], [83], [118], [118] or other iterative algorithms [54], [119]. For example, Ramesh *et al.* [119] and Voynov *et al.* [54] 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.* [54] 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 in 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.* [54] can be applied to a larger number of directions while the approach by Ramesh *et al.* [119] is limited to the maximal number of discovered directions equal to the dimension of latent space.

To deal with the local minima issue, numerous optimization methods have been developed. Generally, there are two types of optimizers: gradient-based (ADAM [120], L-BFGS [121], Hamiltonian Monte Carlo (HMC) [122]) and gradient-free (Covariance Matrix Adaptation (CMA) [123]). 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 various gradient-free optimization methods in the Nevergrad library [124] with the default optimization hyper-parameters. They find that CMA and its variants 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 non-convex, 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], [31]. 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], [96], [102] instead train a deep neural network to minimize (1) directly as introduced in Section 4.1.1.

We note that some [23], [73], [102] propose to use an encoder to provide better initialization for optimization (which will discussed in Section 4.1.3).

4.1.3 Hybrid GAN Inversion

The hybrid methods [22], [23], [26], [73], [102] exploit advantages of both approaches discussed above. As one of the pioneering work 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 $E(x; \theta_E)$, then uses the obtained $\mathbf{z}$ as the initialization for optimization. The learned predictive model serves as a fast bottom-up initialization for the non-convex optimization problem (1).

The subsequent studies basically follow this framework and have proposed several variants. For example, to invert G, Bau *et al.* [73] begin with 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 layer-wise 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 4.2.3 for more details). Zhu *et al.* [22] show that in most existing methods, the 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 the inverted code meaningful for semantic editing (see Section 4.2.2 for more details).

Without using the above framework, Guan *et al.* [102] 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

$$\begin{aligned}\mathcal{L}_{emb} &= \lambda_1 \underbrace{||\mathbf{w_e} - \mathbf{w_o}||_2^2}_{\text{latent loss}} + \lambda_2 \underbrace{||x_e - x_o||_2^2}_{\text{image loss}} + \lambda_3 \underbrace{\Phi(x_e, x_o)}_{\text{feature loss}}, \\ \mathcal{L}_{opt} &= ||G(\mathbf{w}) - x||_2^2 + \alpha\Phi(G(\mathbf{w}), x),\end{aligned} \tag{14}$$

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, λ_3, α are the loss weights.

4.1.4 Closed-Form Solution

Very recently, two methods [107], [110] found that the interpretable directions can be directly computed in *closed-form*, without any kinds of training or optimization. To be specific, Spingarn *et al.* [107] 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 applying it directly to the generated image, *i.e.*, $G(\mathbf{z} + \mathbf{q}) \approx \mathcal{T}\{G(\mathbf{z})\}$ for every $\mathbf{z}$. 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 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. To guarantee that this holds over random draws of $\mathbf{z}$, they formulate the problem as

$$\min_{\mathbf{q}} \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}}[\|\mathbf{D}\Big(\mathbf{W}(\mathbf{z}+\mathbf{q}) + \mathbf{b} - \mathbf{P}(\mathbf{W}\mathbf{z}+\mathbf{b})\Big)\|^2], \tag{15}$$

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 $\mathbf{P}$,

$$\mathbf{q} = (\mathbf{W}^T\mathbf{D}^2\,\mathbf{W})^{-1}\mathbf{W}^T\mathbf{D}^2(\mathbf{P}-\mathbf{I})\,\mathbf{b}. \tag{16}$$

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 have 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 (15) is then formulated as

$$\min_{\mathbf{M},\mathbf{q}} \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}}[\|\mathbf{D}\Big(\mathbf{W}(\mathbf{M}\mathbf{z}+\mathbf{q}) + \mathbf{b} - \mathbf{P}(\mathbf{W}\mathbf{z}+\mathbf{b})\Big)\|^2]. \tag{17}$$

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 (16) and the solution for $\mathbf{M}$ is

$$\mathbf{M}_{i,i} = \frac{\mathbf{w}_i^T\mathbf{D}^2\mathbf{P}\,\mathbf{w}_i}{\mathbf{w}_i^T\mathbf{D}^2\,\mathbf{w}_i}, \tag{18}$$

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

Shen *et al.* [110] observe that the semantic transformation of an image, usually denoted by moving the latent code towards a certain direction $\mathbf{n}' = \mathbf{z}+\alpha\mathbf{n}$, is actually determined by the latent direction $\mathbf{n}$ and is independent of the sampled code $\mathbf{z}$. Based on that, they turn into 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:

$$\mathbf{n}^* = \underset{\{\mathbf{n}\in\mathbb{R}^d:\ \mathbf{n}^T\mathbf{n}=1\}}{\arg\max} ||\mathbf{A}\mathbf{n}||_2^2. \tag{19}$$

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

4.2 Characteristics of GAN Inversion Methods

We discuss some important characteristics of GAN inversion methods in this section.

4.2.1 Interpretable Directions

Some GAN inversion 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.

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 pre-defined labels by introducing a pretrained classifier (*e.g.*, predicting face attributes or light directions) [19], [20], [21], [64] or extracting statistical image information (*e.g.*, color variations) [125]. 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.* [64] learn a semantic mapping between the $\mathcal{Z}$ space and the $\mathcal{W}$ space 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 a manual labeling effort. Jahanian *et al.* [20] optimize trajectories (both linear and non-linear, as shown in Figure 3) in a self-supervised manner. Taking the linear walk $\boldsymbol{w}$ for example, given an inverted source image $G(\mathbf{z})$, they learn $\boldsymbol{w}$ by minimizing the objective function

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} \mathbb{E}_{\mathbf{z},\alpha}[\mathcal{L}(G(\mathbf{z}+\alpha\mathbf{w}), \texttt{edit}(G(\mathbf{z}), \alpha))]. \tag{20}$$

Here, $\mathcal{L}$ 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 $G(\mathbf{z})$.

Unsupervised-Setting. The supervised setting would introduce bias into the experiment since the sampled codes

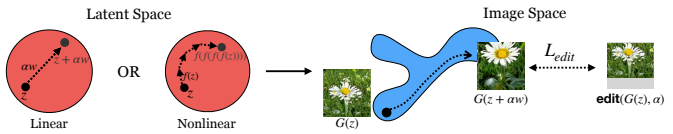

Fig. 3. Illustration of discovering interpretable directions in the latent space [20]. The goal is to find a path in $\mathcal{Z}$ space to transform the generated image $G(\mathbf{z})$ to its edited version $\texttt{edit}(G(\mathbf{z},\alpha))$, *e.g.*, an $\alpha\times$ zoom. The transformation can be represented as $G(\mathbf{z}+\alpha\mathbf{w})$ for a linear walk or $G(f(f(...(\mathbf{z})))$ for a non-linear walk.

and synthesized images used as supervision are different in each sampling and may lead to different discoveries of interpretable directions [110]. 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 non-local. Thus, some [54], [101], [104], [106] 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.* [101] 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 selective application of the principal components allows control of features. Shen *et al.* [110] and Spingarn *et al.* [107] propose to directly compute the interpretable directions in closed form from the pretrained models, without any kinds of training or optimization (see Section 4.1.4 for more details).

4.2.2 Semantic-Aware

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 4, existing approaches typically sample a collection of latent codes $\mathbf{z}$ randomly and feed them into $G(\cdot)$ to get the corresponding synthesis x'. The encoder $E(\cdot)$ is then trained by

$$\min_{\Theta_E} \mathcal{L}_E = \|\mathbf{z} - E(G(\mathbf{z}))\|_2, \tag{21}$$

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

Collins *et al.* [95] 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. Then, they perform instance-level domain-regularized optimization by involving the encoder as a regularization term. Such optimization helps to better reconstruct the pixel values without

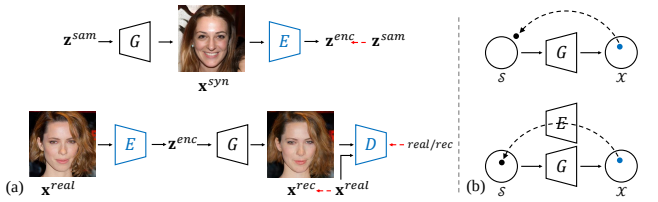

Fig. 4. **Comparisons between the training of (upper) conventional encoder and (lower) domain-guided encoder proposed in [22] for GAN inversion.** Blue blocks represent trainable models and red dashed arrows indicate the supervisions. The domain-guided encoder is trained to recover the real images, instead 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.

affecting the semantic property of the inverted code. The training process is formulated as

$$\min_{\Theta_E} \mathcal{L}_E = \|x - G(E(x))\|_2 + \lambda_1 \|F(x) - F(G(E(x)))\|_2 - \lambda_2 \mathbb{E}[D(G(E(x)))], \tag{22}$$

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.

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 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

$$\mathbf{z}^* = \arg\min_{\mathbf{z}} \|x - G(\mathbf{z})\|_2 + \lambda_1' \|F(x) - F(G(\mathbf{z}))\|_2 + \lambda_2' \|\mathbf{z} - E(G(\mathbf{z}))\|_2, \tag{23}$$

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.

4.2.3 Layer-Wise

As it is not feasible to determine the generator for the full inversion problem defined by (1) when the number of layers is large, a few approaches [26], [84], [126] have been developed to solve a tractable sub-problem by decomposing the generator G into layers:

$$G = G_f(g_n(\cdots((g_1(\mathbf{z})))), \tag{24}$$

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

The simplest layer-wise GAN inversion is based on one layer. We start inverting a single layer to find if $\min_{\mathbf{z}} \|x - G(\mathbf{z})\|_p = 0$, a specific formulation of (1), holds

for any p-norm. Since the problem is non-convex, additional assumptions are required [127] 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:

$$\begin{aligned} {\mathbf{w}_i}^\top \mathbf{z} + b_i &= x_i, \quad \forall i \text{ s.t. } x_i > 0, \\ {\mathbf{w}_i}^\top \mathbf{z} + b_i &\leq 0, \quad \forall i \text{ s.t. } x_i = 0. \end{aligned} \tag{25}$$

The solution set of (25) is convex and forms a polytope. However, it also possibly includes uncountable feasible points [126], which makes it unclear how to invert layer-wisely inversion. Several approaches make additional assumptions to generalize the above result to deeper neural networks. Lei *et al.* [126] 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: (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 [87] or leaky-ReLU [128]. However, these assumptions often do not hold in practice. Aberdam *et al.* [84] relax the expansion assumption of [126], and propose a method that relies on the expansion of the number of non-zero elements. They reformulate problem (1) with ℓ_2 to a layer-wise expression

$$\arg\min_{\mathbf{z}} \left\| \mathbf{y} - \phi((\prod_{i=L}^{0} \mathbf{W}_i^{\hat{\mathcal{S}}_{i+1}})\mathbf{z}) \right\|_2^2, \tag{26}$$

where $\mathbf{y} = G(\mathbf{z}) + \mathbf{e}$, $\{\mathcal{S}_i\}_{i=1}^L$ are support sets of each layers, ϕ is an invertible activation function ReLU, and $\mathbf{W}_i^S$ denotes 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 nor 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 to solve the easier problem of inverting the final layers G_f:

$$x = G_f(\mathbf{r}^*), \tag{27}$$

where $\mathbf{r}^* = \arg\min_{\mathbf{r}} \ell(G_f(\mathbf{r}), \mathbf{r})$ 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 an intermediate representation $\mathbf{r}^* \approx \mathbf{r}_0 = g_n(\cdots(g_1(\mathbf{z}_0)))$ 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, 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 of the 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:

$$\begin{aligned} \mathcal{L}_{\text{L}} &= \mathbb{E}_{\mathbf{z}}[||\mathbf{r}_{i-1} - e(g_i(\mathbf{r}_{i-1}))||_1], \\ \mathcal{L}_{\text{R}} &= \mathbb{E}_{\mathbf{z}}[||\mathbf{r}_i - g_i(e(\mathbf{r}_i))||_1], \\ e_i &= \arg\min_{e} \quad \mathcal{L}_{\text{L}} + \lambda_{\text{R}}\, \mathcal{L}_{\text{R}}, \end{aligned} \tag{28}$$

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

$$E^* = e_1(e_2(\cdots(e_n(e_f(x))))). \tag{29}$$

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.

4.2.4 Non-interference

When several attributes are involved, editing one may affect another since some semantics are not separated. Non-interference GAN inversion aims to tackle multi-attribute image manipulation without interference. This characteristic is also named multi-dimensional [53] or conditional editing [21] in other GAN inversion approaches. 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 $\mathbf{n} \in \mathbb{R}$ 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 $m \leq k(k-1)/2$) corresponding hyperplanes $\{\mathbf{n}_1, \cdots, \mathbf{n}_m\}$. Non-interference manipulation of multi-attributes means that $\{\mathbf{n}_1, \cdots, \mathbf{n}_m\}$ should be orthogonal with each other. If this condition does not hold, 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 5, 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 [98], [102] based on the pretrained StyleGAN [16] or StyleGAN2 [17] models can also manipulate multiple attributes due to the stronger separability of $\mathcal{W}$ space than $\mathcal{Z}$ space. However, as observed by recent methods [67], [68], [129], 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.* [67] 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 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

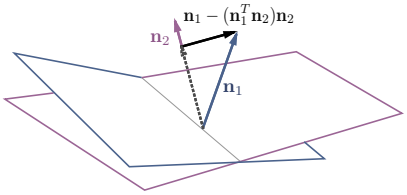

Fig. 5. **Illustration of the non-interference property in subspace.** The projection of $\mathbf{n}_1$ onto $\mathbf{n}_2$ is subtracted from $\mathbf{n}_1$, resulting in a new direction $\mathbf{n}_1 - (\mathbf{n}_1^\top \mathbf{n}_2)\mathbf{n}_2$. This figure is from [21].

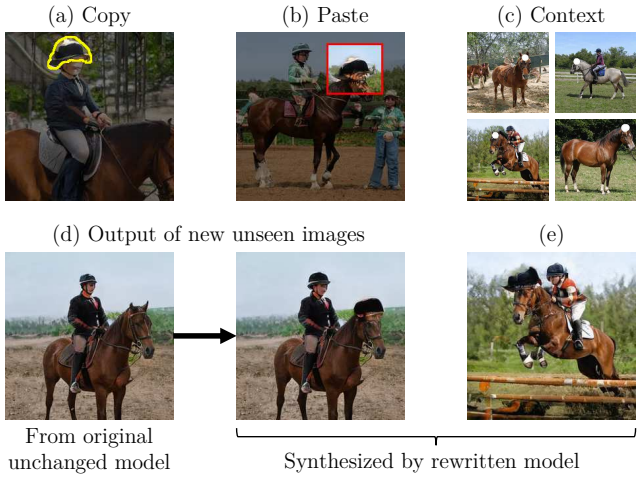

Fig. 6. **Copy-Paste-Context interface for rewriting a model [23].** (a) Copy: the user uses a brush to select a region containing an interesting object or shape, defining the target. (b) Paste: The user positions and pastes the copied object into a single target image. (c) Context: To control generalization, the user selects target regions in several images. (d) The edit is applied to the model, not to a specific image, such that newly generated images will have hats on top of horse heads. (e) The change has been applied to different types of horses and poses.

semantic directions without affecting others and can achieve fine-grained controls on local translations.

4.2.5 Region-of-Interest

The region-of-interest property of GAN inversion allows editing some desired regions in a given image with user manipulation, which often involves additional tools to select the desired region, as shown in Figure 7. For example, to locate and change a specific semantic relationship, Bau *et al.* [97] generalize a linear associative memory [130] to a non-linear convolutional layer of a deep generator. Each layer within a model stores latent rules as a set of key-value relationships over hidden features. They propose a constrained optimization process that can add or edit one specific rule within the associative memory while preserving the existing semantic relationships in the model. As shown in Figure 6, they provide a three-step rewriting process: copy, paste, and context to make model rewriting intuitive for a novice user. Abdal *et al.* [24], [25] analyze the defective image embedding of StyleGAN trained on FFHQ [16], *i.e.*, the embedding of images with 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 the observation,

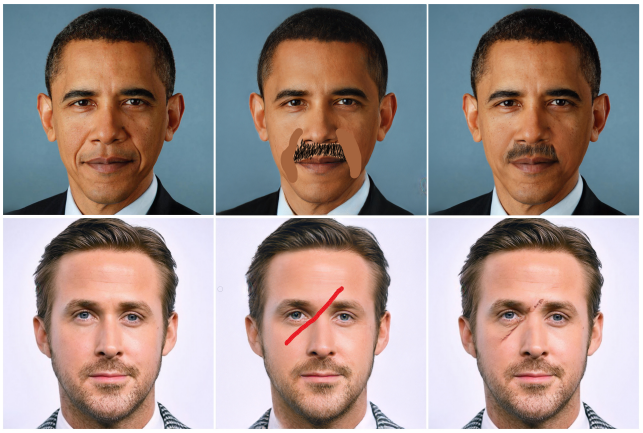

Fig. 7. **Illustration of region-of-interest editing [25].** From left to right: base image; scribbled image; result of local edits.

they develop a masked-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 region-of-interest local editing results of their method can be found in Figure 7 and Figure 11.

4.2.6 Out-of-Distribution

Some GAN inversion methods 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 [131], [132], [133]. This property is a prerequisite for GAN inversion methods to edit real images. In [72], Daras *et al.*show that a local sparse layer (based on local context) can significantly help 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, which did not appear in the training process. Shen *et al.* [21] present the InterFaceGAN model for face editing, which can invert a target image back to a latent code and directly edit the inverted code. They analyze how individual semantic properties are encoded in the latent space and show that a true-or-false facial attribute aligns with a linear subspace of the latent space. By simply modulating the latent code, this method is able to manipulate the gender, age, pose, and expression of given real facial images. In [28], Pan *et al.*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.* [64] introduce 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 8, this method is able to 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

Fig. 8. **Illustration of face image manipulation.** These are real image editing results from using StyleFlow [64].

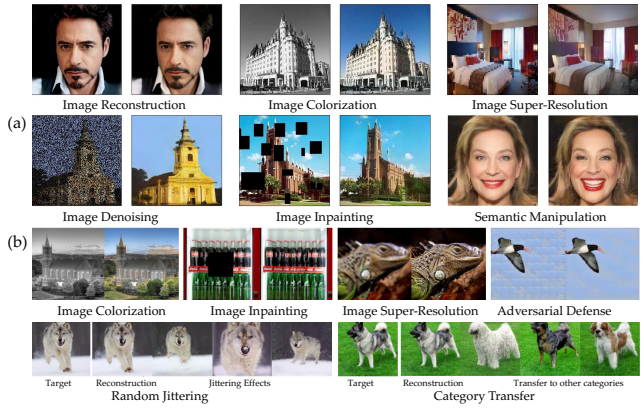

Fig. 9. **Illustration of image processing using GAN inversion.** GAN inversion does not require task-specific dense-labeled datasets and can be applied to many tasks like image reconstruction, image restoration and image manipulation. The upper illustration (a) is from mGAN-Prior [94] and the lower (b) is from DGP [28].

real face images from multiple face datasets [134], [135], [136] but also paintings, caricatures, and black and white photos collected from the Internet. Besides the image, recent methods also show out-of-distribution generalization ability for other modalities, *i.e.*, sketch [103] and text [129].

5 Applications

Finding an accurate solution to the inversion problem allows us to further fine-tune the model weights 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 like image manipulation, image interpolation, image restoration, style tranfer, novel-view synthesis and even adversarial defense, as shown in Figure 9.

5.1 Image Manipulation

Given an image x, we want to edit its certain regions by manipulating its latent codes $\mathbf{z}$ and get 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:

$$x' = G(\mathbf{z}^* + \alpha \mathbf{n}), \tag{30}$$

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 towards a certain direction, the semantics contained in the output image should vary accordingly. For example, Xu *et al.* [111] 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.* [54] determine the direction corresponding to the background removal or background blur gradually 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

Fig. 10. **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].

either decreasing or increasing the semantic degree. Both methods [21], [22] use a projection strategy to search for the semantic direction $\mathbf{n}$.

There are also some methods that can manipulate other information in images, *e.g.*, geometry, texture, and color, than semantics. For example, some [24], [64] can change pose rotation for face manipulation, and others [54] 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 Restoration

Suppose $\hat{x}$ 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 statistics of natural images in a trained GAN model as the priors, *i.e.*, the Deep Generative Prior (DGP). To be specific, they reformulate (13) such that it allows the generator parameters to be fine-tuned on the target image on-the-fly:

$$\theta^*,\, \mathbf{z}^* = \arg\min_{\theta,\,\mathbf{z}} \ell(\hat{x}, \phi(G(\mathbf{z};\theta))). \tag{31}$$

Their method performs visually better than or comparable to state-of-the-art methods on colorization [137], inpainting [138], and super-resolution [139]. While artifacts 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 towards the positive quality direction that defined by a separation hyperplane using a linear SVM [140] (see Figure 10).

5.3 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 done 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]:

$$\mathbf{z} = \lambda \mathbf{z}_A + (1-\lambda)\mathbf{z}_B, \lambda \in (0,1). \tag{32}$$

Abdal *et al.* [24] use linear interpolation operation on vectors by embedding the $\mathcal{W}$ space of StyleGAN into an extended latent space $\mathcal{W}^*$. Similar operation can also be found in [53]. On the other hand, in DGP [28], reconstructing two target images x_A and x_B would result in two generators G_{θ_A} and G_{θ_B}, 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 to both the latent vectors and the generator parameters:

$$\begin{aligned} &\mathbf{z} = \lambda \mathbf{z}_A + (1-\lambda)\mathbf{z}_B, \\ &\theta = \lambda \theta_A + (1-\lambda)\theta_B,\ \lambda \in (0,1), \end{aligned} \tag{33}$$

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

5.4 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 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 non-face painting). The latent codes of the embedded content image are preserved for the first 9 layers (corresponding to 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 spatial resolution from 64^2 to 1024^2). In [25], Abdal *et al.* present a local style transfer method (see Algorithm 1 and Figure 11). 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 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 blurred mask M_b. Finally, they perform the Masked Noise Optimization Mk_n to output the final image. The W_l function only optimizes $\mathbf{w} \in \mathcal{W}^+$ and is given by

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();
$\mathbf{w_o} = W_l(M_b, M_b, 1, \mathbf{w}^*, \mathbf{n_i}, x)$
$\quad + M_{st}(1 - M_b, \mathbf{w}^*, \mathbf{n_i}, y)$;
$\mathbf{n_o} = Mk_n(M_b, \mathbf{w_o}, \mathbf{n_i}, x, G(\mathbf{w_o}))$;

$$\begin{aligned} W_l(M_p, M_m, \mathbf{w_m}, \mathbf{w_i}, \mathbf{n_i}, x) = \arg\min_{\mathbf{w_m}} L_p(M_p, G(\mathbf{w},\mathbf{n}), x) \\ + \|M_m \odot (G(\mathbf{w},\mathbf{n}) - x)\|, \end{aligned} \tag{34}$$

where L_p denotes the perceptual loss [29], $\mathbf{w_i}$ and $\mathbf{n_i}$ are initial variable values, $\mathbf{w_m}$ is a mask for $\mathcal{W}^+$ space, $\odot$ denotes the Hadamard product, and G is the StyleGAN generator. $\mathbf{w_m}$ contains 1s for variables that should be updated and 0s 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

$$M_{st}(M_s, \mathbf{w_i}, \mathbf{n_i}, y) = \arg\min_{\mathbf{w}} L_s(M_s, G(\mathbf{w},\mathbf{n}), y), \tag{35}$$

where L_s is the style loss [141]. 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. They 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).

5.5 Compressive Sensing

Typically, compressive sensing can be formulated as reconstructing an unknown target signal or image $x \in \mathbb{R}^n$

Fig. 11. **Illustration of local style transfer [25].** From left to right: base image, masked region, style image, local style transfer result.

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, $\mathbf{e} \in \mathbb{R}^m$ 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 ill-posed. An alternative way for solution is to obtain an estimate of $\hat{x}$ as the solution to the constrained optimization problem:

$$\begin{aligned} \hat{x} &= \arg\min \ \ell(y; Ax), \\ &\text{s.t.} \quad x \in \mathcal{S}, \end{aligned} \tag{36}$$

where ℓ is the loss function and $\mathcal{S} \subseteq \mathcal{R}^n$ acts as *a priori*. To alleviate the ill-posed nature of the inversion problem (36) 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) [142] or the restricted eigenvalue condition (REC) [143].

Applying GAN inversion to compressive sensing is to estimate the signal as $\hat{x} = G(\hat{\mathbf{z}})$, where $\hat{\mathbf{z}}$ is obtained by minimizing the non-convex cost function

$$f(\mathbf{z}) = \|\mathbf{y} - AG(\mathbf{z})\|_2^2. \tag{37}$$

Bora *et al.* [144] propose to solve (37) using back-propagation and standard gradient-based optimization. Hussein *et al.* [145] handle the limited representation capabilities of the generators by making them image-adaptive (IA) 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

$$f(\theta, \mathbf{z}) = \|y - AG_\theta(\mathbf{z})\|_2^2. \tag{38}$$

In [146], Shah *et al.* present a projected gradient descent (PGD)-based method to solve (37). 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 [146], Raj *et al.* [92] 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.

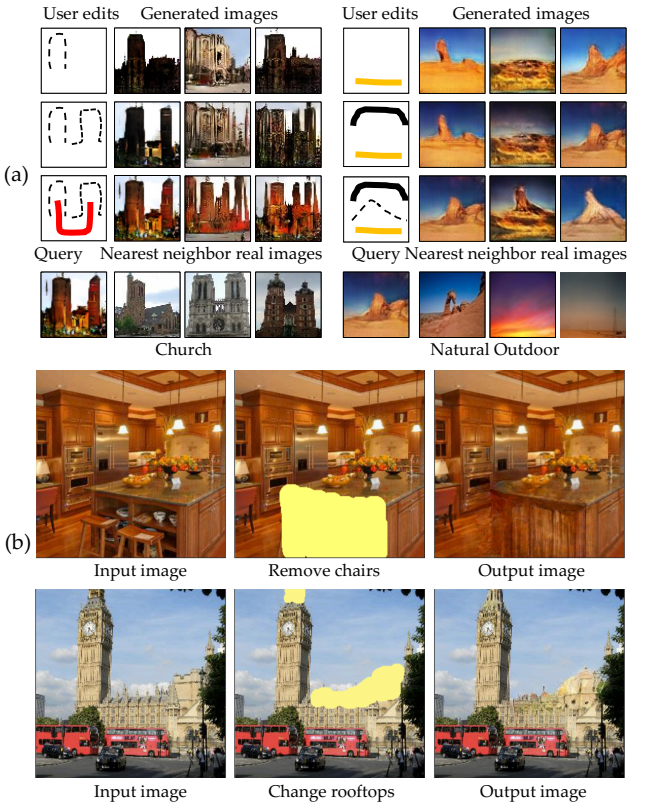

Fig. 12. **Illustration of interactive image generation using GAN inversion.** (a) illustrates results from [23]. The users are allowed to use the brush tools to generate an image from scratch and keep adding more scribbles or sketches for refinement. The last row shows the most similar real images to the generated images. Dashed line represents the sketch tool, and color scribble means the color brush. (b) is from GANPaint [91]. The brushes can draw semantically meaningful units like removing chairs or adding rooftops.

5.6 Other Tasks

5.6.1 *Interactive Generation*

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 use the brush tools to generate an image from scratch and then keep adding more scribbles to refine the result. Bau *et al.* [91] develop a tool (See https://ganpaint.io/demo/) that takes a natural image of a certain class, *e.g.*, church or kitchen, and allows modifications with brushes to draw semantically meaningful units, such as trees or domes. Abdal *et al.* [25] invert the StyleGAN to perform semantic local edits based on user scribbles. With this method, simple scribbles can be converted to photo-realistic edits by embedding into certain layers of StyleGAN. This application is helpful to existing interactive image processing tasks such as sketch-to-image generation [147], [148], [149] and sketch based image retrieval [150], [151], which usually require dense-labeled datasets.

5.6.2 *Semantic Diffusion*

Semantic image diffusion is an image editing task, which inserts the target face to the context and makes them com-

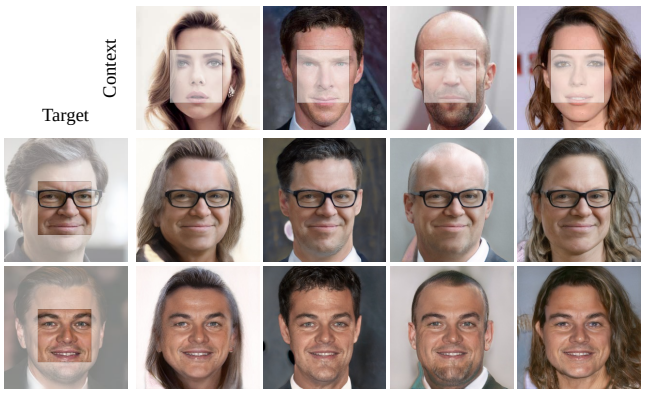

Fig. 13. **Semantic diffusion results using the in-domain GAN inversion method [22]**. Target images in the first column are naturally diffused into context images in the first row with the identify preserved.

patible, as illustrated in Figure 13. It can be seen as a variant of image harmonization [10], [152], [153]. 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. On the other hand, Xu *et al.* [111] 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.

5.6.3 Category Transfer

In Section 5.2, 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.

5.6.4 Adversarial Defense

Adversarial attack methods [154], [155], [156], [157] aim at fooling a CNN classifier by adding a certain perturbation Δx to a target image x. In contrast, adversarial defense [158], [159] aims at preventing the model from being fooled by attackers. If considering the degradation transform of 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 5.2 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.

6 Relevant Problems

6.1 Deep Generative Models

Aside from GANs, numerous generative models have been developed including Deep Boltzmann Machines (DBMs) [160], [161], [162], [163], Variational Autoencoders (VAEs) [164], [165], Deep Autoregressive Models (DARs) [166], [167], [168], [169], [170], and Normalizing Flow Models (NFMs) [171], [172], [173], [174]. A DBM is a binary Markov Random Field (MRF) with multiple layers of hidden random variables. DBMs can learn complex and abstract representations by first training on limited, labeled data and then fine-tuning on a large-scale unlabeled dataset. GANs, VAEs, DARs and NFMs are latent variable models which represent high-dimensional data x using lower-dimensional latent variables $\mathbf{z}$. Different from previous autoencoder models [175], [176] mapping the input x into a fixed vector, VAEs [164] map x into a latent vector $\mathbf{z}$ with prior distribution $p_\theta(\mathbf{z})$, parameterized by θ. The optimal parameter θ^* would be $\theta^* = \arg\max_\theta \sum_{i=1}^n \log p_\theta(x^{(i)})$, which can be solved by optimizing the Kullback–Leible (KL)-divergence of the estimated posterior distribution $q_\phi(\mathbf{z}|x)$ (defining encoder) and the intractable real posterior $p_\theta(\mathbf{z}|x)$ (defining decoder $p_\theta(x|\mathbf{z})$), $D_{\mathrm{KL}}(q_\phi(\mathbf{z}|x)\|p_\theta(\mathbf{z}|x))$ with respect to ϕ, by maximizing the evidence lower bound (ELBO). DARs [166], [167] are feed-forward models which give predictive probability distribution $P(x_{t+1}|x_1, x_2, \cdots, x_t)$ for future values x_{t+1} from past observations $x_1, x_2, \cdots, x_t$. DARs work flexibly on both continuous and discrete signals, including audio (WaveNet [169]), images (PixelCNN++ [170]) and text (Transformer [167]). Different from GANs and VAEs, flow-based deep generative models explicitly learn the probability density function of input x by a sequence of invertible transformations: $x = \mathbf{z}_K = f_K \circ f_{K-1} \circ \cdots \circ f_1(\mathbf{z}_0)$. The path traversed by the random variables $\mathbf{z}_i = f_i(\mathbf{z}_{i-1})$ is the flow and the full chain formed by the successive distributions π_i is called a normalizing flow. A transformation function f_i should satisfy two requirements: efficient inversion and tractable Jacobian determinant calculation. Furthermore, if f_i is framed as an autoregressive model, *i.e.*, each dimension in a vector variable is conditioned on the previous dimensions like ARMA [177] or ARCH [178], this is an autoregressive flow. Classic autoregressive flow models include MAF [179] and IAF [180].

6.2 Adversarial Feature Learning

Similar to GAN inversion, Adversarial Feature Learning (AFL, also known as BiGAN) [116], [181], [182] or Adversarially Learned Inference (ALI) [117], [183], [184] also aims to learn the inverse mapping of GAN models[1]. Different from GAN inversion, which interprets the latent space of a trained GAN model, AFL or ALI trains a generation network and an inference network *jointly* in an adversarial manner. This kind of deep generative model also constitutes a novel approach to integrating efficient inference with the GANs framework. Unlike other deep generative models (like Variational Autoencoders (VAEs) [185] or Generative Latent Optimization (GLO) [186]), the objective function of AFL or ALI involves no explicit constraints during reconstruction. Instead of a pixel-perfect reconstruction, they tend to produce believable reconstructions with flexible variations, albeit at the expense of making some mistakes in capturing exact object placement, color, style or even identity. This novel unsupervised feature learning framework

1. ALI [117] and AFL [116] share the same idea but are developed by two teams independently.

led to the emergence of many applications, especially cross-domain retrieval, detection and recognition since it learns invariant features. For example, Li *et al.* [181] use adversarial autoencoder [187] to learn invariant features for domain generalization. Xie *et al.* [182] propose a framework and provide theoretical analysis to learn invariant representations of data, which can be applied to multiple applications including text generation and image classification. Li *et al.* [188] describe the non-identifiability issue of ALI and provide a unified framework for recently proposed GAN models, including CGAN [189], ALI [117], AFL [116] and CycleGAN [190], from the perspective of joint distribution matching. Pang *et al.* [191] introduce a sketch Re-ID problem and address it by proposing a cross-domain adversarial feature learning method, which can jointly learn identity and domain invariant features. Kim *et al.* [192] propose a novel unsupervised learning framework for cross-spectral pedestrian detection. They apply an adversarial learning scheme to intermediate features of the color and thermal images based on the assumption that images from both domains share their characteristics in a common feature space. Donahue *et al.* [193] propose BigBiGAN (BiGAN with BigGAN generator) for unconditional image generation by adopting the generator and discriminator architectures from the BigGAN model, which shows generative models and inference model benefit from each other.

6.3 Deep Feature Factorization

Similar to GAN inversion, deep feature factorization (DFF) [194] is also able to locate semantic concepts in individual images and across image sets. DFF is the application of non-negative matrix factorization (NMF) [195], [196] to the ReLU feature activations of a pretrained neural network. NMF has previously been shown to be a useful decomposition for multivariate data. It can be understood as factorizing a non-negative data matrix subject to different constraints like Principal Components Analysis (PCA) or Vector Quantization (VQ). PCA enforces a weak orthogonality constraint, resulting in a distributed representation that uses cancellations to capture global variability [197], [198], while VQ uses a strong winner-take-all constraint that leads to clustering the data into mutually exclusive prototypes [199]. In contrast, NMF features are more interpretable. Taking face recognition for example, PCA learns eigenfaces that resemble distorted versions of the whole faces; VQ learns holistic prototypes, each being a whole face; NMF learns local feature representation corresponding to different face details. Many modern neural networks use the rectified linear activation function (ReLU) [87], $\max(x, 0)$, due to its desirable properties for training. NMF is naturally applicable for this case as ReLU results in non-negative activations. DFF is first proposed by Collins *et al.* [194] for concept discovery and produces state-of-the-art results on co-segmentation and co-localization tasks. The authors claim that the returned k factors, where k is the predefined rank of the approximation, correspond to coherent objects or object-parts. They further propose to use DFF to achieve content-based image retrieval and localization [200]. Similar ideas are also proposed to conduct analysis of the activations of generative models. Collins *et al.* [95] apply spherical k-means clustering [201] to the activation vectors that make up the activation tensor at a given layer of generative models. Though Ramesh *et al.* [119] and Voynov *et al.* [54] use Jacobian Decomposition to investigate the latent space of a pretrained GAN model while Härkönen *et al.* [101] use PCA, both of which seem similar to the aforementioned methods [95], [194], [200], we still do not categorize the DDF family as GAN inversion methods since they have no iteration or optimization process.

6.4 Deep Image Prior

Image prior describes statistics of natural images. It has been widely adopted in computer vision tasks, including MRF [202], [203], [204], dark channel prior [205] and total variation regularizer [206], which all model correlation among neighboring pixels via Gibbs distribution. There are also other deep priors developed for low-level restoration tasks like deep denoiser prior [207], [208] and TNRD [209]. These image priors capture low-level statistics and are typically used in denoising, inpainting, and segmentation. Recently, deep image prior (DIP) [138] presents that image statistics implicitly captured by the structure of a randomly-initialized neural network could also be used as prior to restore or translate images. DIP is shown to capture a great deal of low-level image statistics prior and can perform excellent results in standard inverse problems such as denoising, super-resolution and inpainting. SinGAN [139] fine-tunes a randomly initialized GAN on patches of a single image in different scales, achieving various image manipulation or restoration effects. However, SinGAN requires the target image to have rich repeated patterns, for it is biased towards capturing low-level and mid-level textures. Despite of excellent performance in some cases, DIP and SinGAN have limited access to image statistics beyond the input image since they are trained from scratch, which restrains their applicability in tasks such as image colorization. Some other studies [210], [211] propose to search for neural architectures that capture stronger image priors instead of using hand-designed architectures. Specifically, Ho *et al.* [210] automatically optimize the encoder-decoder structure using evolutionary search and meta-parameters of the DIP network, which serves as a content-specific prior to regularize image restoration. Chen *et al.* [211] search for an improved network architecture by leveraging reinforcement learning with a recurrent neural network controller.

The aforementioned approaches of image prior train their networks from scratch. There are also some attempts that use a pretrained generative model as a source of image statistics [18], [145]. For example, Lin *et al.* [212] propose LaDDer, a method for accurately modeling the prior distribution in a VAE framework. Bau *et al.* [18] adopt a GAN to manipulate partial areas of an image, but the proposed method is not applicable to restoration tasks like colorization. Another work [145] also use GAN prior for restoration, but is only applicable for compressed sensing and face hallucination. A concurrent work of multi-code GAN prior (mGANPrior) [94] also conducts image processing by solving the GAN inversion problem. They study and exploit how the priors captured in GAN contribute for versatile restoration and manipulation tasks. The significant

differences between GAN inversion and GAN Prior are determined by using a pretrained model or inverting images to latent space.

7 Challenges and Future Directions

Theoretical Understanding. Despite its success in applications, there still lacks 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 [101]. Nonlinear structure in data can be represented compactly, and the induced geometry necessitates the use of nonlinear statistical tools [213], Riemanian manifold, 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 non-linear Factor Analysis (FA) [214], Independent Component Analysis (ICA) [215], Latent Dirichlet Allocation (LDA) [216], [217] to decompose the network weight and to find interpretable directions of latent space.

Inversion Type. Besides GANs inversion, some methods have been developed to invert generative models based on the encoder-decoder architecture. The IIN method [80] learns invertible disentangled interpretations of variational auto-encoders (VAE) [164]. Zhu *et al.* [218] develop the latently invertible auto-encoder method to learn a disentangled representation of face images, from which contents can be edited based on attributes. The LaDDer approach [212] uses a meta-embedding based on generative prior (including an additive VAE and a mixture of hyper prior) 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 proved to be effective in cross-domain applications such as style transfer and image restoration, which indicates that the 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. Multi-task methods have been developed to collaboratively exploit visual cues, such as image restoration and image segmentation [219], or semantic segmentation and depth estimation [220], [221], 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 [54], [64] 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. The implicit neural representation learning [222], [223], [224], a recent trend in 3D community, is to learn 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 [225], [226], [227], novel-view synthesis [228], [228], face shape generation [229], object modeling [230], [231], and human reconstruction [232], [233], [234], [235]. The recent StyleRig method [96] is trained based on the semantic parameters of 3D Morphable Model (3DMM) [236] 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], [64]. However, there is also some tuning required to achieve the desired granularity of precise fine-grained control, *e.g.*, gaze redirection [7], [237], [238], [239], relighting [240], [241], [242] and continuous view control [243]. 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 need to be made 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 are primarily about images. However, recent advances in generative models are beyond the image domain, such as the GPT-3 language model [244] and WaveNet for audio synthesis [169]. 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 like language style transfer. Furthermore, there are also GAN models for multi-modality generation or translation [70], [245], [246]. It can be substantially rewarding to invert such GAN models as multi-modal representations for creating novel kinds of content, behavior, and interaction.

Evaluation Metrics. The perceptual quality metrics, which can better evaluate photo-realistic and diversity images or consistent identity to the original image, remain to be explored. Furthermore, the evaluations mostly concentrate on photo-realism, or judge if the distribution of generated images is consistent with the real images with regard to classification [26] or segmentation [54] accuracy using models trained for real images. However, there is still a lack of effective assessment tools to evaluate the difference between directions of the predicted results and the expected ones, or measuring the inverted latent codes more directly.

8 Conclusion

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. Lastly we discuss some challenges and future directions for GAN inversion, and we hope this paper would inspire future research to solve them.

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

[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, Z. Cheng, Y. Yang, J.-H. Xue, and W. Feng, "Controllable continuous gaze redirection," in *ACM MM*, 2020. 1, 4, 13, 17

[8] B. Li, X. Qi, T. Lukasiewicz, and P. H. Torr, "Manigan: Text-guided image manipulation," in *CVPR*, 2020. 1, 4

[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, pp. 3789–3797. 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, 4, 6, 7, 12, 13

[15] A. Brock, J. Donahue, and K. Simonyan, "Large scale GAN training for high fidelity natural image synthesis," in *ICLR*, 2019. 1, 2, 4, 6, 7, 8

[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, 10, 11, 13, 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, 2, 4, 5, 6, 7, 10

[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, 16

[19] L. Goetschalckx, A. Andonian, A. Oliva, and P. Isola, "Ganalyze: Toward visual definitions of cognitive image properties," in *ICCV*, 2019. 1, 8

[20] A. Jahanian, L. Chai, and P. Isola, "On the "steerability" of generative adversarial networks," in *ICLR*, 2020. 1, 6, 8, 9

[21] Y. Shen, J. Gu, X. Tang, and B. Zhou, "Interpreting the latent space of GANs for semantic face editing," in *CVPR*, 2020. 1, 4, 6, 8, 10, 11, 12, 13, 17

[22] J. Zhu, Y. Shen, D. Zhao, and B. Zhou, "In-domain gan inversion for real image editing," in *ECCV*, 2020. 1, 4, 6, 7, 9, 11, 12, 15

[23] J.-Y. Zhu, P. Krähenbühl, E. Shechtman, and A. A. Efros, "Generative visual manipulation on the natural image manifold," in *ECCV*, 2016. 1, 5, 6, 7, 11, 14

[24] R. Abdal, Y. Qin, and P. Wonka, "Image2StyleGAN: How to embed images into the StyleGAN latent space?" in *ICCV*, 2019. 1, 6, 7, 11, 12, 13

[25] ——, "Image2StyleGAN++: How to edit the embedded images?" in *CVPR*, 2020. 1, 6, 7, 11, 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, 6, 7, 9, 10, 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, 6, 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, 6, 11, 12, 13, 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, 6, 13

[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, 3, 8

[31] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in *ICLR*, 2016. 2, 4, 5, 6, 7

[32] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *NeurIPS*, 2017, pp. 5767–5777. 2, 6

[33] Y. Jin, J. Zhang, M. Li, Y. Tian, and H. Zhu, "Towards the high-quality anime characters generation with generative adversarial networks," in *Proceedings of the Machine Learning for Creativity and Design Workshop at NeurIPS*, 2017. 2, 6

[34] 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. 2

[35] 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. 2, 6

[36] 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. 2, 6

[37] X. Huang and S. Belongie, "Arbitrary style transfer in real-time with adaptive instance normalization," in *ICCV*, 2017. 2, 4, 6

[38] A. Gabbay and Y. Hoshen, "Style generator inversion for image enhancement and animation," *arXiv preprint arXiv:1906.11880*, 2019. 2

[39] P. Zhu, R. Abdal, Y. Qin, and P. Wonka, "Sean: Image synthesis with semantic region-adaptive normalization," in *CVPR*, 2020, pp. 5104–5113. 2

[40] Z. Zhu, Z. Xu, A. You, and X. Bai, "Semantically multi-modal image synthesis," in *CVPR*, 2020, pp. 5467–5476. 2

[41] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *ICCV*, 2015. 2, 6

[42] C.-H. Lee, Z. Liu, L. Wu, and P. Luo, "Maskgan: Towards diverse and interactive facial image manipulation," in *CVPR*, 2020. 2

[43] 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. 2

[44] Y. LeCun, "The mnist database of handwritten digits," *http://yann. lecun. com/exdb/mnist/*, 1998. 2, 6

[45] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv preprint arXiv:1708.07747*, 2017. 2

[46] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," *Technical Report*, 2009. 2

[47] 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. 2, 6

[48] A. Yu and K. Grauman, "Semantic jitter: Dense supervision for visual comparisons via synthetic images," in *ICCV*, 2017, pp. 5571–5580. 2

[49] Z. Liu, S. Yan, P. Luo, X. Wang, and X. Tang, "Fashion landmark detection in the wild," in *ECCV*, 2016, pp. 229–245. 2

[50] 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. 2

[51] 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. 2

[52] 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. 2, 6

[53] Y. Nitzan, A. Bermano, Y. Li, and D. Cohen-Or, "Face identity disentanglement via latent space mapping," *TOG*, 2020. 2, 4, 6, 10, 13

[54] A. Voynov and A. Babenko, "Unsupervised discovery of interpretable directions in the gan latent space," *ICML*, 2020. 2, 3, 6, 7, 9, 12, 16, 17

[55] 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. 3

[56] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training GANs," in *NeurIPS*, 2016. 3

[57] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *CVPR*, 2016, pp. 2818–2826. 3

[58] 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. 3

[59] 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. 3

[60] 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. 3

[61] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014. 3, 6

[62] K. Shoemake, "Animating rotation with quaternion curves," in *SIGGRAPH*, 1985, pp. 245–254. 3

[63] T. White, "Sampling generative networks: Notes on a few effective techniques," *arXiv preprint arXiv:1609.04468*, 2016. 3

[64] A. Rameen, Z. Peihao, M. Niloy, and W. Peter, "Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows," *TOG*, 2021. 4, 6, 8, 11, 12, 17

[65] A. Geitgey, "Github - face recognition," 2020. [Online]. Available: https://github.com/ageitgey/face_recognition 4

[66] D. E. King, "Dlib-ml: A machine learning toolkit," *JMLR*, 2009. 4

[67] 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. 4, 6, 10

[68] Z. Wu, D. Lischinski, and E. Shechtman, "Stylespace analysis: Disentangled controls for stylegan image generation," *arXiv preprint arXiv:2011.12799*, 2020. 4, 6, 10

[69] X. Tao, Z. Pengchuan, H. Qiuyuan, Z. Han, G. Zhe, X. Huang, and H. Xiaodong, "Attngan: Fine-grained text to image generation with attentional generative adversarial networks," in *CVPR*, 2018. 4

[70] B. Li, X. Qi, T. Lukasiewicz, and P. H. S. Torr, "Controllable text-to-image generation," in *NeurIPS*, 2019. 4, 17

[71] A. Nguyen, A. Dosovitskiy, J. Yosinski, T. Brox, and J. Clune, "Synthesizing the preferred inputs for neurons in neural networks via deep generator networks," in *NeurIPS*, 2016, pp. 3387–3395. 4

[72] 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. 4, 6, 11

[73] 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. 4, 5, 7

[74] W. Xia, Y. Yang, J.-H. Xue, and J. Xiao, "Domain fingerprints for no-reference image quality assessment," *IEEE Transactions on Circuits and Systems for Video Technology*, 2020. 5

[75] Z. Li, R. Tao, H. Niu, and B. Li, "Interpreting the latent space of gans via correlation analysis for controllable concept manipulation," *arXiv preprint arXiv:2006.10132*, 2020. 5

[76] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *JMLR*, vol. 9, pp. 2579–2605, 2008. 5

[77] A. H. Liu, Y. Liu, Y. Yeh, and Y. F. Wang, "A unified feature disentangler for multi-domain image translation and manipulation," in *NeurIPS*, 2018, pp. 2595–2604. 5

[78] B. Lu, J.-C. Chen, and R. Chellappa, "Unsupervised domain-specific deblurring via disentangled representations," *CVPR*, 2019. 5

[79] A. Creswell and A. A. Bharath, "Inverting the generator of a generative adversarial network," *TNNLS*, 2018. 5, 6, 7

[80] P. Esser, R. Rombach, and B. Ommer, "A disentangling invertible interpretation network for explaining latent representations," in *CVPR*, 2020. 5, 17

[81] A. Bora, A. Jalal, E. Price, and A. G. Dimakis, "Compressed sensing using generative models," in *ICML*, vol. 70, 2017, pp. 537–546. 5

[82] A. C. Gilbert, Y. Zhang, K. Lee, Y. Zhang, and H. Lee, "Towards understanding the invertibility of convolutional neural networks," in *IJCAI*, 2017, pp. 1703–1710. 5

[83] F. Ma, U. Ayaz, and S. Karaman, "Invertibility of convolutional generative networks from partial measurements," in *NeurIPS*, 2018, pp. 9651–9660. 5, 7

[84] A. Aberdam, D. Simon, and M. Elad, "When and how can deep generative models be inverted?" *arXiv preprint arXiv:2006.15555*, 2020. 5, 6, 9, 10

[85] P. Hand and V. Voroninski, "Global guarantees for enforcing deep generative priors by empirical risk," *IEEE Transactions on Information Theory*, vol. 66, no. 1, pp. 401–418, 2020. 5

[86] W. Shang, K. Sohn, D. Almeida, and H. Lee, "Understanding and improving convolutional neural networks via concatenated rectified linear units," in *ICML*, 2016, pp. 2217–2225. 5

[87] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *ICML*, 2010. 5, 10, 16

[88] A. Yu and K. Grauman, "Fine-grained visual comparisons with local learning," in *CVPR*, 2014, pp. 192–199. 6

[89] 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. 6

[90] 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. 6

[91] 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. 6, 14

[92] 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. 6, 14

[93] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," in *ICML*, 2019, pp. 7354–7363. 6

[94] J. Gu, Y. Shen, and B. Zhou, "Image processing using multi-code gan prior," in *CVPR*, 2020. 6, 12, 16

[95] C. Edo, B. Raja, P. Bob, and S. Sabine, "Editing in style: Uncovering the local semantics of gans," in *CVPR*, 2020. 6, 9, 16

[96] 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. 6, 7, 17

[97] D. Bau, S. Liu, T. Wang, J.-Y. Zhu, and A. Torralba, "Rewriting a deep generative model," in *ECCV*, 2020. 6, 11

[98] V. Yuri and K. Vladimir, Ivashkin ang Evgeny, "Stylegan2 distillation for feed-forward image manipulation," in *ECCV*, 2020. 6, 10

[99] 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. 6

[100] 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. 6

[101] H. Erik, H. Aaron, L. Jaakko, and P. Sylvain, "Ganspace: Discovering interpretable gan controls," in *NeurIPS*, 2020. 6, 9, 16, 17

[102] 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. 6, 7, 10

[103] E. Richardson, Y. Alaluf, O. Patashnik, Y. Nitzan, Y. Azar, S. Shapiro, and D. Cohen-Or, "Encoding in style: a style-

gan encoder for image-to-image translation," *arXiv preprint arXiv:2008.00951*, 2020. 6, 12

[104] 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. 6, 9

[105] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *ICLR*, 2018. 6

[106] A. Cherepkov, A. Voynov, and A. Babenko, "Navigating the gan parameter space for semantic image editing," *arXiv preprint arXiv:2011.13786*, 2020. 6, 9

[107] S. Nurit, B. Ron, and M. Tomer, "Gan steerability without optimization," in *ICLR*, 2021. 6, 8, 9

[108] P. Zhuang, O. Koyejo, and A. G. Schwing, "Enjoy your editing: Controllable gans for image editing via latent space navigation," in *ICLR*, 2021. 6

[109] L. Chai, J. Wulff, and P. Isola, "Using latent space regression to analyze and leverage compositionality in gans." in *ICLR*, 2021. 6

[110] Y. Shen and B. Zhou, "Closed-form factorization of latent semantics in gans," in *CVPR*, 2021. 6, 8, 9

[111] Y. Xu, Y. Shen, J. Zhu, C. Yang, and B. Zhou, "Generative hierarchical features from synthesizing images," in *CVPR*, 2021. 6, 12, 15

[112] H.-P. Wang, N. Yu, and M. Fritz, "Hijack-gan: Unintended-use of pretrained, black-box gans," in *CVPR*, 2021. 6

[113] O. Tov, Y. Alaluf, Y. Nitzan, O. Patashnik, and D. Cohen-Or, "Designing an encoder for stylegan image manipulation," *arXiv preprint arXiv:2102.02766*, 2021. 6

[114] 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. 6

[115] G. Perarnau, J. Van De Weijer, B. Raducanu, and J. M. Álvarez, "Invertible conditional gans for image editing," *arXiv preprint arXiv:1611.06355*, 2016. 5, 6

[116] J. Donahue, P. Krähenbühl, and T. Darrell, "Adversarial feature learning," *arXiv preprint arXiv:1605.09782*, 2016. 7, 15, 16

[117] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville, "Adversarially learned inference," *arXiv preprint arXiv:1606.00704*, 2016. 7, 15, 16

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

[119] A. Ramesh, Y. Choi, and Y. LeCun, "A spectral regularizer for unsupervised disentanglement," *arXiv preprint arXiv:1812.01161*, 2018. 7, 16

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

[121] 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

[122] S. Duane, A. D. Kennedy, B. J. Pendleton, and D. Roweth, "Hybrid monte carlo," *Physics letters B*, vol. 195, no. 2, pp. 216–222, 1987. 7

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

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

[125] A. Plumerault, H. L. Borgne, and C. Hudelot, "Controlling generative models with continuous factors of variations," *ICLR*, 2020. 8

[126] Q. Lei, A. Jalal, I. S. Dhillon, and A. G. Dimakis, "Inverting deep generative models, one layer at a time," in *NeurIPS*, 2019, pp. 13 910–13 919. 9, 10

[127] 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. 10

[128] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *ICML*, 2013. 10

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

[130] T. Kohonen and M. Ruohonen, "Representation of associated data by matrix operators," *IEEE Transactions on Computers*, vol. 22, no. 7, pp. 701–702, 1973. 11

[131] 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. 11

[132] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," *arXiv preprint arXiv:1610.02136*, 2016. 11

[133] 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. 11

[134] 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. 12

[135] 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. 12

[136] 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. 12

[137] G. Larsson, M. Maire, and G. Shakhnarovich, "Learning representations for automatic colorization," in *ECCV*, 2016, pp. 577–593. 13

[138] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," in *CVPR*, 2018, pp. 9446–9454. 13, 16

[139] 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, 16

[140] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995. 13

[141] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in *CVPR*, 2016. 13

[142] E. J. Candès *et al.*, "Compressive sampling," in *ICM*, vol. 3, 2006, pp. 1433–1452. 14

[143] D. L. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006. 14

[144] A. Bora, A. Jalal, E. Price, and A. G. Dimakis, "Compressed sensing using generative models," *arXiv preprint arXiv:1703.03208*, 2017. 14

[145] S. A. Hussein, T. Tirer, and R. Giryes, "Image-adaptive gan based reconstruction," *arXiv preprint arXiv:1906.05284*, 2019. 14, 16

[146] V. Shah and C. Hegde, "Solving linear inverse problems using gan priors: An algorithm with provable guarantees," in *ICASSP*, 2018, pp. 4609–4613. 14

[147] 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. 14

[148] 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. 14

[149] S.-Y. Chen, W. Su, L. Gao, S. Xia, and H. Fu, "DeepFaceDrawing: Deep generation of face images from sketches," *TOG*, vol. 39, no. 4, pp. 72:1–72:16, 2020. 14

[150] 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. 14

[151] 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. 14

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

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

[154] 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

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

[156] 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

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

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

[159] 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

[160] R. Salakhutdinov and G. Hinton, "Deep boltzmann machines," in *AISTATS*, 2009, pp. 448–455. 15

[161] R. Salakhutdinov and H. Larochelle, "Efficient learning of deep boltzmann machines," in *AISTATS*, 2010, pp. 693–700. 15

[162] G. Montavon and K.-R. Müller, "Deep boltzmann machines and the centering trick," in *Neural Networks: Tricks of the Trade*. Springer, 2012, pp. 621–637. 15

[163] N. Srivastava and R. R. Salakhutdinov, "Multimodal learning with deep boltzmann machines," in *NeurIPS*, 2012, pp. 2222–2230. 15

[164] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *ICLR*, 2013. 15, 17

[165] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, "beta-vae: Learning basic visual concepts with a constrained variational framework," in *ICLR*, 2017. 15

[166] M. Germain, G. Karol, M. Iain, and L. Hugo, "Made: Masked autoencoder for distribution estimation," in *ICML*, 2015. 15

[167] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *NeurIPS*, 2017, pp. 5998–6008. 15

[168] A. v. d. Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel recurrent neural networks," *arXiv preprint arXiv:1601.06759*, 2016. 15

[169] 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. 15, 17

[170] T. Salimans, A. Karpathy, X. Chen, and D. P. Kingma, "PixelCNN++: Improving the PixelCNN with discretized logistic mixture likelihood and other modifications," in *ICLR*, 2017. 15

[171] L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density estimation using real NVP," in *ICLR*, 2017. 15

[172] L. Dinh, D. Krueger, and Y. Bengio, "NICE: Non-linear independent components estimation," in *ICLR*, 2015. 15

[173] D. P. Kingma and P. Dhariwal, "Glow: Generative flow with invertible 1x1 convolutions," in *NeurIPS*, 2018, pp. 10 215–10 224. 15

[174] A. Lugmayr, M. Danelljan, L. Van Gool, and R. Timofte, "Srflow: Learning the super-resolution space with normalizing flow," in *ECCV*, 2020. 15

[175] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio, "Contractive auto-encoders: Explicit invariance during feature extraction," in *ICML*, 2011. 15

[176] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *ICML*, 2008, pp. 1096–1103. 15

[177] G. E. Box, G. M. Jenkins, and G. C. Reinsel, *Time series analysis: forecasting and control*. John Wiley & Sons, 2011, vol. 734. 15

[178] R. F. Engle, "Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation," *Econometrica: Journal of the Econometric Society*, pp. 987–1007, 1982. 15

[179] G. Papamakarios, T. Pavlakou, and I. Murray, "Maf: Masked autoregressive flow for density estimation," in *NeurIPS*, 2017, pp. 2338–2347. 15

[180] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling, "Iaf: Improved variational inference with inverse autoregressive flow," in *NeurIPS*, 2016, pp. 4743–4751. 15

[181] H. Li, S. Jialin Pan, S. Wang, and A. C. Kot, "Domain generalization with adversarial feature learning," in *CVPR*, 2018, pp. 5400–5409. 15, 16

[182] Q. Xie, Z. Dai, Y. Du, E. Hovy, and G. Neubig, "Controllable invariance through adversarial feature learning," in *NeurIPS*, 2017, pp. 585–596. 15, 16

[183] C. Du, C. Du, X. Xie, C. Zhang, and H. Wang, "Multi-view adversarially learned inference for cross-domain joint distribution matching," in *ACM KDD*, 2018, pp. 1348–1357. 15

[184] M. I. Belghazi, S. Rajeswar, O. Mastropietro, N. Rostamzadeh, J. Mitrovic, and A. Courville, "Hierarchical adversarially learned inference," *arXiv preprint arXiv:1802.01071*, 2018. 15

[185] C. Doersch, "Tutorial on variational autoencoders," *arXiv preprint arXiv:1606.05908*, 2016. 15

[186] P. Bojanowski, A. Joulin, D. Lopez-Paz, and A. Szlam, "Optimizing the latent space of generative networks," *arXiv preprint arXiv:1707.05776*, 2017. 15

[187] A. Makhzani, J. Shlens, N. Jaitly, and I. J. Goodfellow, "Adversarial autoencoders," *arXiv preprint arXiv:1511.05644*, 2015. 16

[188] C. Li, H. Liu, C. Chen, Y. Pu, L. Chen, R. Henao, and L. Carin, "Alice: Towards understanding adversarial learning for joint distribution matching," in *NeurIPS*, 2017, pp. 5495–5503. 16

[189] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014. 16

[190] J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, and E. Shechtman, "Toward multimodal image-to-image translation," in *NeurIPS*, 2017. 16

[191] L. Pang, Y. Wang, Y. Song, T. Huang, and Y. Tian, "Cross-domain adversarial feature learning for sketch re-identification," in *ACM MM*, 2018, pp. 609–617. 16

[192] M. Kim, S. Joung, K. Park, S. Kim, and K. Sohn, "Unpaired cross-spectral pedestrian detection via adversarial feature learning," in *ICIP*. IEEE, 2019, pp. 1650–1654. 16

[193] J. Donahue and K. Simonyan, "Large scale adversarial representation learning," in *NIPS*, 2019, pp. 10 542–10 552. 16

[194] E. Collins, R. Achanta, and S. Susstrunk, "Deep feature factorization for concept discovery," in *ECCV*, vol. 14, 2018, pp. 352–368. 16

[195] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *NeurIPS*, 2001, pp. 556–562. 16

[196] C.-J. Lin, "Projected gradient methods for nonnegative matrix factorization," *Neural Computation*, vol. 19, no. 10, pp. 2756–2779, 2007. 16

[197] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics and Intelligent Laboratory Systems*, vol. 2, no. 1-3, pp. 37–52, 1987. 16

[198] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71–86, 1991. 16

[199] A. Gersho and R. M. Gray, *Vector quantization and signal compression*. Springer Science & Business Media, 2012, vol. 159. 16

[200] E. Collins and S. Susstrunk, "Deep feature factorization for content-based image retrieval and localization," in *ICIP*, 2019, pp. 874–878. 16

[201] C. Buchta, M. Kober, I. Feinerer, and K. Hornik, "Spherical k-means clustering," *Journal of Statistical Software*, vol. 50, no. 10, pp. 1–22, 2012. 16

[202] S. Roth and M. J. Black, "Fields of experts: A framework for learning image priors," in *CVPR*, 2005, pp. 860–867. 16

[203] S. C. Zhu and D. Mumford, "Prior learning and gibbs reaction-diffusion," *TPAMI*, vol. 19, no. 11, pp. 1236–1250, 1997. 16

[204] S. Geman and D. Geman, "Stochastic relaxation, gibbs distributions, and the bayesian restoration of images," *TPAMI*, no. 6, pp. 721–741, 1984. 16

[205] K. He, J. Sun, and X. Tang, "Single image haze removal using dark channel prior," *TPAMI*, vol. 33, no. 12, pp. 2341–2353, 2010. 16

[206] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: Nonlinear Phenomena*, 1992. 16

[207] K. Zhang, W. Zuo, S. Gu, and L. Zhang, "Learning deep cnn denoiser prior for image restoration," in *CVPR*, 2017, pp. 3929–3938. 16

[208] S. A. Bigdeli, M. Zwicker, P. Favaro, and M. Jin, "Deep mean-shift priors for image restoration," in *NeurIPS*, 2017, pp. 763–772. 16

[209] Y. Chen and T. Pock, "Trainable nonlinear reaction diffusion: A flexible framework for fast and effective image restoration," *TPAMI*, vol. 39, no. 6, pp. 1256–1272, 2016. 16

[210] K. Ho, A. Gilbert, H. Jin, and J. Collomosse, "Neural architecture search for deep image prior," *arXiv preprint arXiv:2001.04776*, 2020. 16

[211] Y.-C. Chen, C. Gao, E. Robb, and J.-B. Huang, "Nas-dip: Learning deep image prior with neural architecture search," in *ECCV*, 2020. 16

[212] L. Shuyu and C. Ronald, "Ladder: Latent data distribution modelling with a generative prior," in *BMVC*, 2020. 16, 17

[213] L. Kuhnel, T. Fletcher, S. Joshi, and S. Sommer, "Latent space non-linear statistics," *arXiv preprint arXiv:1805.07632*, 2018. 17

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

[215] M. E. Davies and C. J. James, "Source separation using single channel ica," *Signal Processing*, vol. 87, no. 8, pp. 1819–1832, 2007. 17

[216] M. Hoffman, F. R. Bach, and D. M. Blei, "Online learning for latent dirichlet allocation," in *NeurIPS*, 2010, pp. 856–864. 17

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

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

[219] 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

[220] 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*. IEEE, 2019, pp. 7101–7107. 17

[221] W. Zhan, X. Ou, Y. Yang, and L. Chen, "Dsnet: Joint learning for scene segmentation and disparity estimation," in *ICRA*, 2019, pp. 2946–2952. 17

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

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

[224] 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

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

[226] 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

[227] 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

[228] 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," *arXiv preprint arXiv:2008.02268*, 2020. 17

[229] 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

[230] 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

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

[232] 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

[233] B. L. Bhatnagar, C. Sminchisescu, C. Theobalt, and G. Pons-Moll, "Combining implicit function learning and parametric models for 3d human reconstruction," *arXiv preprint arXiv:2007.11432*, 2020. 17

[234] T. He, J. Collomosse, H. Jin, and S. Soatto, "Geo-pifu: Geometry and pixel aligned implicit functions for single-view human reconstruction," *arXiv preprint arXiv:2006.08072*, 2020. 17

[235] 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

[236] 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

[237] Y. Ganin, D. Kononenko, D. Sungatullina, and V. Lempitsky, "Deepwarp: Photorealistic image resynthesis for gaze manipulation," in *ECCV*, 2016, pp. 311–326. 17

[238] E. Wood, T. Baltrusaitis, L. Morency, P. Robinson, and A. Bulling, "Gazedirector: Fully articulated eye gaze redirection in video," *Comput. Graph. Forum*, vol. 37, no. 2, pp. 217–225, 2018. 17

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

[240] H. Zhou, S. Hadap, K. Sunkavalli, and D. W. Jacobs, "Deep single-image portrait relighting," in *ICCV*, 2019, pp. 7194–7202. 17

[241] T. Sun, J. T. Barron, Y.-T. Tsai, Z. Xu, X. Yu, G. Fyffe, C. Rhemann, J. Busch, P. E. Debevec, and R. Ramamoorthi, "Single image portrait relighting." *ACM Trans. Graph.*, vol. 38, no. 4, pp. 79–1, 2019. 17

[242] X. C. Zhang, Y.-T. Tsai, R. Pandey, X. Zhang, R. Ng, D. E. Jacobs *et al.*, "Portrait shadow manipulation," *arXiv preprint arXiv:2005.08925*, 2020. 17

[243] X. Chen, J. Song, and O. Hilliges, "Monocular neural image based rendering with continuous view control," in *ICCV*, 2019, pp. 4090–4100. 17

[244] 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

[245] 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

[246] 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