EigenGAN: Layer-Wise Eigen-Learning for GANs

Zhenliang He[1,2], Meina Kan[1,2], Shiguang Shan[1,2,3]
[1] Key Laboratory of Intelligent Information Processing, ICT, CAS
[2] University of Chinese Academy of Sciences, Beijing, China
[3] Peng Cheng Laboratory, Shenzhen, China
zhenliang.he@vipl.ict.ac.cn, {kanmeina,sgshan}@ict.ac.cn

Abstract

Recent studies on Generative Adversarial Network (GAN) reveal that different layers of a generative CNN hold different semantics of the synthesized images. However, few GAN models have explicit dimensions to control the semantic attributes represented in a specific layer. This paper proposes EigenGAN which is able to unsupervisedly mine interpretable and controllable dimensions from different generator layers. Specifically, EigenGAN embeds one linear subspace with orthogonal basis into each generator layer. Via generative adversarial training to learn a target distribution, these layer-wise subspaces automatically discover a set of "eigen-dimensions" at each layer corresponding to a set of semantic attributes or interpretable variations. By traversing the coefficient of a specific eigen-dimension, the generator can produce samples with continuous changes corresponding to a specific semantic attribute. Taking the human face for example, EigenGAN can discover controllable dimensions for high-level concepts such as pose and gender in the subspace of deep layers, as well as low-level concepts such as hue and color in the subspace of shallow layers. Moreover, in the linear case, we theoretically prove that our algorithm derives the principal components as PCA does. Codes can be found in https://github.com/LynnHo/EigenGAN-Tensorflow.

1. Introduction

Strong evidences [40, 42, 2] show that different layers of a discriminative CNN capture different semantic concepts in terms of abstraction level, e.g., shallower layers detect color and texture while deeper layers focus more on objects and parts. Accordingly, we can expect that a generative CNN also has similar property, which is confirmed by the recent studies of generative adversarial network (GAN) [18, 39, 3]. StyleGAN [18] shows that deeper generator layers control higher-level attributes such as pose and glasses while shallower layers control lower-level features such as color and edge. Yang *et al*. [39] found similar phenomenon in scene synthesis, showing that deep layers tend to determine the spatial layout while shallow layers determine the color scheme. Similar conclusion is also made by Bau *et al*. [3]. All these evidences reveal a property that *different generator layers hold different semantics of the synthesized images in terms of abstraction level*.

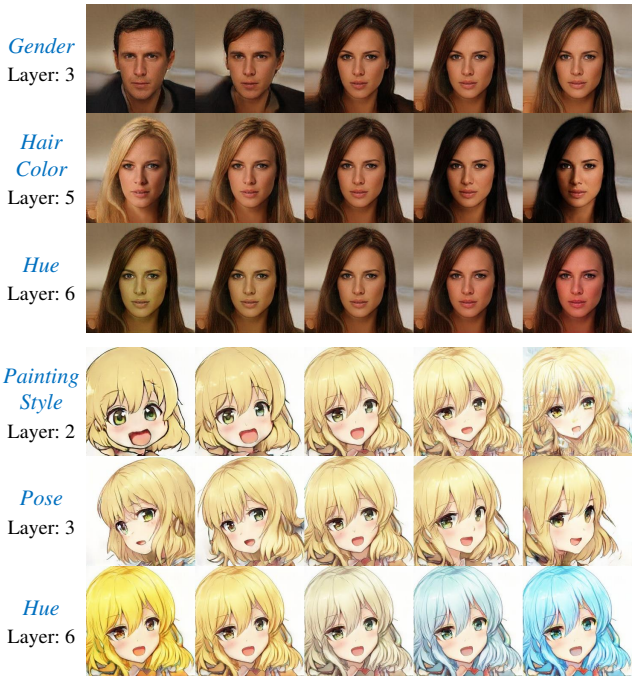


Figure 1. Example of interpretable dimensions learned by EigenGAN. The smaller the index, the deeper the layer.

According to this property, one can identify semantic attributes from different layers of a well-trained generator by performing *post-processing* algorithms [3, 12, 36, 39], and then can manipulate these attributes on the synthesized images. For example, Bau *et al*. [3] identify the causal units for a specific concept (such as "tree") by dissection and intervention on each generator layer. Turning on or off the causal units causes the concept to appear or disappear on the synthesized image. However, these post-processing methods can only be applied to a well-trained and fixed generator. As for the generator itself, it still operates as a black box and lacks explicit dimensions to directly control the semantic attributes represented in different layers. In other words, we do not know what attributes are represented in different generator layers or how to manipulate them, unless we deeply inspect each layer by these post-processing methods.

Under above discussion, this paper starts with a question: *can a generator itself automatically/unsupervisedly learn explicit dimensions that control the semantic attributes represented in different layers?* To this end, we propose to embed one linear subspace model with orthogonal basis into each generator layer, named as EigenGAN. First, via generative adversarial training, the generator tries to capture the principal variations of the data distribution, and these principal variations are separately represented in different layers in terms of their abstraction level. Second, with the help of the subspace model, the principal variations of a specific layer are further orthogonally separated into different basis vectors. Finally, each basis vector discovers an "eigen-dimension" that controls an attribute or interpretable variation corresponding to the semantics of its layer. For example, as shown at the top of Fig. 1, an eigen-dimension of the subspace embedded in a deep layer controls gender, while another of the subspace embedded in the shallowest layer controls the hue of the image. Furthermore, in the linear case, i.e., one layer model, we theoretically prove that our EigenGAN is able to discover the principal components as PCA [15] does, which gives us a strong insight and reason to embed the subspace models into different generator layers. Besides, we also provide a manifold perspective showing that our EigenGAN decomposes the data generation modeling into layer-wise dimension expanding steps.

2. Related Works

2.1. Interpretability Learning for GANs

The first attempt to learn interpretable representations for GAN generators is InfoGAN [6] which employs mutual information maximization (MIM) between the latent variable and synthesized samples. Including InfoGAN, MIM based methods [6, 16, 17, 14, 20, 21, 22] can automatically discover interpretable dimensions which respectively control different semantic attributes such as pose, glasses and emotion of human face. However, the learning of these interpretable dimensions is mainly driven by the MIM objective, and there is no direct link from these dimensions to the semantics of any specific generator layer. Ramesh et al. [33] found that the principal right-singular subspace of the generator Jacobian shows local disentanglement property, then they apply a spectral regularization to align the singular vectors with straight coordinates, and finally obtain globally interpretable representations. However, this work also does not investigate the correspondence between these interpretable representations and the semantics of different generator layers. Different from these methods, the interpretability of our EigenGAN comes from the special design of layer-wise subspace embedding, rather than imposing any objective or regularization. Moreover, our EigenGAN establishes an explicit connection between the interpretable dimensions and the semantics of a specific layer by directly embedding a subspace model into that layer.

The above methods try to learn a GAN generator with explicit interpretable representations; in contrast, another class of methods, post-processing methods, try to reveal the interpretable factors from a well-trained GAN generator [9, 3, 35, 39, 32, 12, 38, 36]. [9, 3, 35, 39] adopt pre-trained semantic predictors to identify the corresponding semantic factors in the GAN latent space, e.g., Yang *et al*. [39] use layout estimator, scene category recognizer, and attribute classifier to find out the decision boundaries for these concepts in the latent space. Without introducing external supervision, several methods search interpretable factors in self-supervised [32] or unsupervised [12, 36] manners. Plumerault *et al*. [32] utilize simple image transforms (e.g., translation and zoom) to search the axes for these transforms in the latent space. Harkonen *et al*. [12] apply PCA to the feature space of the early layers, and the resulting principal components represent interpretable variations. Shen and Zhou [36] show that the weight matrix of the very first fully-connected layer of a generator determines a set of critical latent directions which dominate the image synthesis, and the moving along these directions controls a set of semantic attributes. Among these methods, [3, 35, 39, 12, 36] carefully investigate the semantics represented in different generator layers. However, these post-processing methods must first learn and fix a GAN generator then learn interpretable dimensions under separated objectives (two steps). On the contrary, our EigenGAN learns the interpretable dimensions for each generator layer along with the GAN training in an end-to-end manner (one step). Therefore, our method should have a better optimum because the learning of generator and the learning of interpretable dimensions can interact with each other.

2.2. Generative Adversarial Networks

Generative adversarial network (GAN) [10] is a sort of generative model which can synthesize data from noise. The learning process of GAN is the competition between a generator and a discriminator. Specifically, the discriminator tries to distinguish the synthesized samples from the real ones, while the generator tries to make the synthesized samples as realistic as possible to fool the discriminator. When the competition reaches Nash equilibrium, the synthesized data distribution is identical to the real data distribution.

GANs show promising performance and properties on data synthesis. Therefore, plenty of researches on GANs appear, including loss functions [30, 25, 1], regularizations [34, 26, 28], conditional generation [27, 31, 29], representation learning [24, 6, 8], architecture design [7, 5, 18], applications [13, 43, 41], and etc. Our EigenGAN can be categorized into representation learning as well as architecture design for GANs.

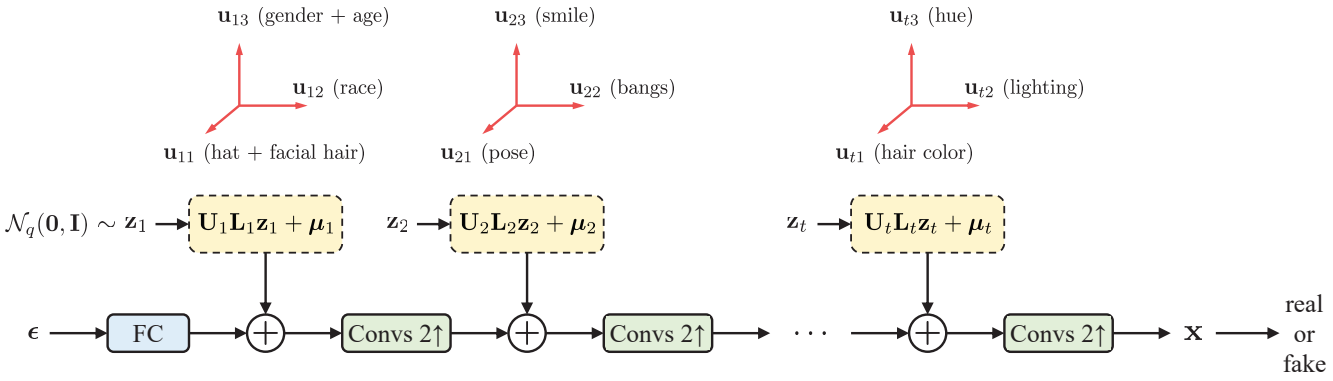


Figure 2. Overview of the proposed EigenGAN. The main stream of the model is a chain of 2-stride transposed convolutional blocks which gradually enlarges the resolution of the feature maps and finally outputs a synthesized sample. In the i^{th} layer, we embed a linear subspace with orthonormal basis $\mathbf{U}_i = [\mathbf{u}_{i1}, \ldots, \mathbf{u}_{iq}]$, and each basis vector $\mathbf{u}_{ij}$ is intended to ***unsupervisedly*** discover an "eigen-dimension" which holds an interpretable variation of the synthesized samples such as race, pose, and lighting for human face.

3. EigenGAN

In this section, we first introduce the EigenGAN generator design with layer-wise subspace models in Sec. 3.1. Then in Sec. 3.2, we make a discussion from the linear case to the general case of EigenGAN and finally provide a manifold perspective.

3.1. Generator with Layer-Wise Subspaces

Fig. 2 shows our generator architecture. Our target is to learn a t-layer generator mapping from a set of latent variables $\{\mathbf{z}_i \in \mathbb{R}^q \,|\, \mathbf{z}_i \sim \mathcal{N}_q(\mathbf{0}, \mathbf{I}), i = 1, \ldots, t\}$ to the synthesized image $\mathbf{x} = G(\mathbf{z}_1, \ldots, \mathbf{z}_t)$, where $\mathbf{z}_i$ is directly injected into the i^{th} generator layer; q denotes the number of dimensions of each subspace.

In the i^{th} layer, we embed a linear subspace model $S_i = (\mathbf{U}_i, \mathbf{L}_i, \boldsymbol{\mu}_i)$ where

- $\mathbf{U}_i = [\mathbf{u}_{i1}, \ldots, \mathbf{u}_{iq}]$ is the orthonormal basis of the subspace, and each basis vector $\mathbf{u}_{ij} \in \mathbb{R}^{H_i \times W_i \times C_i}$ is intended to unsupervisedly discover an "eigen-dimension" which holds an interpretable variation of the synthesized samples.

- $\mathbf{L}_i = \text{diag}(l_{i1}, \ldots, l_{iq})$ is a diagonal matrix with l_{ij} deciding the "importance" of the basis vector $\mathbf{u}_{ij}$. To be specific, high absolute value of l_{ij} means that $\mathbf{u}_{ij}$ controls major variation of the the i^{th} layer while low absolute value denotes minor variation, which can be also viewed as a kind of dimension selection.

- $\boldsymbol{\mu}_i$ denotes the origin of the subspace.

Then, we use the i^{th} latent variable $\mathbf{z}_i = [z_{i1}, \ldots, z_{iq}]^{\text{T}}$ as the coordinates (linear combination coefficients) to sample a point from the subspace S_i:

$$\boldsymbol{\phi}_i = \mathbf{U}_i\mathbf{L}_i\mathbf{z}_i + \boldsymbol{\mu}_i \tag{1}$$

$$= \sum_{j=1}^{q} z_{ij} l_{ij} \mathbf{u}_{ij} + \boldsymbol{\mu}_i. \tag{2}$$

This sample point $\boldsymbol{\phi}_i$ will be added to the network feature of the i^{th} layer as stated next.

Let $\mathbf{h}_i \in \mathbb{R}^{H_i \times W_i \times C_i}$ denote the feature maps of the i^{th} layer and $\mathbf{x} = \mathbf{h}_{t+1}$ denote the final synthesized image, then the forward relation between the adjacent layers is

$$\mathbf{h}_{i+1} = \text{Conv2x}\left(\mathbf{h}_i + f\left(\boldsymbol{\phi}_i\right)\right), \quad i = 1, \ldots, t, \tag{3}$$

where "Conv2x" denotes transposed convolutions that double the resolution of the feature maps, and f can be identity function or a simple transform (1x1 convolution in practice). As can be seen from Eq. (3), the sample point $\boldsymbol{\phi}_i$ from the subspace S_i directly interacts with the network feature $\mathbf{h}_i$ of the i^{th} layer. Therefore, the subspace S_i directly determines the variations of the i^{th} layer, more concretely, q coordinates $\mathbf{z}_i = [z_{i1}, \ldots, z_{iq}]^{\text{T}}$ respectively control q different variations.

Besides, we also inject a noise input $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ to the bottom of the generator intended to capture the rest variations missed by the subspaces, as follows,

$$\mathbf{h}_1 = \text{FC}\left(\boldsymbol{\epsilon}\right), \tag{4}$$

where "FC" denotes the fully-connected layer.

The bases $\{\mathbf{U}_i\}_{i=1}^t$, the importance matrices $\{\mathbf{L}_i\}_{i=1}^t$, the origins $\{\boldsymbol{\mu}_i\}_{i=1}^t$, and the convolution kernels are all learnable parameters and the learning can be driven by various adversarial losses [10, 25, 1, 28]. In this paper, hinge loss [28] is used for the adversarial training. Besides, the orthogonality of $\mathbf{U}_i$ is achieved by the regularization of

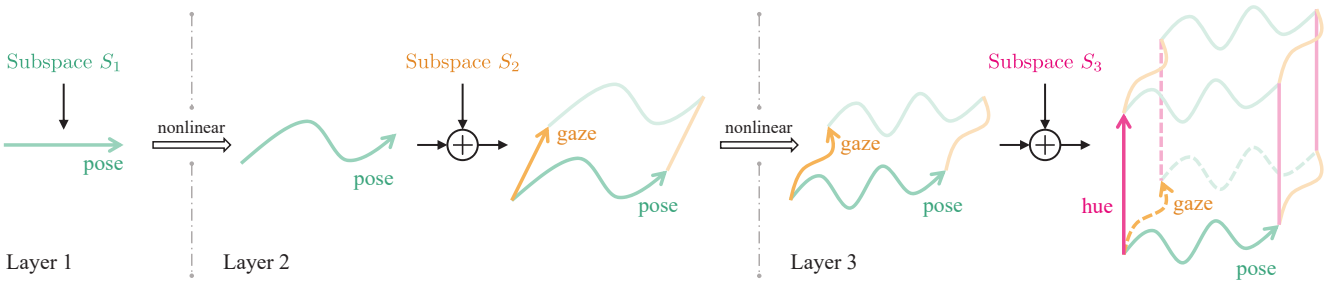


Figure 3. Manifold perspective of EigenGAN. At each layer, a linear subspace is added to the feature manifold, expanding the manifold with "straight" directions along which the variation of some semantic attributes are linear. At the end of each layer, nonlinear mappings "bend" these straight directions, yet another subspace at the next layer will continue to add new straight directions. Here, we only show one semantic direction of each subspace just for simplicity, generally, each subspace contains multiple orthogonal directions.

$\|\mathbf{U}_i^{\mathrm{T}}\mathbf{U}_i - \mathbf{I}\|_F^2$. After training, each latent dimension z_{ij} can explicitly control an interpretable variation corresponding to the semantic of its layer.

3.2. Discussion

Linear Case To better understand how our model works, we first discuss the linear case of our EigenGAN. Adapted from Eq. (1), the linear model is formulated as below,

$$\mathbf{x} = \mathbf{U}\mathbf{L}\mathbf{z} + \boldsymbol{\mu} + \sigma\boldsymbol{\epsilon}. \tag{5}$$

This equation relates a d-dimension observation vector $\mathbf{x}$ to a corresponding q-dimension ($q < d$) latent variables $\mathbf{z} \sim \mathcal{N}_q(\mathbf{0}, \mathbf{I})$ by an affine transform $\mathbf{U}\mathbf{L}$ and a translation $\boldsymbol{\mu}$. Besides, a noise vector $\boldsymbol{\epsilon} \sim \mathcal{N}_d(\mathbf{0}, \mathbf{I})$ is introduce to compensate the missing energy. We also constrain $\mathbf{U}$ with orthonormal columns and $\mathbf{L}$ as a diagonal matrix like the general case in Sec. 3.1. This formulation can also be regarded as a constrained case of Probabilistic PCA [37].

To estimate $\mathbf{U}$, $\mathbf{L}$, $\boldsymbol{\mu}$ and σ in Eq. (5) with n observations $\{\mathbf{x}_i\}_{i=1}^n$, an analytical solution is maximum likelihood estimation (MLE). Please refer to the appendix for detailed derivation of the MLE results. One important result is that the columns of $\mathbf{U}^{\mathrm{ML}} = \left[\mathbf{u}_1^{\mathrm{ML}}, \ldots, \mathbf{u}_q^{\mathrm{ML}}\right]$ are the eigenvectors of data covariance corresponding to the q largest eigenvalues, which is exactly the same as the result of PCA [15]. That is to say, *the linear EigenGAN is able to discover the principal dimensions*, which gives us a strong insight and motivation to embed such a linear model (Eq. (5)) hierarchically into different generator layers as stated in Sec. 3.1.

EigenGAN (General Case) With the insight of the linear case, we suppose that the linear subspace model embedded in a specific layer can capture the principal semantic variations of that layer, and these principal variations are orthogonally separated into the basis vectors. In consequence, each basis vector discovers an "eigen-dimension" that controls an attribute or interpretable variation corresponding to the semantics of its layer.

Manifold Perspective Fig. 3 shows a manifold perspective of EigenGAN. From this aspect, the subspace of each layer expands the feature manifold with "straight" directions along which the variations of some semantic attributes are linear. At the end of each layer, nonlinear mappings "bend" these straight directions, yet another subspace at the next layer will continue to add new straight directions. In a word, EigenGAN *decomposes the data generation modeling into hierarchical dimension expanding steps*, i.e., expanding the feature manifold with linear semantic dimensions layer-by-layer.

4. Experiments

Dataset We test our method on CelebA [23], FFHQ [18], and Danbooru2019 Portraits [4]. CelebA contains 202,599 celebrity face images with annotations of 40 binary attributes. FFHQ contains 70,000 high-quality face images and Danbooru2019 Portraits contains 302,652 anime face images. We use CelebA attributes for the quantitative evaluations and use FFHQ and Danbooru2019 Portraits for more visual results.

Implementation Details We use hinge loss [28] and R_1 penalty [26] for the adversarial training. We adopt Adam solver [19] for all networks and parameter moving average for the generator. The generator is designed for 256×256 images and contains 6 upsampling convolutional blocks. A whole block with one upsampling is defined as a "layer", and one linear subspace with 6 basis vectors is embedded into each generator layer. Please refer to the appendix for detailed network architectures.

4.1. Discovered Semantic Attributes

Visual Analysis Fig. 4 shows the semantic attributes learned by the subspace of different layers, where "Li Dj" means the j$^{\text{th}}$ dimension of the i$^{\text{th}}$ layer and smaller index of layer means deeper. As shown, moving along an eigen-dimension (i.e., a basis vector of a subspace), the synthe-

Figure 4. Discovered semantic attributes at different layers for CelebA dataset [23]. Traversing the coordinate value in $[-4.5\sigma, 4.5\sigma]$, each dimension controls an attribute, colored in blue. The attributes colored in green are the most correlated CelebA attributes, and the bracket value is the entropy coefficient: what fraction of the information of the CelebA attribute is contained in the corresponding dimension. "Li Dj" means the j^{th} dimension of the i^{th} layer. We only show the most meaningful dimensions, please refer to the appendix for all dimensions.

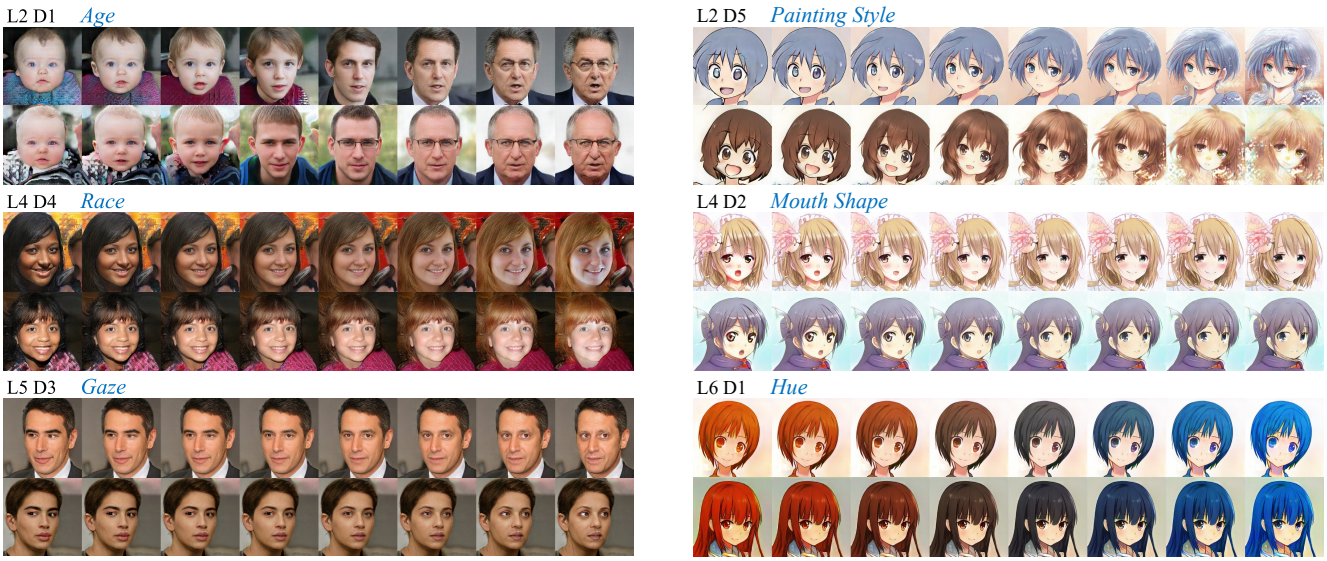


Figure 5. Interpretable dimensions of FFHQ dataset [18] and anime dataset [4].

sized images consistently change by an interpretable meaning. Shallower layers tend to learn lower-level attributes, e.g., L6 and L5 learn color-related attributes such as "Hue" in L6 and "Hair Color" in L5. As the layer goes deeper, the generator discovers attributes with higher-level or more complicated concepts. For example, L4 and L3 learn geometric or structural attributes such as "Face Shape" in L4 and "Body Side" in L3. Deep layers tend to learn multiple attributes in one dimension, e.g., L1 D5 learns "Facial Hair" on the left axis but "Hat" on the right axis. Besides, entanglement of attributes is likely to happen in deep layer dimensions, e.g., L2 D2 learns to simultaneously change "Hair Side" and "Background Texture Orientation", because complex attribute composition might mislead the network into believing their whole as one high-level attribute.

In summary, shallow layers learn low-level or simple attributes while deep layers learn high-level or complicated attributes. Entanglement might happen in some dimensions of deep layers, and this is one of our limitations. Nonetheless, the entanglement is interpretable, i.e., we can identify what attributes are entangled in a dimension. Moreover, our method can still discover well disentangled dimensions that are highly consistent with the visual concepts of humans. Fig. 5 show additional results of FFHQ dataset [18] and Danbooru2019 Portraits dataset [4]. Please refer to the appendix for more results and more interpretable dimensions.

Identifying Well-Defined Attributes In the previous part, we visually identify semantic attributes for each dimension. In this part, we identify the attributes in a statistical manner, utilizing 40 well-defined binary attributes in CelebA dataset [23]. Specifically, we investigate the correlation between a dimension Z and a CelebA attribute Y in terms of *entropy coefficient* (normalized mutual information), which represents what fraction of the information of Y is contained in Z:

$$\mathbf{U}(Y|Z) = \frac{\mathbf{I}(Y;Z)}{\mathbf{H}(Y)} = \frac{\mathbf{H}(Y) - \mathbf{H}(Y|Z)}{\mathbf{H}(Y)} \in [0,1] \quad (6)$$

where

$$\begin{aligned}\mathbf{H}(Y|Z) = \int_{\mathcal{Z}} p_{\mathrm{Z}}(z)\big[&- p_{\mathrm{Y}|\mathrm{Z}}(y{=}1|z)\ln\left(p_{\mathrm{Y}|\mathrm{Z}}(y{=}1|z)\right) \\ &- (1-p_{\mathrm{Y}|\mathrm{Z}}(y{=}1|z))\ln\left(1-p_{\mathrm{Y}|\mathrm{Z}}(y{=}1|z)\right)\big]dz, \quad (7)\end{aligned}$$

$$\begin{aligned}\mathbf{H}(Y) = &- p_{\mathrm{Y}}(y{=}1)\ln\left(p_{\mathrm{Y}}(y{=}1)\right) \\ &- (1-p_{\mathrm{Y}}(y{=}1))\ln\left(1-p_{\mathrm{Y}}(y{=}1)\right). \quad (8)\end{aligned}$$

$p_{\mathrm{Y}|\mathrm{Z}}(y{=}1|z)$ and $p_{\mathrm{Y}}(y{=}1)$ can be calculated by[1]

$$p_{\mathrm{Y}|\mathrm{Z}}(y{=}1|z) = \int_{\mathcal{X}} p_{\mathrm{Y}|\mathrm{X}}(y{=}1|x)p_{\mathrm{G}}(x|z)dx, \quad (9)$$

$$p_{\mathrm{Y}}(y{=}1) = \int_{\mathcal{Z}} p_{\mathrm{Y}|\mathrm{Z}}(y{=}1|z)p_{\mathrm{Z}}(z)dz, \quad (10)$$

where $p_{\mathrm{G}}(x|z)$ is the generator distribution, and $p_{\mathrm{Y}|\mathrm{X}}(y = 1|x)$ is the posterior distribution which is approximated by a pre-trained attribute classifier on CelebA dataset. We set $p_{\mathrm{Z}}(z)$ as $\mathcal{U}[-4.5, 4.5]$ and discretize it into 100 equal bins for approximation of the integral $\int_{\mathcal{Z}} \cdot\, p_{\mathrm{Z}}(z)dz$ in Eq. (7) and (10); and we sample 1000 x from the generator $p_{\mathrm{G}}(x|z)$ in each bin of z, then approximate the integral $\int_{\mathcal{X}} \cdot\, p_{\mathrm{G}}(x|z)dx$ in Eq. (9) by averaging over the samples.

[1] y and z are conditionally independent given x, i.e., $p_{\mathrm{Y}|\mathrm{X}\mathrm{Z}}(y = 1|x, z) = p_{\mathrm{Y}|\mathrm{X}}(y{=}1|x)$.

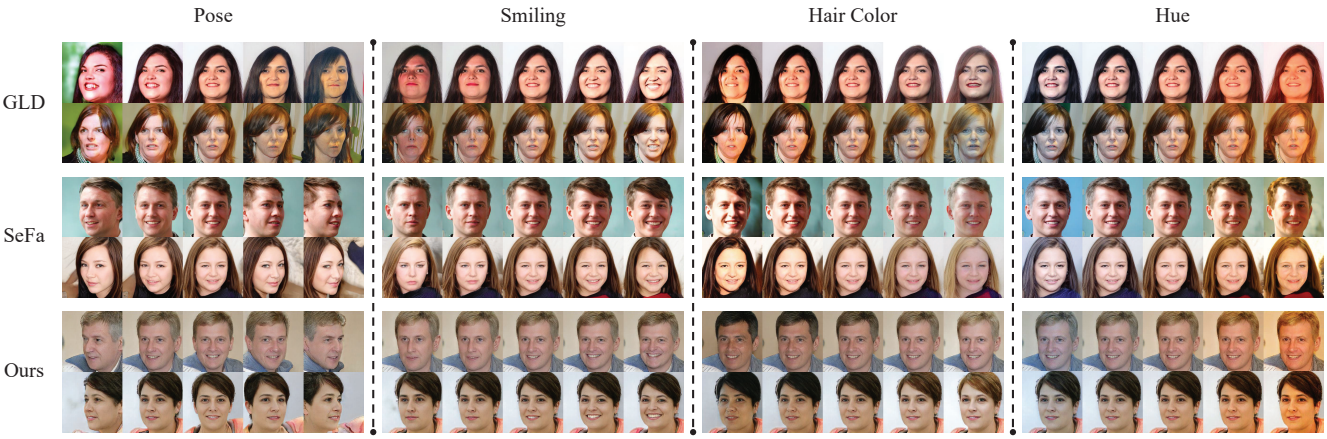


Figure 6. Qualitative comparison among GLD [38], SeFa [36], and our EigenGAN.

Table 1. Correlation between the discovered attributes and the CelebA attributes in terms of entropy coefficient. Each row denotes a discovered attributes by GLD [38], SeFa [36] and our EigenGAN, and each column denotes a CelebA attribute.

GLD	Gender	Eyeglasses	Smiling	Black Hair
Gender	28%	2%	11%	3%
Eyeglasses	3%	5%	5%	4%
Smiling	0%	0%	24%	1%
Black Hair	1%	0%	1%	9%

SeFa	Gender	Eyeglasses	Smiling	Black Hair
Gender	49%	14%	2%	4%
Eyeglasses	5%	49%	2%	0%
Smiling	1%	1%	52%	8%
Black Hair	1%	0%	1%	18%

Ours	Gender	Eyeglasses	Smiling	Black Hair
Gender	57%	14%	12%	2%
Eyeglasses	2%	33%	0%	1%
Smiling	1%	0%	55%	2%
Black Hair	0%	0%	0%	38%

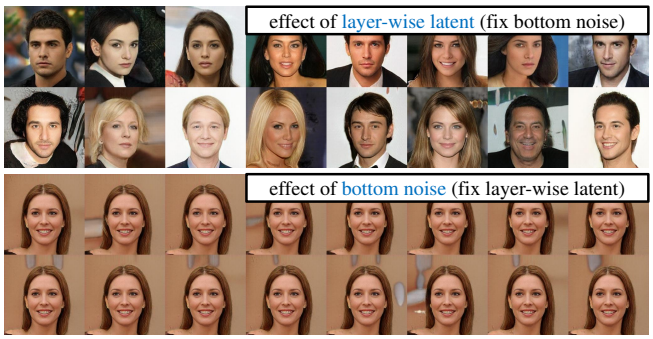
For each dimension in Fig. 4, the five most correlated CelebA attributes with entropy coefficient larger than 30% are shown (green text). As shown, the identified CelebA attributes according to entropy coefficient are highly consistent with our visual perception. Several dimensions have no correlated CelebA attributes just because the attributes represented by these dimensions are not included in the CelebA, but these dimensions are still interpretable, e.g., L4 D1 learns "Pose" which is not a CelebA attribute. Several dimensions correlate to multiple CelebA attributes mainly because these CelebA attributes are themselves highly correlated, e.g., L4 D5 learns "Smile" therefore it has high entropy coefficient for "Smile" correlated attributes: "High Cheekbones", "Mouth Open", and "Narrow Eyes". In conclusion, this experiment statistically verifies that, EigenGAN can indeed discover interpretable dimensions controlling attributes which are highly consistent with human-defined ones (e.g., CelebA attributes).

Comparison In this part, we compare our method to two state-of-the-art post-processing methods GANLatentDiscovery (GLD) [38] and SeFa [36]. We use their official models with GLD trained on StyleGAN2-FFHQ-1024 and SeFa trained on StyleGAN-FFHQ-256. Fig. 6 shows the qualitative comparison. As can be seen, both SeFa and our EigenGAN can achieve smooth and consistent change of the identified attributes, more natural and realistic than GLD. However, entanglement to some extent still happens in all three methods, e.g., "Pose" dimension also changes lighting in GLD, "Smiling" dimension also changes bangs in SeFa, and "Hair Color" dimension also changes skin color in EigenGAN. This is because all of them are unsupervised methods, and it is difficult to precisely decouple all the attributes without any supervision. Table 1 shows the quantitative comparison of the correlation between the discovered attributes and the CelebA attributes, in terms of entropy coefficient introduced in the previous part. As can be seen, the discovered attributes by both SeFa and our EigenGAN have high correlation to the corresponding CelebA attributes, demonstrating that both methods can indeed discover meaningful semantic attributes. Overall, our EigenGAN achieves comparable performance to the state-of-the-art SeFa on the learned attributes and disentanglement, and both methods perform better than GLD.

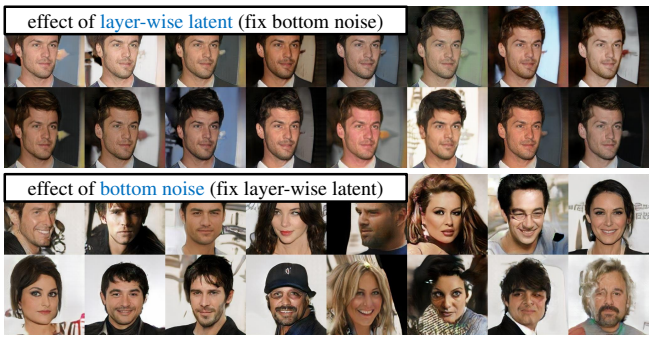
4.2. Model Analysis

Effect of the Latent Variables EigenGAN contains two kinds of latent variables: 1) layer-wise latent variables $\{\mathbf{z}_i\}_{i=1}^{t}$, which are used as the subspace coordinates; 2) bottom noise $\boldsymbol{\epsilon}$ to compensate the missing variations. In Fig. 7a, we respectively fix one of them and randomly sample another to generate images. As can be seen, the layer-wise latent variables $\{\mathbf{z}_i\}_{i=1}^{t}$ dominate the major variations while the bottom noise $\boldsymbol{\epsilon}$ captures subtle changes. That is to say, EigenGAN tends to put major variations into the layer-wise latent variables rather than the bottom noise used in typical GANs, but the bottom noise can still capture some subtle variations missed by the subspace models.

Effect of the Subspace Model We remove all the layer-wise subspace models to investigate their effect, instead, we directly add the layer-wise latent variables to the network features. As shown in Fig. 7b, without the subspace models, the layer-wise latent variables can only capture minor variations, which is completely opposite to the original setting in Fig. 7a. In conclusion, the subspace model is the key



(a) With the subspace models (EigenGAN), major variations are captured by the layer-wise latent variables.



(b) Without the subspace models (typical GANs), major variations are captured by the bottom noise.

Figure 7. Effect of the layer-wise latent variables (top) and the bottom noise (down).

Table 2. Basis similarity with PCA. $P = \mathcal{N}_d(\mathbf{0}, \mathbf{I})$.

GAN Loss	Data Rank → Subspace Rank							
	5→1	5→3	10→1	10→3	10→5	20→1	20→5	20→10
KL-f-GAN [30]	1.00	0.98	0.99	0.90	0.93	0.97	0.78	0.79
Vanilla GAN [10]	1.00	0.99	1.00	0.90	0.94	0.98	0.77	0.81
WGAN [11]	0.99	0.98	1.00	0.89	0.92	0.99	0.76	0.83
LSGAN [25]	0.99	0.99	1.00	0.89	0.92	0.99	0.76	0.80
HingeGAN [28]	0.99	0.99	1.00	0.92	0.93	0.96	0.77	0.81

Table 3. Basis similarity with PCA. $P = \mathcal{U}_d(0, 1)$.

GAN Loss	Data Rank → Subspace Rank							
	5→1	5→3	10→1	10→3	10→5	20→1	20→5	20→10
KL-f-GAN [30]	0.96	0.98	0.97	0.89	0.93	0.89	0.72	0.82
Vanilla GAN [10]	0.97	0.97	0.97	0.92	0.92	0.92	0.76	0.84
WGAN [11]	0.98	0.97	0.98	0.93	0.94	0.98	0.77	0.84
LSGAN [25]	0.97	0.97	0.96	0.89	0.95	0.91	0.74	0.82
HingeGAN [28]	0.97	0.98	0.97	0.87	0.94	0.92	0.75	0.82

point to enable the generator to put major variations into the layer-wise variables, therefore can further let the layer-wise variables capture different semantics of different layers.

Linear Case Study Sec. 3.2 theoretically proves that the linear case of EigenGAN can discover the principal components under maximum likelihood estimation (MLE). In this part, we validate this statement by applying adversarial training on the linear EigenGAN (we do not directly use MLE since we train the general EigenGAN with adversarial loss rather than MLE objective, and we keep this consistency between the linear and the general case). Specifically, we use the linear EigenGAN to learn a low-rank subspace model for toy datasets, then compare the basis vectors learned by our model and learned by PCA in terms of cosine similarity. The toy datasets are generated as follows,

$$\mathcal{D}_{\mathbf{A},\mathbf{b},P} = \{\mathbf{y}_i = \mathbf{A}\mathbf{x}_i + \mathbf{b} \mid \mathbf{x}_i \sim P\} \tag{11}$$

where $\mathbf{A}$ is a random transform matrix, $\mathbf{b}$ is a random translation vector, and P is a distribution selected from $\mathcal{N}_d(\mathbf{0}, \mathbf{I})$ or $\mathcal{U}_d(0, 1)$. We test typical adversarial losses including Vanilla GAN [10], LSGAN [25], WGAN [11], HingeGAN [28], and f-GAN [30] with KL divergence (KL-f-GAN). Note that the objective of KL-f-GAN is theoretically equivalent to MLE, thus we are actually also testing MLE in the adversarial training manner.

Table 2 and Table 3 report the average similarity between EigenGAN basis vectors and PCA basis vectors, where each result is the average over 100 random toy datasets. As can be seen, when the data rank is no more than 10, EigenGAN basis is highly similar to PCA basis with cosine similarity of about 0.9-1.0. When the data rank increases to 20, there are two situations: 1) if we only search the most principal one basis vector (20→1), the vectors found by linear EigenGAN and by PCA are still very close; 2) but if we want to find 5 or more basis vectors, the average similarity decreases to 0.7-0.8. We suppose the reason is that higher dimension data leads to the curse of dimensionality and further results in learning instability. Besides, various GAN losses have very consistent results, which shows the potential of generalizability of our theoretical results in Sec. 3.2 from KL divergence (MLE) to more general statistical distance such JS divergence and Wasserstein distance. In conclusion, we experimentally verify the theoretical statement that the linear EigenGAN can indeed discover principal components.

5. Limitations and Future Works

Discovered semantic attributes are not always the same at different training times in two cases: 1) E.g., sometimes the gender and pose are learned as separated dimensions but sometimes are entangled in one dimension at a deeper layer. This is because, without supervision, some complex attribute compositions might mislead the model into believing their whole as one higher-level attribute. 2) Sometimes the model can discover a specific attribute but sometimes cannot, such as eyeglasses, mainly because these attributes appear less frequently in the dataset. Future works will study the layer-wise eigen-learning with better disentanglement techniques and more powerful GAN architectures.

Acknowledgement This work is partially supported the National Key Research and Development Program of China (No. 2017YFA0700800) and the Natural Science Foundation of China (No. 61772496 and No. 61976219).

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. In *Int. Conf. Mach. Learn.*, 2017. 2, 3

[2] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017. 1

[3] David Bau, Jun-Yan Zhu, Hendrik Strobelt, Bolei Zhou, Joshua B. Tenenbaum, William T. Freeman, and Antonio Torralba. Gan dissection: Visualizing and understanding generative adversarial networks. In *Int. Conf. Learn. Represent.*, 2019. 1, 2

[4] Gwern Branwen, Anonymous, and Danbooru Community. Danbooru2019 portraits: A large-scale anime head illustration dataset, 2019. 4, 6

[5] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *Int. Conf. Learn. Represent.*, 2018. 2

[6] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Adv. Neural Inform. Process. Syst.*, 2016. 2

[7] Emily Denton, Soumith Chintala, Arthur Szlam, and Rob Fergus. Deep generative image models using a laplacian pyramid of adversarial networks. In *Adv. Neural Inform. Process. Syst.*, 2015. 2

[8] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. In *Int. Conf. Learn. Represent.*, 2017. 2

[9] Lore Goetschalckx, Alex Andonian, Aude Oliva, and Phillip Isola. Ganalyze: Toward visual definitions of cognitive image properties. In *Int. Conf. Comput. Vis.*, 2019. 2

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. In *Adv. Neural Inform. Process. Syst.*, 2014. 2, 3, 8

[11] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. In *Adv. Neural Inform. Process. Syst.*, 2017. 8

[12] Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. Ganspace: Discovering interpretable gan controls. *Adv. Neural Inform. Process. Syst.*, 2020. 1, 2

[13] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017. 2

[14] Insu Jeon, Wonkwang Lee, Myeongjang Pyeon, and Gunhee Kim. Ib-gan: Disentangled representation learning with information bottleneck gan. In *AAAI*, 2021. 2

[15] Ian T Jolliffe. *Principal component analysis*. 1986. 2, 4

[16] Takuhiro Kaneko, Kaoru Hiramatsu, and Kunio Kashino. Generative attribute controller with conditional filtered generative adversarial networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2017. 2

[17] Takuhiro Kaneko, Kaoru Hiramatsu, and Kunio Kashino. Generative adversarial image synthesis with decision tree latent controller. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2018. 2

[18] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 1, 2, 4, 6

[19] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Int. Conf. Learn. Represent.*, 2015. 4

[20] Wonkwang Lee, Donggyun Kim, Seunghoon Hong, and Honglak Lee. High-fidelity synthesis with disentangled representation. In *Eur. Conf. Comput. Vis.*, 2020. 2

[21] Zinan Lin, Kiran Thekumparampil, Giulia Fanti, and Sewoong Oh. Infogan-cr and modelcentrality: Self-supervised model training and selection for disentangling gans. In *Int. Conf. Mach. Learn.*, 2020. 2

[22] Bingchen Liu, Yizhe Zhu, Zuohui Fu, Gerard de Melo, and Ahmed Elgammal. Oogan: Disentangling gan with one-hot sampling and orthogonal regularization. In *AAAI*, 2020. 2

[23] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, 2015. 4, 5, 6

[24] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. In *Int. Conf. Learn. Represent.*, 2016. 2

[25] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *Int. Conf. Comput. Vis.*, 2017. 2, 3, 8

[26] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In *Int. Conf. Mach. Learn.*, 2018. 2, 4

[27] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv:1411.1784*, 2014. 2

[28] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *Int. Conf. Learn. Represent.*, 2018. 2, 3, 4, 8

[29] Takeru Miyato and Masanori Koyama. cgans with projection discriminator. In *Int. Conf. Learn. Represent.*, 2018. 2

[30] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In *Adv. Neural Inform. Process. Syst.*, 2016. 2, 8

[31] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *Int. Conf. Mach. Learn.*, 2017. 2

[32] Antoine Plumerault, Hervé Le Borgne, and Céline Hudelot. Controlling generative models with continuous factors of variations. In *Int. Conf. Learn. Represent.*, 2019. 2

[33] Aditya Ramesh, Youngduck Choi, and Yann LeCun. A spectral regularizer for unsupervised disentanglement. *arXiv:1812.01161*, 2018. 2

[34] Kevin Roth, Aurelien Lucchi, Sebastian Nowozin, and Thomas Hofmann. Stabilizing training of generative adver-

sarial networks through regularization. In *Adv. Neural Inform. Process. Syst.*, 2017. 2

[35] Yujun Shen, Ceyuan Yang, Xiaoou Tang, and Bolei Zhou. Interfacegan: Interpreting the disentangled face representation learned by gans. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020. 2

[36] Yujun Shen and Bolei Zhou. Closed-form factorization of latent semantics in gans. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2021. 1, 2, 7

[37] Michael E Tipping and Christopher M Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, 1999. 4

[38] Andrey Voynov and Artem Babenko. Unsupervised discovery of interpretable directions in the gan latent space. In *Int. Conf. Mach. Learn.*, 2020. 2, 7

[39] Ceyuan Yang, Yujun Shen, and Bolei Zhou. Semantic hierarchy emerges in deep generative representations for scene synthesis. *Int. J. Comput. Vis.*, 2021. 1, 2

[40] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *Eur. Conf. Comput. Vis.*, 2014. 1

[41] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Int. Conf. Comput. Vis.*, 2017. 2

[42] Bolei Zhou, Aditya Khosla, Àgata Lapedriza, Aude Oliva, and Antonio Torralba. Object detectors emerge in deep scene cnns. In *Int. Conf. Learn. Represent.*, 2015. 1

[43] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Int. Conf. Comput. Vis.*, 2017. 2