

## Domain Expansion of Image Generators

Yotam Nitzan<sup>1,2</sup>    Michaël Gharbi<sup>1</sup>    Richard Zhang<sup>1</sup>    Taesung Park<sup>1</sup>    Jun-Yan Zhu<sup>3</sup>  
 Daniel Cohen-Or<sup>2</sup>    Eli Shechtman<sup>1</sup>

<sup>1</sup> Adobe Research    <sup>2</sup> Tel-Aviv University    <sup>3</sup> Carnegie Mellon University

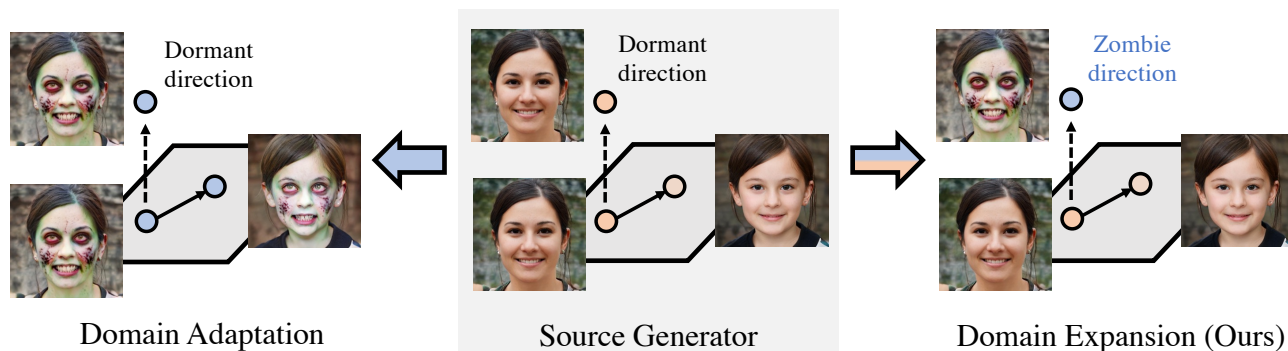


Figure 1. (center) Traversing the latent space of generative models along some directions changes the image significantly while traversing others has no perceptible effect. We call such directions dormant. (left) Domain adaptation methods, transform the entire generator from a source domain to a target domain. (right) We introduce an approach for a new task – domain expansion. Rather than transforming the generator fully, we expand it to include new data domains. Our method learns to represent the new domain in a disentangled manner by repurposing a single dormant direction.

### Abstract

Can one inject new concepts into an already trained generative model, while respecting its existing structure and knowledge? We propose a new task – domain expansion – to address this. Given a pretrained generator and novel (but related) domains, we expand the generator to jointly model all domains, old and new, harmoniously. First, we note the generator contains a meaningful, pretrained latent space. Is it possible to minimally perturb this hard-earned representation, while maximally representing the new domains? Interestingly, we find that the latent space offers unused, “dormant” directions, which do not affect the output. This provides an opportunity: By “repurposing” these directions, we can represent new domains without perturbing the original representation. In fact, we find that pretrained generators have the capacity to add several – even hundreds – of new domains! Using our expansion method, one “expanded” model can supersede numerous domain-specific models, without expanding the model size. Additionally, a single expanded generator natively supports smooth transitions between domains, as well as composition of domains.

Code and project page available [here](#).

### 1. Introduction

Recent *domain adaptation* techniques piggyback on the tremendous success of modern generative image models [3, 12, 32, 40], by adapting a pretrained generator so it can generate images from a new target domain. Oftentimes, the target domain is defined with respect to the source domain [5, 21, 22], e.g., changing the “stylization” from a photorealistic image to a sketch. When such a relationship holds, domain adaptation typically seeks to preserve the factors of variations learned in the source domain, and transfer them to the new one (e.g., making the human depicted in a sketch smile based on the prior from a face generator). With existing techniques, however, the adapted model loses the ability to generate images from the original domain.

In this work, we introduce a novel task — *domain expansion*. Unlike domain adaptation, we aim to *augment* the space of images a single model can generate, without overriding its original behavior (see Fig. 1). Rather than viewing similar image domains as disjoint data distributions, we treat them as different modes in a joint distribution. As a result, the domains share a semantic prior inherited from the original data domain. For example, the inherent factors of variation for photorealistic faces, such as pose and face shape, can equally apply to the domain of “zombies”.



Figure 2. Example of a *domain expansion* result. Starting from dogs as the source domain, we expand a *single* generator to model new domains such as facial expressions, breeds of dogs and other animals, and artistic styles. Finally, as the representations are disentangled, the expanded generator is able to generalize and compose the different domains, although they were never seen jointly in training.

To this end, we carefully structure the model training process for expansion, respecting the original data domain. It is well-known that modern generative models with low-dimensional latent spaces offer an intriguing, emergent property – through training, the latent spaces represent the factors of variation, in a linear and interpretable manner [3, 6, 10, 12, 28, 30, 39, 40]. We wish to extend this advantageous behavior and represent the new domains along linear and disentangled directions. Interestingly, it was previously shown that many latent directions have insignificant perceptible effect on generated images [6]. Taking advantage of this finding, we repurpose such directions to represent the new domains.

In practice, we start from an orthogonal decomposition of the latent space [36] and identify a set of low-magnitude directions that have no perceptible effect on the generated images, which we call *dormant*. To add a new domain, we select a dormant direction to repurpose. Its orthogonal subspace, which we call *base subspace*, is sufficient to represent the original domain [6]. We aim to repurpose the dormant direction such that traversing it would now cause a transition between the original and the new domain. Specifically, the transition should be disentangled from the original domain’s factors of variation. To this end, we define a *repurposed* affine subspace by transporting the base subspace along the chosen dormant direction, as shown in Fig. 3. We capture the new domain by applying a domain adaptation method, transformed to operate only on latent codes sampled from the repurposed subspace. A regularization loss is applied on the base subspace to ensure that the original domain is preserved. The original domain’s factors of variation are implicitly preserved due to the subspaces being parallel and the latent space being disentangled. For multiple new domains, we simply repeat this procedure across multiple dormant directions.

We apply our method to the StyleGAN [13] architecture,

with multiple datasets, and expand the generator with *hundreds* of new factors of variation. Crucially, we show our expanded model simultaneously generates high-quality images from both original and new domains, comparable to specialized, domain-specific generators. Thus, a single expanded generator supersedes hundreds of adapted generators, facilitating the deployment of generative models for real-world applications. We additionally demonstrate that the new domains are learned as global and disentangled factors of variation, alongside existing ones. This enables fine-grained control over the generative process and paves the way to new applications and capabilities, *e.g.*, compositing multiple domains (See Fig. 2). Finally, we conduct a detailed analysis of key aspects of our method, such as the effect of the number of newly introduced domains, thus shedding light on our method and, in the process, on the nature of the latent space of generative models.

To summarize, our contributions are as follows:

- We introduce a new task – domain expansion of a pre-trained generative model.
- We propose a novel latent space structure that is amenable to representing new knowledge in a disentangled manner, while maintaining existing knowledge intact.
- We present a simple paradigm transforming domain adaptation methods into domain expansion methods.
- We demonstrate successful domain expansion to hundreds of new domains and illustrate its advantage over domain adaptation methods.

## 2. Related Work

**Fine-tuning generative models.** Starting from a generator pretrained on a source domain and training it for a target domain, often called fine-tuning, is a common technique applied for various purposes and settings.

Some works wish to model only the target domain. In which case, the pretrained model is leveraged simply as an efficient initialization, shortening the training time, and improving image quality [11, 16, 19, 47, 50]. Others, wish to learn the target domain alongside the source domain, in a setting called continuous learning, and propose methods to ensure that the source domain is not forgotten [34, 44]. Although in a single generator, the domains are modeled separately, each as its own class.

A prominent line of works have sought to make the target domain inherit knowledge from the source domain [1, 2, 5, 14, 17, 21–23, 31, 33, 42, 43, 51]. This approach allows generalization beyond the target domain per-se and is especially useful when training data is scarce.

Our work similarly involves fine-tuning, but for a novel purpose. Our perspective is that, since the target domain is introduced with knowledge from the source domain – it is in essence, an expansion of it. Therefore, in contrast to the aforementioned works, we aim to model the domains jointly. The proposed method does not replace previous fine-tuning methods, but allows applying them jointly.

**Latent directions in generative models.** Generative models learn to represent the factors of variation of observed data in their latent space. Disentangled representations are especially useful as they facilitate intuitive control over the generative process. With recent architectures, disentanglement miraculously emerges without intervention [12, 24, 28, 39]. In such models, disentanglement is manifested through the existence of linear latent directions, each ideally controlling a single factor of variation.

Due to the spontaneous emergence of such directions, many works have been proposed to identify them after the model has been trained [6, 25, 35–37, 41, 45] and used them for downstream applications, most commonly semantic image editing. At the same time, it has also been observed that some latent directions have no perceptible effect on the generated images [6, 46]. These directions, which we call *dormant*, were not previously leveraged for any purpose.

In this work, we rely on existing methods to factorize the latent space into such linear directions. As we aim to expand the pretrained generator to additional domains, we decide to explicitly encode the “new knowledge” along the dormant directions, while keeping other directions intact. This design ensures that the original domain is preserved and that the different domains are represented in a disentangled fashion.

### 3. Method

We start with a pretrained generator  $G_{\text{src}}$  that maps from latent codes  $z \in \mathcal{Z} \subseteq \mathbb{R}^D$  to images in a source domain  $\mathcal{D}_{\text{src}}$ , and a set of  $N$  domain adaptation tasks, each defined by a loss function  $\mathcal{L}_i$ ,  $i \in \{1, \dots, N\}$ . In domain adaptation, fine-tuning  $G_{\text{src}}$  to minimize  $\mathcal{L}_i$  yields a generator  $G_i$  that generates images from the new domain  $\mathcal{D}_i$ . In contrast, our goal is domain expansion, which aims at training a single expanded generator  $G^+$  that can *simultaneously* model all the new domains  $\cup_{i=1}^N \mathcal{D}_i$ , along with the original domain  $\mathcal{D}_{\text{src}}$ . We want to ensure that the new domains  $\mathcal{D}_i$  are disentangled from each other and also share the factors of variation from the source domain, which remain intact.

Our solution is to partition the latent space into disjoint subspaces, one for each new domain, and to restrict the effect of each domain adaptation to the corresponding subspace. To this end, we endow the latent space with an explicit structure that supports domain expansion (Sec. 3.1), and optimize each domain adaptation loss only using latents from specific subspace reserved for the new domain (Sec. 3.2). Our decomposition reserves a base subspace for the original domain  $\mathcal{D}_{\text{src}}$ , on which we impose a regularization objective to maintain the behavior of the source generator (Sec. 3.3). Fig. 3 gives an overview of our domain expansion algorithm.

#### 3.1. Structuring the Latent Space for Expansion

Modern generative models conveniently learn to represent the factors of variation along linear latent directions,

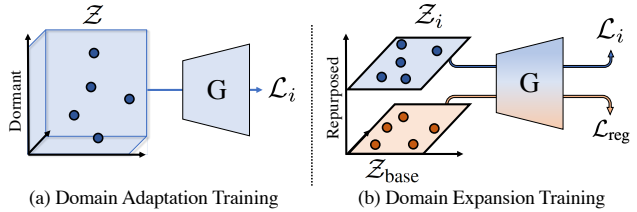


Figure 3. Our method transforms a domain adaptation task to a domain expansion task. (a) Generator  $G$  is optimized to satisfy the loss  $\mathcal{L}_i$  for every latent code in space. The entire generator and latent space now represent the new domain, indicated with the color blue. (b) Generator  $G$  is optimized to satisfy the same loss,  $\mathcal{L}_i$ , only on a subspace  $\mathcal{Z}_i$ , dedicated to the new domain. Simultaneously,  $G$  is optimized to satisfy a regularization term  $\mathcal{L}_{\text{reg}}$  on a parallel subspace,  $\mathcal{Z}_{\text{base}}$ , ensuring the original knowledge is preserved there. The generator and latent space now represent both domains, indicated by being colored both blue and orange. The latent direction between the two spaces was originally dormant in generator  $G$ , and now represents a transition between the domains.

in a completely unsupervised manner [12, 26, 28, 39]. We decide to explicitly extend this model by structuring the latent space such that the effect defined by an adaptation task would be represented along a single linear direction. Formally, there should exist some scalar  $s$  and latent direction  $v_i$ , for which images generated from  $G^+(z)$ ,  $G^+(z + sv_i)$ , relate to each other as the corresponding images from the source and adapted generators  $G_{\text{src}}(z)$ ,  $G_i(z)$  do.

Concretely, following SeFA [36], we obtain a semantic and orthogonal basis  $V$  of the latent space from the right singular vectors (produced by SVD) of the very first generator layer, which acts on the latent space  $\mathcal{Z}$ . With a similar factorization technique [6], it was observed that a relatively small subset of the basis vector is sufficient to represent most of the generators  $G_{\text{src}}$ ’s variability. Other basis vectors have barely any perceptible effect on the generated images. We find this to be the case with SeFA as well. We refer to vectors with no perceptible effect as *dormant*.

As the dormant directions do not affect the model’s generation capabilities, they are available to be repurposed with new desired behavior. We thus choose to represent the domains  $\mathcal{D}_{\text{src}}$  and  $\mathcal{D}_i$  in regions that are separated by only a dormant direction.

Formally, for each of the  $N$  adaptation tasks, we dedicate a single dormant direction,  $v_i$ , that will be repurposed. The remaining directions  $\{v_{N+1}, \dots, v_D\}$  will remain intact. We finally define a subspace of  $\mathcal{Z}$ , dubbed the *base subspace*, as

$$\mathcal{Z}_{\text{base}} = \text{span}(v_{N+1}, \dots, v_D) + \bar{z} \quad (1)$$

where  $\bar{z}$  is the mean of the distribution over the latents used to train the generator. Then, for each repurposed direction,  $v_i$ , we define a *repurposed subspace*  $\mathcal{Z}_i$  that is the base subspace transported along direction  $v_i$  by a predetermined scalar size  $s$ .

$$\mathcal{Z}_i = \mathcal{Z}_{\text{base}} + sv_i. \quad (2)$$

The choice of direction  $v_i$  and scalar  $s$  are discussed in Appendices C.2 and C.4.

Our domain expansion training procedure described hereafter will ensure subspace  $\mathcal{Z}_i$  is the only part of the latent space affected by the training objective  $\mathcal{L}_i$ , and is reserved to generate images from domain  $\mathcal{D}_i$ . Intuitively, shifting the base subspace along direction  $v_i$  aims to achieve two goals: 1. preserve the factors of variations inherited from  $\mathcal{Z}_{\text{base}}$ , and 2. restrict the new factor of variation (corresponding to  $\mathcal{D}_i$ ) to a single latent direction,  $v_i$ .

### 3.2. From Domain Adaptation to Expansion

Having defined disjoint affine subspaces  $\mathcal{Z}_i$  of the latent space  $\mathcal{Z}$  for our new domains  $\mathcal{D}_i$ , we now describe how we constrain each domain adaptation objective  $\mathcal{L}_i$  to affect only the corresponding subspace.

The domain adaptation objective is applied to images generated from latent codes  $z \in \mathcal{Z}$ , sampled from distribution  $p(z)$  defined on the entire space  $\mathcal{Z}$ . Commonly the distribution is a Gaussian, or is derived from it [12] but some exceptions exist [21, 43]. Our strategy is to transform this sample distribution into one restricted to the affine subspace  $\mathcal{Z}_i$ . We do so by projecting the samples from  $p(z)$  onto  $\mathcal{Z}_i$ , using a standard orthogonal projection operator

$$\text{proj}_{\mathcal{Z}_i}(z) = \sum_{j=N+1}^D (v_j^\top (z - \bar{z}))v_j + \bar{z} + sv_i. \quad (3)$$

Denoting by  $p_i$  the sampling distribution over  $\mathcal{Z}$  for each of the new domains we seek to adapt, the training loss over all tasks is defined as

$$\mathcal{L}_{\text{expand}} = \sum_{i=1}^N \mathbb{E}_{z \sim p_i(z)} \mathcal{L}_i(G(\text{proj}_{\mathcal{Z}_i}(z))). \quad (4)$$

### 3.3. Regularization

Optimizing  $\mathcal{L}_{\text{expand}}$  lets us learn to generate data from the new domains  $\mathcal{D}_i$  within a single generator, but unfortunately it leaves the base subspace  $\mathcal{Z}_{\text{base}}$  under-constrained and, therefore, does not guarantee it will remain unaltered during training. In practice, we observe that the effect of  $\mathcal{L}_i$  “leaks” outside  $\mathcal{Z}_i$ , causing *catastrophic forgetting* [18] in subspace  $\mathcal{Z}_{\text{base}}$ , and undesirably affecting other subspaces  $\mathcal{Z}_j$ . We show an example of this leakage in Fig. 4.

To prevent this failure mode, we explicitly enforce the preservation of  $G_{\text{src}}$ ’s behavior over the base subspace  $\mathcal{Z}_{\text{base}}$  by regularization. We adopt two successful regularization techniques. First, we keep optimizing the generator with the loss it was originally trained on,  $\mathcal{L}_{\text{src}}$ , which is known to mitigate forgetting [15]. Second, we apply *replay alignment* [44], which is a reconstruction loss that compares the output of a frozen copy of the source generator to that produced by our generator. We use a weighted combination of

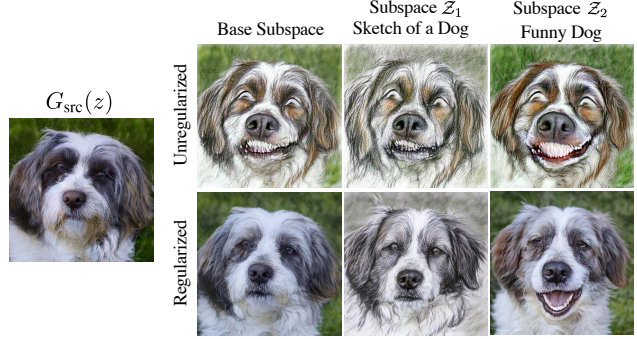


Figure 4. Regularization prevents leakage. Without regularization (top row), new factors of variation “Sketch” and “Funny” are leaking into the base subspace and the other repurposed subspace. Note, for example, that the image from the base subspace is both a sketch and depicts a smiling dog. Our regularization, described in Eq. (6), solves the issue (bottom row).

an  $L_2$  pixel loss and LPIPS [49]:

$$\mathcal{L}_{\text{recon}} = \lambda_{\text{lpiips}} \mathcal{L}_{\text{lpiips}}(G_{\text{src}}(z), G(z)) + \lambda_{L_2} \|G_{\text{src}}(z) - G(z)\|_2, \quad (5)$$

where  $\lambda_{\text{lpiips}} = \lambda_{L_2} = 10$  are weighting hyperparameters. Not only does replay alignment preserve the source domain  $\mathcal{D}_{\text{src}}$ , it also has the added benefit of aligning  $G^+$  to the source generator  $G_{\text{src}}$ , in the sense that they will produce similar outputs given the same latent code  $z$ .

Crucially, we only regularize the base subspace  $\mathcal{Z}_{\text{base}}$ , since the subspaces  $\mathcal{Z}_i$  should be allowed to change to learn the new behaviors. To this end, we project the latent codes to the base subspace  $\mathcal{Z}_{\text{base}}$ , before calculating the regularization terms. Our overall regularization objective is thus:

$$\mathcal{L}_{\text{reg}} = \mathbb{E}_{z \sim p_{\text{src}}(z)} [\lambda_{\text{src}} \mathcal{L}_{\text{src}}(G(\text{proj}_{\mathcal{Z}_{\text{base}}}(z))) + \mathcal{L}_{\text{recon}}(G(\text{proj}_{\mathcal{Z}_{\text{base}}}(z)))], \quad (6)$$

where  $\lambda_{\text{src}} = 1$  balances the two terms and  $p_{\text{src}}(z)$  is the latent distribution over  $\mathcal{Z}$  used to train  $G_{\text{src}}$ . Our final, regularized domain expansion objective is, therefore:

$$\mathcal{L}_{\text{full}} = \mathcal{L}_{\text{expand}} + \mathcal{L}_{\text{reg}}. \quad (7)$$

## 4. Experiments

We evaluate our method and analyze its key characteristics. Sec. 4.1 first details the experimental setting, focusing on StyleGAN. We start by analyzing the knowledge encoded along repurposed directions and compare it to domain adaptation methods (Sec. 4.2). We then delve deeper and evaluate the effects (Sec. 4.3) and opportunities (Sec. 4.4) presented by expanding a generator to multiple domains simultaneously. Next, we demonstrate that the quality of the source domain is maintained in the base subspace (Sec. 4.5). Finally, we demonstrate that our method generalizes to other generators (Sec. 4.6).

Further experiments, results, and details are provided in the supplementary.

### 4.1. Experimental Setting

We adopt StyleGAN2 [13] as the main generator architecture, for its disentangled latent space and because it has been the dominant test bed for generative domain adaptation methods in recent years [1, 5, 21, 22, 43, 51].

**Latent space and subspaces.** Several latent spaces have been considered in the context of StyleGAN. We use the intermediate latent space  $\mathcal{W}$  in all our experiments but note it as  $\mathcal{Z}$  for consistency. We use SeFA [36] for the orthogonal decomposition of  $\mathcal{Z}$ . As SeFA performs SVD, there is a native indication to how dormant is a given latent direction – the corresponding singular value. As singular values are commonly sorted in decreasing orders, the last basis vectors are most dormant. When expanding with  $N$  new domains, unless specified otherwise, we repurpose the last  $N$  basis vectors. We use  $s = 20$  in all experiments. These decisions are evaluated in greater depth in Appendices C.2 and C.4.

**Adaptation methods.** We demonstrate our expansion method with two domain adaptation tasks - StyleGAN-NADA [5] and MyStyle [21]. These two tasks were chosen as they differ significantly in key aspects – source of supervision, sampling distribution and loss.

StyleGAN-NADA is a zero-shot, text-guided, domain adaptation method. It takes as input a pair of text prompts,  $t_{source}$  and  $t_{target}$ , describing the desired transformation  $source \rightarrow target$  to be applied on the domain of the pre-trained generator,  $\mathcal{D}_{src}$ . The loss function  $\mathcal{L}$  is given by

$$\begin{aligned} \Delta T &= E_T(t_{target}) - E_T(t_{source}), \\ \Delta I &= E_I(G(z)) - E_I(G_{src}(z)), \\ \mathcal{L} &= 1 - \frac{\Delta I \cdot \Delta T}{|\Delta I| |\Delta T|}, \end{aligned} \tag{8}$$

where  $E_I$  and  $E_T$  are CLIP’s [29] image and text encoders respectively.

MyStyle is a few-shot, image-supervised, domain adaptation method. As input, it takes a set of images  $\{x_m\}_{m=1}^M$  of an individual ( $M \sim 100$ ), and adapts  $G_{src}$  to form a personalized prior for that individual. The generator is trained to better reconstruct  $x_m$  from their original latent space inversions  $z_m \in \mathcal{Z}$  [38]. Formally, the loss function is given by

$$\mathcal{L} = \sum_{m=1}^M [\mathcal{L}_{lips}(G(z_m), x_m) + \|G(z_m) - x_m\|_2], \tag{9}$$

where  $\mathcal{L}_{lips}$  is again the LPIPS loss [49].

**Datasets and models.** We demonstrate our method on four datasets – FFHQ [12], AFHQ Dog [4], LSUN Church [48] and SD-Elephant [20]. The FFHQ model is expanded with 105 new domains, 100 introduced with the expanded

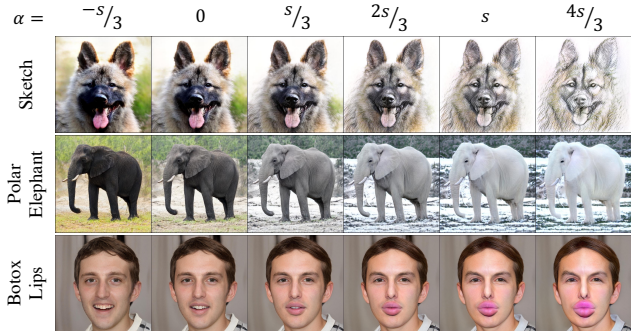


Figure 5. Continuous traversal along repurposed directions. As can be seen, the traversal between the base subspace ( $\alpha = 0$ ) and repurposed subspace ( $\alpha = s$ ) portrays a smooth transition between the source and newly introduced domains. Advantageously, the semantic meaning of the repurposed direction is preserved in the extrapolation, representing the opposite relationship between the domains ( $\alpha < 0$ ) or exaggerations of it ( $\alpha > s$ ).

Method	User % ( $\uparrow$ )	ID ( $\uparrow$ )	Diversity $\times 10$ ( $\uparrow$ )
StyleGAN-NADA	41.2%	-	$2.42 \pm 0.13$
Ours w/ NADA	<b>58.8%</b>	-	$2.42 \pm 0.13$
MyStyle	-	<b><math>0.80 \pm 0.06</math></b>	$3.08 \pm 0.15$
Ours w/ MyStyle	-	$0.76 \pm 0.05$	<b><math>3.14 \pm 0.14</math></b>

Table 1. Quantitative comparison of images generated from our repurposed subspaces to those generated by corresponding domain adaptation methods - StyleGAN-NADA [5] and MyStyle [21]. We follow each adaptation method’s quantitative evaluation protocol.

variant of StyleGAN-NADA and 5 from the expanded variant of MyStyle. The AFHQ Dog, LSUN Church and SD-Elephant are expanded with 50, 20, and 20 new domains correspondingly, all introduced from the expanded variant of StyleGAN-NADA.

### 4.2. Evaluating Domains Individually

**Traversing a repurposed direction.** We start by investigating what knowledge, if any, is encoded along the repurposed latent directions. To this end, starting from a random latent code  $z \in \mathcal{Z}_{base}$ , we individually traverse different repurposed directions,  $v_i$ , and inspect the generated images  $G^+(z + \alpha v_i)$ . Sample results from our dogs, elephants, and faces models are displayed in Fig. 5. We find that each individual repurposed direction now successfully encodes the desired factor of variation, in a global and continuous way.

Our training paradigm is inherently discrete – encouraging the source behavior on the base subspace ( $\alpha = 0$ ) and the newly introduced effect on the repurposed space ( $\alpha = s$ ). Therefore, obtaining a smooth effect might seem surprising at first glance. Nevertheless, this phenomenon can be clearly traced to the well-established observation that generators are smooth with respect to their latent space [12].

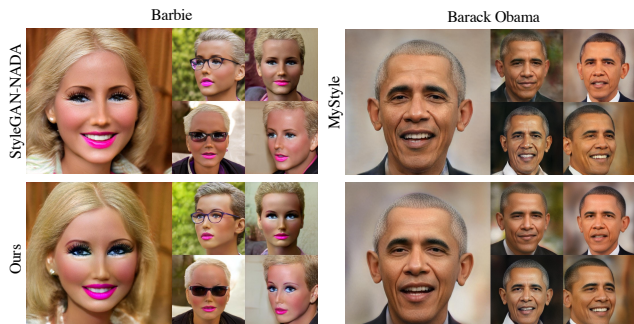


Figure 6. A random set of images generated by our generator from repurposed subspaces (bottom) and by corresponding domain adaptation methods (top). The images are similar and differences are subtle.

**Behavior on the repurposed subspace.** We have transformed adaptation tasks into expansion tasks by limiting the training effect to the repurposed subspaces only. But, for latents in repurposed subspaces ( $\alpha = s$ ), the domain adaptation could be considered to have been applied as-is.

We next directly compare the images generated by our generator from the repurposed subspace to the corresponding images generated by the domain-adapted generator. We inherit and repeat the quantitative evaluation protocols performed by each of the adaptation tasks. To compare quality with StyleGAN-NADA [5] we perform a two-alternative forced choice user study. Users were asked to pick the image that has higher-quality and better aligns with the target text used for training. We gathered 1440 responses from 32 unique users. To compare quality with MyStyle [21], we evaluate preservation of identity in generated images, as observed by a face recognition network [9]. For both methods, the diversity is compared based on intra-cluster LPIPS [49] distance, first suggested by Ojha et al. [22]. We use 10 domains for comparison with StyleGAN-NADA and 5 for comparison with MyStyle. Note that we use a single generator  $G^+$ , expanded with 105 domains, while competing methods use a dedicated model per domain, 15, overall. We report the results in Tab. 1 and Fig. 6.

As can be seen, on the repurposed subspaces, our method produces comparable images to that generated by the dedicated, domain-adapted generator. Perhaps surprisingly, users somewhat prefer our results over StyleGAN-NADA’s. We speculate this is due to the significantly greater difficulty of choosing hyperparameters for their training.

### 4.3. Effect of Domains on Each Other

Previous evaluation of individual repurposed directions already indicates disentanglement between different factors of variation. For example, “Barbie” images in Fig. 6 show no sign of being caricature, Barack Obama, or any of the other hundred factors of variation introduced to that generator. In this section, we delve deeper into evaluating the effects of expanding with multiple factors of variation.

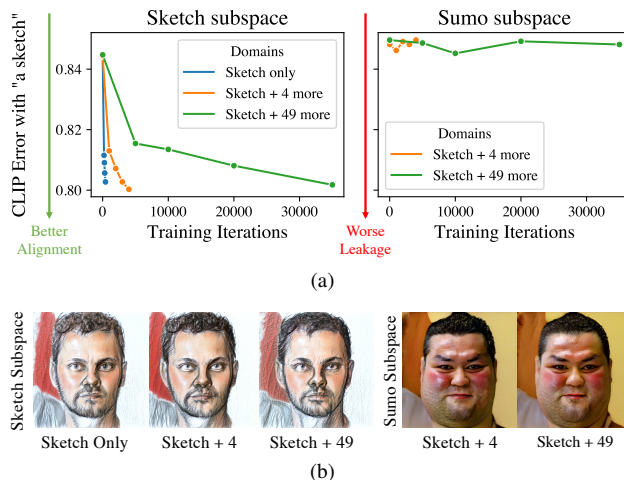


Figure 7. Investigating the effect of introducing multiple domains simultaneously. (a) Reports the CLIP error of generated images with the text “a sketch”, as a function of training iterations. Images are generated from the “Sketch” and “Sumo” subspaces of models trained with a different number of domains. (b) Depicts generated images from models that have similar CLIP errors. As can be seen, the sketch domain does not “leak” into the sumo subspace. Additionally, introducing additional domains delays, but does not prevent, the introduction of sketch.

To this end, we train three models to expand the FFHQ parent model with either 1, 5 or 50 new domains, all induced by StyleGAN-NADA [5]. All models are expanded with “Sketch”, and the latter two with “Sumo” as well as other factors of variation. We quantify the strength of introduced factor of variation using *CLIP error*, the 1-complement of the score produced by CLIP [29]. We use the top-performing version of the CLIP encoder available, ViT-L/14, which is not used during training. We note that simply minimizing CLIP error is not the objective, as it might lead to favoring mode-collapsed and adversarial examples [5]. Nevertheless, together with qualitative inspection, it is useful for comparing different models.

In Fig. 7a, we report the CLIP error of generated images with the text “a sketch”, as a function of the training iterations. Images are generated from the “Sketch”, and if exists, “Sumo” subspace. First, we observe that CLIP error is decreasing for “Sketch” subspaces in all models, as expected. Conversely, CLIP error in the “Sumo” subspace does not significantly change, indicating it is not becoming any more or less of a sketch. This result quantitatively supports our previous finding, that factors of variations do not interfere with each other, and demonstrates it is true regardless of the number of other factors of variations learned simultaneously. Additionally, we observe that expanding with additional factors of variation delays, but does not prevent,  $G^+$  the introduction of “Sketch” effectively. The observed delay is expected, as expanding with more variations corresponds to  $G^+$  optimizing and balancing additional loss functions.

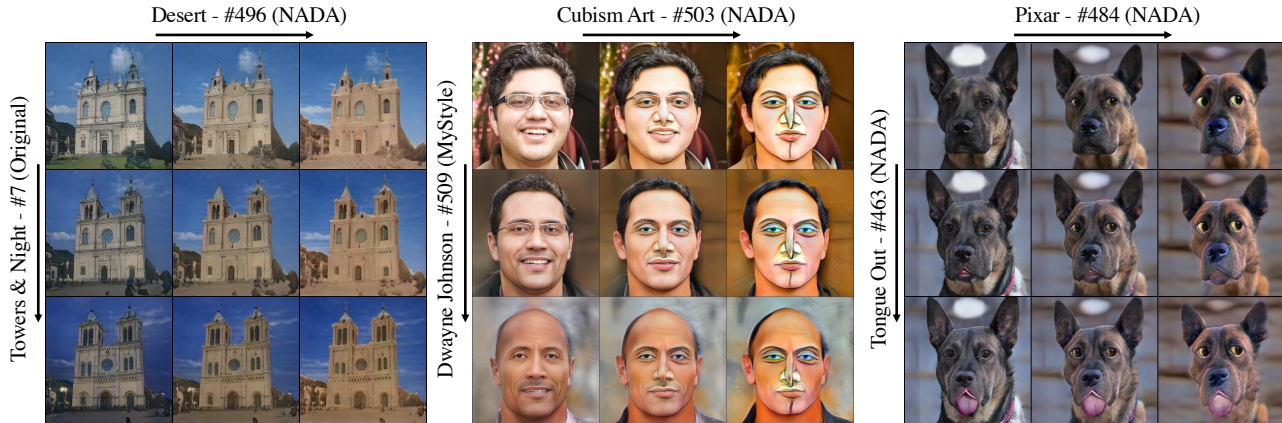


Figure 8. Composing multiple effects by simple latent traversal. In each grid, we start from the latent code that generates the top-left image and traverse along two latent directions, represented by advancement in rows and columns. For each direction, we note the associated domain, its ordinal number in the latent space’s basis, and the training method used (“NADA” or “MyStyle”) to learn the domain. As can be seen,  $G^+$  has learned a disentangled representation, allowing meaningful composition of concepts. Specifically, note the disentanglement between directions, as traversing left-right does not affect the magnitude of the effect corresponding to up-down traversal, and vice versa.

Generated samples from the sketch and sumo subspaces are provided in Fig. 7b.

#### 4.4. Compositionality

While accidental “leakage” between latent directions during training is undesired, intentionally composing variations at test time is useful. For generative models with a disentangled latent space, summing together latent directions aggregates their semantic meaning, and should not affect the magnitude of their effects if applied separately. For example, if direction  $v_1$  controls head pose and direction  $v_2$  controls an unrelated variation, images  $G(z + v_1 + v_2)$  and  $G(z + v_1)$  should depict the same head pose.

We find that the latent space of the expanded generator  $G^+$  is disentangled, and variations can indeed be composed effectively. Crucially, variations can be composed with each other regardless of their originating training task, including those on the base subspace, learned from the source domain. Fig. 8 shows a sample of gradual composition results across models and training tasks.

**Comparison to existing techniques.** Several domain adaptation methods have proposed techniques to combine multiple variations. These methods still train a *separate* generative model per variation, but combine their effects in test-time. Specifically, in the realm of CLIP-supervised training, StyleGAN-NADA [5] interpolates the generators’ weights, while DiffusionCLIP [14] interpolates intermediate activations of the generators. Next, we compare the disentanglement of composition in our generator to that made possible using these techniques.

For each method, we start with a setting that was optimized to generate images that align with one of two text prompts. In our case, this setting is  $G^+$  with latent codes in certain subspaces. For the baselines, these settings are

Model	FFHQ	AFHQ	LSUN Church	SD Elephant
Parent	2.77	7.43	3.92	2.30
Ours	2.80	7.51	3.76	2.70
Only $\mathcal{L}_{src}$	$2.75 \pm 0.08$	$7.38 \pm 0.09$	$3.31 \pm 0.22$	$3.91 \pm 0.67$

Table 2. We generate images from the base subspace and report FID [7] ( $\downarrow$ ) with respect to source domain dataset. We compare our FID to that of the source generator  $G_{src}$ . For reference, we also continue training the source generator for the same number of iterations with its original loss -  $\mathcal{L}_{src}$ , and report the mean and standard deviations of FID along the training. As can be seen, on the base subspace, our models have comparable FID scores to their parents. Furthermore, similar magnitude of change in FID are observed by simply continuing training, indicating that the change in FID might be, at least in part, due to “random” fluctuations.

dedicated generators with any latent code. Then, for each method, we gradually introduce the variation described by the other text prompt and generate the corresponding images. Finally, we measure normalized CLIP error between generated images and the two prompts. We normalize all errors by the error of the initial setting, to make the metric comparable across methods and text prompts. Fig. 9 reports the mean and standard deviation of the CLIP error, on 10 pairs of prompts, and provides a sample of qualitative results. As can be seen, both baseline methods directly trade-off one domain for the other, expressed by a linear-looking trend. Conversely, our method obtains significantly lower errors and allows for a true composition of concepts.

#### 4.5. Preservation of the Source Domain

We next evaluate the preservation of the source domain in  $G^+$ . To this end, using FID [7], we compare the quality

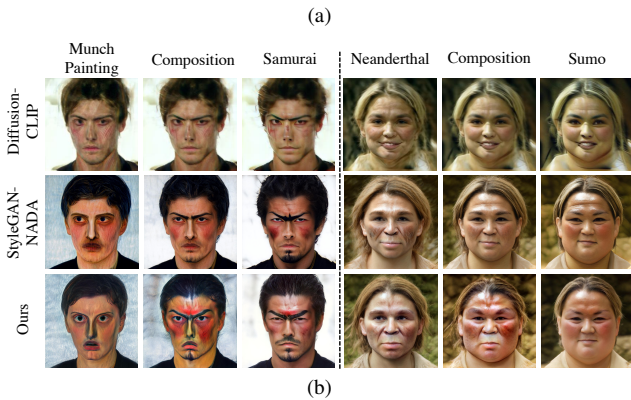
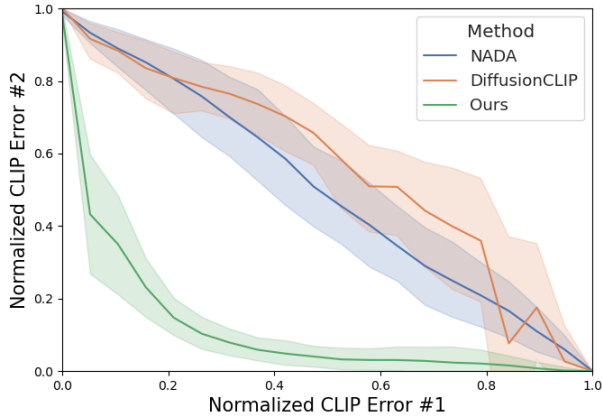


Figure 9. Comparing compositionality in our generator to methods of combining multiple domains proposed by StyleGAN-NADA [5] and DiffusionCLIP [14]. Starting from a setting optimized for either text prompt #1 or #2, we gradually introduce the variation described by the other text. (a) Reports the CLIP error to both prompts along the gradual introduction, normalized to the error obtained for each text prompt in isolation. (b) Portrays a sample of qualitative results, where the composition is such that assigns equal strengths to both effects. As can be seen in, both quantitatively and qualitatively, NADA and DiffusionCLIP directly trade-off one effect for the other – strengthening the effect of one prompt directly lessens that of the other. In contrast, our generator allows true composition of modalities.

of images generated from the base subspace  $Z_{\text{base}}$  of  $G^+$  to those generated by the source generator  $G_{\text{src}}$ . Since the generator is being trained, some change in FID is expected. Therefore, we also report the average and standard deviation over FID scores for a generator that simply continues training, *i.e.*, using only the original loss  $\mathcal{L}_{\text{src}}$ . Results are reported in Tab. 2 and vary between datasets. Across all datasets, we observe that the FID from our base subspace is within  $1\sigma$  of the that obtained from either the parent or a generator that continues training. For reference, generator’s adapted with StyleGAN-NADA and MyStyle have an average FID of 125 and 183, respectively. We conclude that the expansion method might have a slight impact on FID, but it is negligible.

#### 4.6. Generalization Beyond StyleGAN

We finally demonstrate that our expansion method generalizes to other models, beyond StyleGAN. To this end, we experiment with Diffusion Autoencoder (DiffAE) [28]. Despite differences in architecture and training method, DiffAE also has a semantic latent space, marked with  $z_{\text{sem}}$ . We apply our approach over DiffAE and its latent space, using StyleGAN-NADA [5] as the base adaptation method. We find that our expansion methods works as well with DiffAE, and provide a sample of qualitative results in Fig. 10.

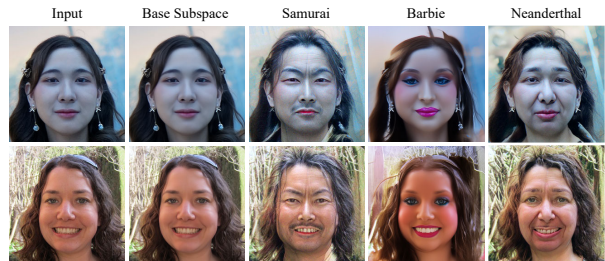


Figure 10. Applying our domain expansion method with Diffusion Autoencoder [28].

#### 5. Conclusions

We present a new problem – *domain expansion* – and propose an approach to solve it. The core of our method is to carefully structure the latent space, such that it is amenable to learning additional knowledge, while keeping the existing knowledge intact. Our method takes advantage of the existence of dormant latent directions, and the task itself implicitly relies on the capacity of the model weights to represent more knowledge. If one of these assumptions does not hold, it might not be possible to apply domain expansion. However, the popularity of methods squeezing neural networks, such as Knowledge Distillation [8], and current estimates of the intrinsic dimensionality of image datasets [27], indicate that these assumptions commonly hold. In our experiments, we were able to expand to hundreds of directions. A plausible limitation is that the model can be expanded to a certain point but ultimately limited by factors such as the latent space or network capacity. Overcoming this limitation, perhaps by considering more complex latent space structures, is an avenue for future work.

**Acknowledgment.** This work was done while Yotam Nitzan was an intern at Adobe. This research was supported in part by the Israel Science Foundation (grants no. 2492/20 and 3441/21), Len Blavatnik and the Blavatnik family foundation, and The Tel Aviv University Innovation Laboratories (TILabs).



## References

- [1] David Bau, Steven Liu, Tongzhou Wang, Jun-Yan Zhu, and Antonio Torralba. Rewriting a deep generative model. In *European conference on computer vision*, pages 351–369. Springer, 2020. [2](#), [5](#)
- [2] David Bau, Hendrik Strobelt, William Peebles, Jonas Wulff, Bolei Zhou, Jun-Yan Zhu, and Antonio Torralba. Semantic photo manipulation with a generative image prior. *arXiv preprint arXiv:2005.07727*, 2020. [2](#)
- [3] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018. [1](#), [2](#)
- [4] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. StarGAN v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8188–8197, 2020. [5](#)
- [5] Rinon Gal, Or Patashnik, Haggai Maron, Gal Chechik, and Daniel Cohen-Or. Stylegan-nada: Clip-guided domain adaptation of image generators. *arXiv preprint arXiv:2108.00946*, 2021. [1](#), [2](#), [5](#), [6](#), [7](#), [8](#)
- [6] Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. GANSpace: Discovering interpretable GAN controls. *arXiv preprint arXiv:2004.02546*, 2020. [2](#), [3](#)
- [7] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017. [7](#)
- [8] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015. [8](#)
- [9] Yuge Huang, Yuhan Wang, Ying Tai, Xiaoming Liu, Pengcheng Shen, Shaoxin Li, Jilin Li, and Feiyue Huang. CurricularFace: Adaptive curriculum learning loss for deep face recognition, 2020. [6](#)
- [10] Ali Jahanian, Lucy Chai, and Phillip Isola. On the “steerability” of generative adversarial networks. *arXiv preprint arXiv:1907.07171*, 2019. [2](#)
- [11] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. In *Proc. NeurIPS*, 2020. [2](#)
- [12] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proc. CVPR*, pages 4401–4410, 2019. [1](#), [2](#), [3](#), [4](#), [5](#)
- [13] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In *Proc. CVPR*, pages 8110–8119, 2020. [2](#), [5](#)
- [14] Gwanghyun Kim, Taesung Kwon, and Jong Chul Ye. Diffusionclip: Text-guided diffusion models for robust image manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2426–2435, 2022. [2](#), [7](#), [8](#)
- [15] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017. [4](#)
- [16] Nupur Kumari, Richard Zhang, Eli Shechtman, and Jun-Yan Zhu. Ensembling off-the-shelf models for gan training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10651–10662, 2022. [2](#)
- [17] Yijun Li, Richard Zhang, Jingwan Lu, and Eli Shechtman. Few-shot image generation with elastic weight consolidation. *arXiv preprint arXiv:2012.02780*, 2020. [2](#)
- [18] Michael McCloskey and Neal J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In Gordon H. Bower, editor, *Psychology of Learning and Motivation*, volume 24, pages 109–165. Academic Press, 1989. [4](#)
- [19] Sangwoo Mo, Minsu Cho, and Jinwoo Shin. Freeze the discriminator: a simple baseline for fine-tuning GANs. *arXiv preprint arXiv:2002.10964*, 2020. [2](#)
- [20] Ron Mokady, Michal Yarom, Omer Tov, Oran Lang, Michal Irani Daniel Cohen-Or, Tali Dekel, and Inbar Mosseri. Self-distilled stylegan: Towards generation from internet photos, 2022. [5](#)
- [21] Yotam Nitzan, Kfir Aberman, Qiurui He, Orly Liba, Michal Yarom, Yossi Gandelsman, Inbar Mosseri, Yael Pritch, and Daniel Cohen-Or. Mystyle: A personalized generative prior. *arXiv preprint arXiv:2203.17272*, 2022. [1](#), [2](#), [4](#), [5](#), [6](#)
- [22] Utkarsh Ojha, Yijun Li, Jingwan Lu, Alexei A Efros, Yong Jae Lee, Eli Shechtman, and Richard Zhang. Few-shot image generation via cross-domain correspondence. *arXiv preprint arXiv:2104.06820*, 2021. [1](#), [2](#), [5](#), [6](#)
- [23] Xingang Pan, Xiaohang Zhan, Bo Dai, Dahua Lin, Chen Change Loy, and Ping Luo. Exploiting deep generative prior for versatile image restoration and manipulation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. [2](#)
- [24] Taesung Park, Jun-Yan Zhu, Oliver Wang, Jingwan Lu, Eli Shechtman, Alexei Efros, and Richard Zhang. Swapping autoencoder for deep image manipulation. *Advances in Neural Information Processing Systems*, 33:7198–7211, 2020. [3](#)
- [25] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. StyleCLIP: Text-driven manipulation of StyleGAN imagery. *arXiv preprint arXiv:2103.17249*, 2021. [3](#)
- [26] William Peebles, John Peebles, Jun-Yan Zhu, Alexei Efros, and Antonio Torralba. The hessian penalty: A weak prior for unsupervised disentanglement. In *ECCV*. Springer, 2020. [3](#)
- [27] Phillip Pope, Chen Zhu, Ahmed Abdelkader, Micah Goldblum, and Tom Goldstein. The intrinsic dimension of images and its impact on learning. *arXiv preprint arXiv:2104.08894*, 2021. [8](#)
- [28] Konpat Preechakul, Nattanat Chatthee, Suttisak Wizadwongsa, and Supasorn Suwajanakorn. Diffusion autoencoders: Toward a meaningful and decodable representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10619–10629, 2022. [2](#), [3](#), [8](#)
- [29] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language super-

- vision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 5, 6
- [30] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 2
- [31] Daniel Roich, Ron Mokady, Amit H Bermano, and Daniel Cohen-Or. Pivotal tuning for latent-based editing of real images. *arXiv preprint arXiv:2106.05744*, 2021. 2
- [32] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022. 1
- [33] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. *arXiv preprint arXiv:2208.12242*, 2022. 2
- [34] Ari Seff, Alex Beaton, Daniel Suo, and Han Liu. Continual learning in generative adversarial nets. *arXiv preprint arXiv:1705.08395*, 2017. 2
- [35] Yujun Shen, Ceyuan Yang, Xiaoou Tang, and Bolei Zhou. InterFaceGAN: interpreting the disentangled face representation learned by GANs. *arXiv preprint arXiv:2005.09635*, 2020. 3
- [36] Yujun Shen and Bolei Zhou. Closed-form factorization of latent semantics in GANs. *arXiv preprint arXiv:2007.06600*, 2020. 2, 3, 5
- [37] Nurit Spingarn-Eliezer, Ron Banner, and Tomer Michaeli. Gan” steerability” without optimization. *arXiv preprint arXiv:2012.05328*, 2020. 3
- [38] Omer Tov, Yuval Alaluf, Yotam Nitzan, Or Patashnik, and Daniel Cohen-Or. Designing an encoder for StyleGAN image manipulation. *arXiv preprint arXiv:2102.02766*, 2021. 5
- [39] Arash Vahdat and Jan Kautz. Nvae: A deep hierarchical variational autoencoder. *Advances in Neural Information Processing Systems*, 33:19667–19679, 2020. 2, 3
- [40] Arash Vahdat, Karsten Kreis, and Jan Kautz. Score-based generative modeling in latent space. *Advances in Neural Information Processing Systems*, 34:11287–11302, 2021. 1, 2
- [41] Andrey Voynov and Artem Babenko. Unsupervised discovery of interpretable directions in the GAN latent space. *arXiv preprint arXiv:2002.03754*, 2020. 3
- [42] Sheng-Yu Wang, David Bau, and Jun-Yan Zhu. Sketch your own gan. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14050–14060, 2021. 2
- [43] Sheng-Yu Wang, David Bau, and Jun-Yan Zhu. Rewriting geometric rules of a gan. *ACM Transactions on Graphics (TOG)*, 41(4):1–16, 2022. 2, 4, 5
- [44] Chenshen Wu, Luis Herranz, Xialei Liu, Joost van de Weijer, Bogdan Raducanu, et al. Memory replay gans: Learning to generate new categories without forgetting. *Advances in Neural Information Processing Systems*, 31, 2018. 2, 4
- [45] Zongze Wu, Dani Lischinski, and Eli Shechtman. StyleSpace analysis: Disentangled controls for StyleGAN image generation. *arXiv:2011.12799*, 2020. 3
- [46] Zongze Wu, Yotam Nitzan, Eli Shechtman, and Dani Lischinski. Stylealign: Analysis and applications of aligned stylegan models. *arXiv preprint arXiv:2110.11323*, 2021. 3
- [47] Ceyuan Yang, Yujun Shen, Yinghao Xu, and Bolei Zhou. Data-efficient instance generation from instance discrimination. *arXiv preprint arXiv:2106.04566*, 2021. 2
- [48] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015. 5
- [49] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 4, 5, 6
- [50] Shengyu Zhao, Zhijian Liu, Ji Lin, Jun-Yan Zhu, and Song Han. Differentiable augmentation for data-efficient gan training. *arXiv preprint arXiv:2006.10738*, 2020. 2
- [51] Peihao Zhu, Rameen Abdal, John Femiani, and Peter Wonka. Mind the gap: Domain gap control for single shot domain adaptation for generative adversarial networks. *arXiv preprint arXiv:2110.08398*, 2021. 2, 5