Adversarial Latent Autoencoders

Stanislav Pidhorskyi Donald A. Adjeroh Gianfranco Doretto
Lane Department of Computer Science and Electrical Engineering
West Virginia University, Morgantown, WV 26506
{ stpidhorskyi, daadjeroh, gidoretto }@mix.wvu.edu

Abstract

Autoencoder networks are unsupervised approaches aiming at combining generative and representational properties by learning simultaneously an encoder-generator map. Although studied extensively, the issues of whether they have the same generative power of GANs, or learn disentangled representations, have not been fully addressed. We introduce an autoencoder that tackles these issues jointly, which we call Adversarial Latent Autoencoder (ALAE). *It is a general architecture that can leverage recent improvements on GAN training procedures. We designed two autoencoders: one based on a MLP encoder, and another based on a StyleGAN generator, which we call StyleALAE. We verify the disentanglement properties of both architectures. We show that StyleALAE can not only generate* 1024×1024 *face images with comparable quality of StyleGAN, but at the same resolution can also produce face reconstructions and manipulations based on real images. This makes ALAE the first autoencoder able to compare with, and go beyond the capabilities of a generator-only type of architecture.*

1. Introduction

Generative Adversarial Networks (GAN) [13] have emerged as one of the dominant unsupervised approaches for computer vision and beyond. Their strength relates to their remarkable ability to represent complex probability distributions, like the face manifold [33], or the bedroom images manifold [53], which they do by learning a *generator* map from a known distribution onto the data space. Just as important are the approaches that aim at learning an *encoder* map from the data to a latent space. They allow learning suitable representations of the data for the task at hand, either in a supervised [29, 46, 40, 14, 52], or unsupervised [37, 58, 19, 25, 4, 3] manner.

Autoencoder (AE) [28, 41] networks are unsupervised approaches aiming at combining the "generative" as well as the "representational" properties by learning simultaneously an *encoder-generator* map. General issues subject of investigation in AE structures are whether they can: (a) have the same generative power as GANs; and, (b) learn disentangled representations [1]. Several works have addressed (a) [35, 31, 6, 9, 20]. An important testbed for success has been the ability for an AE to generate face images as rich and sharp as those produced by a GAN [23]. Progress has been made but victory has not been declared. A sizable amount of work has addressed also (b) [19, 25, 10], but not jointly with (a).

We introduce an AE architecture that is general, and has generative power comparable to GANs while learning a less entangled representation. We observed that every AE approach makes the same assumption: *the latent space should have a probability distribution that is fixed* a priori *and the autoencoder should match it.* On the other hand, it has been shown in [24], the state-of-the-art for synthetic image generation with GANs, that an intermediate latent space, far enough from the imposed input space, tends to have improved disentanglement properties.

The observation above has inspired the proposed approach. We designed an AE architecture where we allow the latent distribution to be learned from data to address entanglement (A). The output data distribution is learned with an adversarial strategy (B). Thus, we retain the generative properties of GANs, as well as the ability to build on the recent advances in this area. For instance, we can seamlessly include independent sources of stochasticity, which have proven essential for generating image details, or can leverage recent improvements on GAN loss functions, regularization, and hyperparameters tuning [2, 30, 38, 34, 36, 3]. Finally, to implement (A) and (B) we impose the AE reciprocity in the latent space (C). Therefore, we can avoid using reconstruction losses based on simple ℓ_2 norm that operate in data space, where they are often suboptimal, like for the image space. We regard the unique combination of (A), (B), and (C) as the major techical novelty and strength of the approach. Since it works on the latent space, rather than autoencoding the data space, we named it *Adversarial Latent Autoencoder (ALAE)*.

We designed two ALAEs, one with a multilayer perceptron (MLP) as encoder with a symmetric generator, and

another with the generator derived from a StyleGAN [24], which we call *StyleALAE*. For this one, we designed a companion encoder and a progressively growing architecture. We verified qualitatively and quantitatively that both architectures learn a latent space that is more disentangled than the imposed one. In addition, we show qualitative and quantitative results about face and bedroom image generation that are comparable with StyleGAN at the highest resolution of 1024×1024. Since StyleALAE learns also an encoder network, we are able to show at the highest resolution, face reconstructions as well as several image manipulations based on real images rather then generated.

2. Related Work

Our approach builds directly on the vanilla GAN architecture [12]. Since then, a lot of progress has been made in the area of synthetic image generation. LAPGAN [5] and StackGAN [55, 56] train a stack of GANs organized in a multi-resolution pyramid to generate high-resolution images. HDGAN [57] improves by incorporating hierarchically-nested adversarial objectives inside the network hierarchy. In [51] they use a multi-scale generator and discriminator architecture to synthesize high-resolution images with a GAN conditioned on semantic label maps, while in BigGAN [3] they improve the synthesis by applying better regularization techniques. In PGGAN [23] it is shown how high-resolution images can be synthesized by progressively growing the generator and the discriminator of a GAN. The same principle was used in StyleGAN [24], the current state-of-the-art for face image generation, which we adapt it here for our StyleALAE architecture. Other recent work on GANs has focussed on improving the stability and robustness of the training [44]. New loss functions have been introduced [2], along with gradient regularization methods [39, 36], weight normalization techniques [38], and learning rate equalization [23]. Our framework is amenable to these improvements, as we explain in later sections.

Variational AE architectures [28, 41] have not only been appreciated for their theoretical foundation, but also for their stability during training, and the ability to provide insightful representations. Indeed, they stimulated research in the area of disentanglement [1], allowing learning representations with controlled degree of disentanglement between factors of variation in [19], and subsequent improvements in [25], leading to more elaborate metrics for disentanglement quantification [10, 4, 24], which we also use to analyze the properties of our approach. VAEs have also been extended to learn a latent prior different than a normal distribution, thus achieving significantly better models [48].

A lot of progress has been made towards combining the benefits of GANs and VAEs. AAE [35] has been the precursor of those approaches, followed by VAE/GAN [31] with a more direct approach. BiGAN [6] and ALI [9] provide an elegant framework fully adversarial, whereas VEEGAN [47] and AGE [49] pioneered the use of the latent space for autoencoding and advocated the reduction of the architecture complexity. PIONEER [15] and IntroVAE [20] followed this line, with the latter providing the best generation results in this category. Section 4.1 describes how the proposed approach compares with those listed here.

Finally, we quickly mention other approaches that have shown promising results with representing image data distributions. Those include autoregressive [50] and flow-based methods [27]. The former forego the use of a latent representation, but the latter does not.

3. Preliminaries

A Generative Adversarial Network (GAN) [13] is composed of a generator network $\mathtt{G}$ mapping from a space $\mathcal{Z}$ onto a *data space* $\mathcal{X}$, and a discriminator network $\mathtt{D}$ mapping from $\mathcal{X}$ onto $\mathbb{R}$. The $\mathcal{Z}$ space is characterized by a known distribution $p(z)$. By sampling from $p(z)$, the generator $\mathtt{G}$ produces data representing a *synthetic* distribution $q(x)$. Given training data $\mathcal{D}$ drawn from a *real* distribution $p_{\mathcal{D}}(x)$, a GAN network aims at learning $\mathtt{G}$ so that $q(x)$ is as close to $p_{\mathcal{D}}(x)$ as possible. This is achieved by setting up a zero-sum two-players game with the discriminator $\mathtt{D}$. The role of $\mathtt{D}$ is to distinguish in the most accurate way data coming from the real versus the synthetic distribution, while $\mathtt{G}$ tries to fool $\mathtt{D}$ by generating synthetic data that looks more and more like real.

Following the more general formulation introduced in [39], the GAN learning problem entails finding the minimax with respect to the pair $(\mathtt{G}, \mathtt{D})$ (i.e., the Nash equilibrium), of the value function defined as

$$V(\mathtt{G}, \mathtt{D}) = E_{p_{\mathcal{D}}(x)}[f(\mathtt{D}(x))] + E_{p(z)}[f(-\mathtt{D}(\mathtt{G}(z)))] \,, \quad (1)$$

where $E[\cdot]$ denotes expectation, and $f : \mathbb{R} \to \mathbb{R}$ is a concave function. By setting $f(t) = -\log(1 + \exp(-t))$ we obtain the original GAN formulation [13]; instead, if $f(t) = t$ we obtain the Wasserstein GAN [2].

4. Adversarial Latent Autoencoders

We introduce a novel autoencoder architecture by modifying the original GAN paradigm. We begin by decomposing the generator $\mathtt{G}$ and the discriminator $\mathtt{D}$ in two networks: F, G, and E, D, respectively. This means that

$$\mathtt{G} = G \circ F \,, \quad \text{and} \quad \mathtt{D} = D \circ E \,, \quad (2)$$

see Figure 1. In addition, we assume that the *latent* spaces at the interface between F and G, and between E and D are the same, and we indicate them as $\mathcal{W}$. In the most general case we assume that F is a deterministic map, whereas we

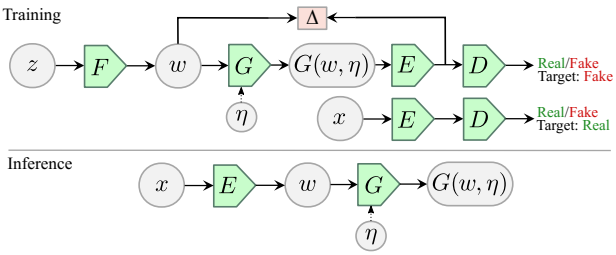

Figure 1: **ALAE Architecture.** Architecture of an Adversarial Latent Autoencoder.

allow E and G to be stochastic. In particular, we assume that G might optionally depend on an independent noisy input η, with a known fixed distribution $p_\eta(\eta)$. We indicate with $G(w, \eta)$ this more general stochastic generator.

Under the above conditions we now consider the distributions at the output of every network. The network F simply maps $p(z)$ onto $q_F(w)$. At the output of G the distribution can be written as

$$q(x) = \int_w \int_\eta q_G(x|w,\eta) q_F(w) p_\eta(\eta)\, \mathrm{d}\eta\, \mathrm{d}w \;, \qquad (3)$$

where $q_G(x|w, \eta)$ is the conditional distribution representing G. Similarly, for the output of E the distribution becomes

$$q_E(w) = \int_x q_E(w|x) q(x) \mathrm{d}x \;, \qquad (4)$$

where $q_E(w|x)$ is the conditional distribution representing E. In (4) if we replace $q(x)$ with $p_\mathcal{D}(x)$ we obtain the distribution $q_{E,\mathcal{D}}(w)$, which describes the output of E when the real data distribution is its input.

Since optimizing (1) leads toward the synthetic distribution matching the real one, i.e., $q(x) = p_\mathcal{D}(x)$, it is obvious from (4) that doing so also leads toward having $q_E(w) = q_{E,\mathcal{D}}(w)$. In addition to that, we propose to ensure that the distribution of the output of E be the same as the distribution at the input of G. This means that we set up an additional goal, which requires that

$$q_F(w) = q_E(w) \;. \qquad (5)$$

In this way we could interpret the pair of networks (G, E) as a *generator-encoder* network that autoencodes the *latent space* $\mathcal{W}$.

If we indicate with $\Delta(p\|q)$ a measure of discrepancy between two distributions p and q, we propose to achieve the goal (5) via regularizing the GAN loss (1) by alternating the following two optimizations

$$\min_{F,G} \max_{E,D} V(G \circ F, D \circ E) \qquad (6)$$

$$\min_{E,G} \Delta(F\|E \circ G \circ F) \qquad (7)$$

where the left and right arguments of Δ indicate the distributions generated by the networks mapping $p(z)$, which correspond to $q_F(w)$ and $q_E(w)$, respectively. We refer to a network optimized according to (6) (7) as an *Adversarial Latent Autoencoder (ALAE)*. The building blocks of an ALAE architecture are depicted in Figure 1.

Autoencoder	(a) Data Distribution	(b) Latent Distribution	(c) Reciprocity Space
VAE [28, 41]	similarity	imposed/divergence	data
AAE [35]	similarity	imposed/adversarial	data
VAE/GAN [31]	similarity	imposed/divergence	data
VampPrior [48]	similarity	learned/divergence	data
BiGAN [6]	adversarial	imposed/adversarial	adversarial
ALI [9]	adversarial	imposed/adversarial	adversarial
VEEGAN [47]	adversarial	imposed/divergence	latent
AGE [49]	adversarial	imposed/adversarial	latent
IntroVAE [20]	adversarial	imposed/adversarial	data
ALAE (ours)	adversarial	learned/divergence	latent

Table 1: Autoencoder criteria used: (a) for matching the real to the synthetic data distribution; (b) for setting/learning the latent distribution; (c) for which space reciprocity is achieved.

4.1. Relation with other autoencoders

Data distribution. In architectures composed by an *encoder* network and a *generator* network, the task of the encoder is to map input data onto a space characterized by a *latent distribution*, whereas the generator is tasked to map latent codes onto a space described by a *data distribution*. Different strategies are used to learn the data distribution. For instance, some approaches impose a similarity criterion on the output of the generator [28, 41, 35, 48], or even learn a similarity metric [31]. Other techniques instead, set up an adversarial game to ensure the generator output matches the training data distribution [6, 9, 47, 49, 20]. This latter approach is what we use for ALAE.

Latent distribution. For the latent space instead, the common practice is to set a desired target latent distribution, and then the encoder is trained to match it either by minimizing a divergence type of similarity [28, 41, 31, 47, 48], or by setting up an adversarial game [35, 6, 9, 49, 20]. Here is where ALAE takes a fundamentally different approach. Indeed, we do not impose the latent distribution, i.e., $q_E(w)$, to match a target distribution. The only condition we set, is given by (5). In other words, we do not want F to be the identity map, and are very much interested in letting the learning process decide what F should be.

Reciprocity. Another aspect of autoecoders is whether and how they achieve reciprocity. This property relates to the ability of the architecture to reconstruct a data sample x from its code w, and viceversa. Clearly, this requires that $x = G(E(x))$, or equivalently that $w = E(G(w))$. In the first case, the network must contain a reconstruction term that operates in the data space. In the latter one, the term operates in the latent space. While most approaches follow the first strategy [28, 41, 35, 31, 20, 48], there are some that implement the second [47, 49], including ALAE. Indeed, this can be achieved by choosing the divergence in (7) to be

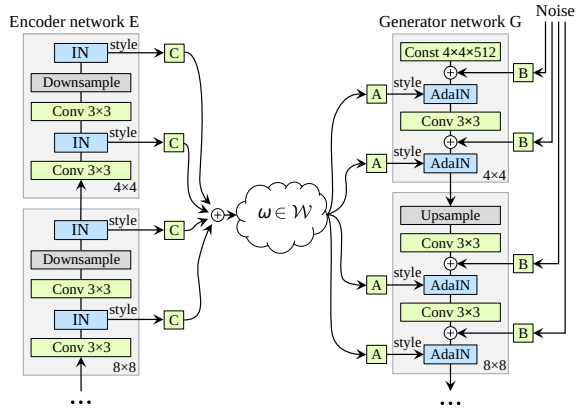

Figure 2: **StyleALAE Architecture.** The StyleALAE encoder has Instance Normalization (IN) layers to extract multiscale style information that is combined into a latent code w via a learnable multilinear map.

the expected coding reconstruction error, as follows

$$\Delta(F\|E\circ G\circ F) = E_{p(z)}\left[\|F(z) - E\circ G\circ F(z)\|_2^2\right] \quad (8)$$

Imposing reciprocity in the latent space gives the significant advantage that simple ℓ_2, ℓ_1 or other norms can be used effectively, regardless of whether they would be inappropriate for the data space. For instance, it is well known that element-wise ℓ_2 norm on image pixel differences does not reflect human visual perception. On the other hand, when used in latent space its meaning is different. For instance, an image translation by a pixel could lead to a large ℓ_2 discrepancy in image space, while in latent space its representation would hardly change at all. Ultimately, using ℓ_2 in image space has been regarded as one of the reasons why autoencoders have not been as successful as GANs in reconstructing/generating sharp images [31]. Another way to address the same issue is by imposing reciprocity adversarially, as it was shown in [6, 9]. Table 1 reports a summary of the main characteristics of most of the recent generator-encoder architectures.

5. StyleALAE

We use ALAE to build an autoencoder that uses a StyleGAN based generator. For this we make our latent space $\mathcal{W}$ play the same role as the intermediate latent space in [24]. Therefore, our G network becomes the part of StyleGAN depicted on the right side of Figure 2. The left side is a novel architecture that we designed to be the encoder E.

Since at every layer, G is driven by a style input, we design E symmetrically, so that from a corresponding layer we extract style information. We do so by inserting Instance Normalization (IN) layers [21], which provide instance averages and standard deviations for every channel. Specifically, if y_i^E is the output of the i-th layer of E, the IN module extracts the statistics $\mu(y_i^E)$ and $\sigma(y_i^E)$ representing the style at that level. The IN module also provides as output the normalized version of the input, which continues down the pipeline with no more style information from that level. Given the information flow between E and G, the architecture is effectively mimicking a multiscale style transfer from E to G, with the difference that there is not an extra input image that provides the content [21, 22].

The set of styles that are inputs to the Adaptive Instance Normalization (AdaIN) layers [21] in G are related linearly to the latent variable w. Therefore, we propose to combine the styles output by the encoder, and to map them onto the latent space, via the following multilinear map

$$w = \sum_{i=1}^{N} C_i \begin{bmatrix}\mu(y_i^E)\\ \sigma(y_i^E)\end{bmatrix} \quad (9)$$

where the C_i's are learnable parameters, and N is the number of layers.

Similarly to [23, 24] we use progressive growing. We start from low-resolution images (4×4 pixels) and progressively increase the resolution by smoothly blending in new blocks to E and G. For the F and D networks we implement them using MLPs. The $\mathcal{Z}$ and $\mathcal{W}$ spaces, and all layers of F and D have the same dimensionality in all our experiments. Moreover, for StyleALAE we follow [24], and chose F to have 8 layers, and we set D to have 3 layers.

6. Implementation

Adversarial losses and regularization. We use a non-saturating loss [13, 36], which in (1) we introduce by setting $f(\cdot)$ to be a SoftPlus function [11]. This is a smooth version of the rectifier activation function, defined as $f(t) = \text{softplus}(t) = \log(1+\exp(t))$. In addition, we use gradient regularization techniques [8, 36, 43]. We utilize R_1 [44,

Algorithm 1 ALAE Training

$\theta_F, \theta_G, \theta_E, \theta_D \leftarrow$ Initialize network parameters
while not converged **do**
  Step I. Update E, and D
  $x \leftarrow$ Random mini-batch from dataset
  $z \leftarrow$ Samples from prior $\mathcal{N}(0, I)$
  $L_{adv}^{E,D} \leftarrow \text{softplus}(D\circ E\circ G\circ F(z))) + \text{softplus}(-D\circ E(x)) + \frac{\gamma}{2}\,\mathrm{E}_{p_{\mathcal{D}}(x)}\left[\|\nabla D\circ E(x)\|^2\right]$
  $\theta_E, \theta_D \leftarrow \text{Adam}(\nabla_{\theta_D,\theta_E} L_{adv}^{E,D}, \theta_D, \theta_E, \alpha, \beta_1, \beta_2)$
  Step II. Update F, and G
  $z \leftarrow$ Samples from prior $\mathcal{N}(0, I)$
  $L_{adv}^{F,G} \leftarrow \text{softplus}(-D\circ E\circ G\circ F(z)))$
  $\theta_F, \theta_G \leftarrow \text{Adam}(\nabla_{\theta_F,\theta_G} L_{adv}^{F,G}, \theta_F, \theta_G, \alpha, \beta_1, \beta_2)$
  Step III. Update E, and G
  $z \leftarrow$ Samples from prior $\mathcal{N}(0, I)$
  $L_{error}^{E,G} \leftarrow \|F(z) - E\circ G\circ F(z)\|_2^2$
  $\theta_E, \theta_G \leftarrow \text{Adam}(\nabla_{\theta_E,\theta_G} L_{error}^{E,G}, \theta_E, \theta_G, \alpha, \beta_1, \beta_2)$
end while

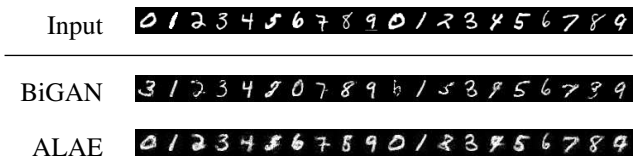

Figure 3: **MNIST reconstruction.** Reconstructions of the permutation-invariant MNIST. Top row: real images. Middle row: BiGAN reconstructions. Bottom row: ALAE reconstructions. The same MLP architecture is used in both methods.

36], a zero-centered gradient penalty term which acts only on real data, and is defined as $\frac{\gamma}{2}\,\mathrm{E}_{p_{\mathcal{D}}(x)}\left[\|\nabla D \circ E(x)\|^2\right]$, where the gradient is taken with respect to the parameters θ_E and θ_D of the networks E and D, respectively.

Training. In order to optimizate (6) (7) we use alternating updates. One iteration is composed of three updating steps: two for (6) and one for (7). Step I updates the discriminator (i.e., networks E and D). Step II updates the generator (i.e., networks F and G). Step III updates the latent space autoencoder (i.e., networks G and E). The procedural details are summarized in Algorithm 1. For updating the weights we use the Adam optimizer [26] with $\beta_1 = 0.0$ and $\beta_2 = 0.99$, coupled with the learning rate equalization technique [23] described below. For non-growing architectures (i.e., MLPs) we use a learning rate of 0.002, and batch size of 128. For growing architectures (i.e., StyleALAE) learning rate and batch size depend on the resolution.

7. Experiments

Code and uncompressed images are available at https://github.com/podgorskiy/ALAE.

7.1. Representation learning with MLP

We train ALAE with MNIST [32], and then use the feature representation for classification, reconstruction, and analyzing disentanglement. We use the permutation-invariant setting, where each 28×28 MNIST image is treated as a 784D vector without spatial structure, which requires to use a MLP instead of a CNN. We follow [7] and use a three layer MLP with a latent space size of 50D. Both networks, E and G have two hidden layers with 1024 units each. In [7] the features used are the activations of the layer before the last of the encoder, which are 1024D vectors. We

Figure 4: **MNIST traversal.** Reconstructions of the interpolations in the $\mathcal{Z}$ space, and the $\mathcal{W}$ space, between the same digits. The latter transition appears to be smoother.

Table 2: **MNIST classification.** Classification accuracy (%) on the permutation-invariant MNIST [32] using 1NN and linear SVM, with *same writers* (SW) and *different writers* (DW) settings, and short features (sf) vs. long features (lf), indicated as sf/lf.

	1NN SW	Linear SVM SW	1NN DW	Linear SVM DW
AE(ℓ_1)	97.15/97.43	88.71/97.27	96.84/96.80	89.78/97.72
AE(ℓ_2)	**97.52**/97.37	88.78/97.23	**97.05**/96.77	89.78/97.72
LR	92.79/97.28	89.74/97.56	91.90/96.69	90.03/97.80
JLR	92.54/97.02	89.23/97.19	91.97/96.45	90.82/97.62
BiGAN [7]	95.83/97.14	90.52/97.59	95.38/96.81	91.34/97.74
ALAE (ours)	93.79/**97.61**	**93.47/98.20**	94.59/**97.47**	**94.23/98.64**

refer to those as *long features*. We also use, as features, the 50D vectors taken from the latent space, $\mathcal{W}$. We refer to those as *short features*.

MNIST has an official split into training and testing sets of sizes 60000 and 10000 respectively. We refer to it as *different writers* (DW) setting since the human writers of the digits for the training set are different from those who wrote the testing digits. We consider also a *same writers* (SW) setting, which uses only the official training split by further splitting it in two parts: a train split of size 50000 and a test split of size 10000, while the official testing split is ignored. In SW the pools of writers in the train and test splits overlap, whereas in DW they do not. This makes SW an easier setting than DW.

Results. We report the accuracy with the 1NN classifier as in [7], and extend those results by reporting also the accuracy with the linear SVM, because it allows a more direct analysis of disentanglement. Indeed, we recall that a disentangled representation [45, 42, 1] refers to a space consisting of linear subspaces, each of which is responsible for one factor of variation. Therefore, a linear classifier based on a disentangled feature space should lead to better performance compared to one working on an entangled space. Table 2 summarizes the average accuracy over five trials for ALAE, BiGAN, as well as the following baselines proposed in [7]: Latent Regressor (LR), Joint Latent Regressor (JLR), Autoencoders trained to minimize the ℓ_2 (AE(ℓ_2)) or the ℓ_1 (AE(ℓ_1)) reconstruction error.

The most significant result of Table 2 is drawn by comparing the 1NN with the corresponding linear SVM columns. Since 1NN does not presume disentanglement in order to be effective, but linear SVM does, larger performance drops signal stronger entanglement. ALAE is the approach that remains more stable when switching from 1NN to linear SVM, suggesting a greater disentanglement of the space. This is true especially for short features, whereas for long features this effect fades away because linear separability grows.

We also note that ALAE does not always provide the best accuracy, and the baseline AE (especially AE(ℓ_2)) does well with 1NN, and more so with short features. This might

Method	FFHQ	LSUN Bedroom
StyleGAN [24]	4.40	2.65
PGGAN [23]	-	8.34
IntroVAE [20]	-	8.84
Pioneer [16]	-	18.39
Balanced Pioneer [17]	-	17.89
StyleALAE Generation	13.09	17.13
StyleALAE Reconstruction	16.52	15.92

Table 3: **FID scores.** FID scores (lower is better) measured on FFHQ [24] and LSUN Bedroom [54].

be explained by the baseline AE learning a representation that is closer to a discriminative one. Other approaches instead focus more on learning representations for drawing synthetic random samples, which are likely richer, but less discriminative. This effect also fades for longer features.

Another observation is about SW vs. DW. 1NN generalizes less effectively for DW, as expected, but linear SVM provides a small improvement. This is unclear, but we speculate that DW might have fewer writers in the test set, and potentially slightly less challenging.

Figure 3 shows qualitative reconstruction results. It can be seen that BiGAN reconstructions are subject to semantic label flipping much more often than ALAE. Finally, Figure 4 shows two traversals: one obtained by interpolating in the $\mathcal{Z}$ space, and the other by interpolating in the $\mathcal{W}$ space. The second shows a smoother image space transition, suggesting a lesser degree of entanglement.

7.2. Learning style representations

FFHQ. We evaluate StyleALAE with the FFHQ [24] dataset. It is very recent and consists of 70000 images of people faces aligned and cropped at resolution of 1024×1024. In contrast to [24], we split FFHQ into a training set of 60000 images and a testing set of 10000 images. We do so in order to measure the reconstruction quality for which we need images that were not used during training.

We implemented our approach with PyTorch. Most of the experiments were conducted on a machine with $4\times$ GPU Titan X, but for training the models at resolution 1024×1024 we used a server with $8\times$ GPU Titan RTX. We trained StyleALAE for 147 epochs, 18 of which were spent at resolution 1024×1024. Starting from resolution 4×4 we grew StyleALAE up to 1024×1024. When growing to a new resolution level we used 500k training samples during the transition, and another 500k samples for training stabilization. Once reached the maximum resolution of 1024×1024, we continued training for 1M images. Thus, the total training time measured in images was 10M. In contrast, the total training time for StyleGAN [24] was 25M images, and 15M of them were used at resolution 1024×1024. At the same resolution we trained StyleALAE with only 1M images, so, 15 times less.

Table 3 reports the FID score [18] for generations and reconstructions. Source images for reconstructions are from the test set and were not used during training. The scores of StyleALAE are higher, and we regard the large training time difference between StyleALAE and StyleGAN (1M vs 15M) as the likely cause of the discrepancy.

Method		Path length full	Path length end
StyleGAN	$\mathcal{Z}$	412.0	415.3
StyleGAN no mixing	$\mathcal{W}$	200.5	160.6
StyleGAN	$\mathcal{W}$	231.5	182.1
StyleALAE	$\mathcal{Z}$	300.5	292.0
StyleALAE	$\mathcal{W}$	**134.5**	**103.4**

Table 4: **PPL.** Perceptual path lengths on FFHQ measured in the $\mathcal{Z}$ and the $\mathcal{W}$ spaces (lower is better).

Table 4 reports the perceptual path length (PPL) [24] of SyleALAE. This is a measurement of the degree of disentanglement of representations. We compute the values for representations in the $\mathcal{W}$ and the $\mathcal{Z}$ space, where StyleALAE is trained with style mixing in both cases. The StyleGAN score measured in $\mathcal{Z}$ corresponds to a traditional network, and in $\mathcal{W}$ for a style-based one. We see that the PPL drops from $\mathcal{Z}$ to $\mathcal{W}$, indicating that $\mathcal{W}$ is perceptually more linear than $\mathcal{Z}$, thus less entangled. Also, note that for our models the PPL is lower, despite the higher FID scores.

Figure 6 shows a random collection of generations obtained from StyleALAE. Figure 5 instead shows a collection of reconstructions. In Figure 9 instead, we repeat the style mixing experiment in [24], but with real images as sources and destinations for style combinations. We note that the original images are faces of celebrities that we downloaded from the internet. Therefore, they are not part of FFHQ, and come from a different distribution. Indeed, FFHQ is made of face images obtained from Flickr.com depicting non-celebrity people. Often the faces do not wear any makeup, neither have the images been altered (e.g., with Photoshop). Moreover, the imaging conditions of the FFHQ acquisitions are very different from typical photoshoot stages, where professional equipment is used. Despite this change of image statistics, we observe that StyleALAE works effectively on both reconstruction and mixing.

LSUN. We evaluated StyleALAE with LSUN Bedroom [54]. Figure 7 shows generations and reconstructions from unseen images during training. Table 3 reports the FID scores on the generations and the reconstructions.

	FID	PPL full
PGGAN [23]	**8.03**	229.2
GLOW [27]	68.93	219.6
PIONEER [16]	39.17	155.2
Balanced PIONEER [17]	25.25	146.2
StyleALAE (ours)	19.21	**33.29**

Table 5: Comparison of FID and PPL scores for CelebA-HQ images at 256×256 (lower is better). FID is based on 50,000 generated samples compared to training samples.

Figure 5: **FFHQ reconstructions.** Reconstructions of unseen images with StyleALAE trained on FFHQ [24] at 1024×1024.

Figure 6: **FFHQ generations.** Generations with StyleALAE trained on FFHQ [24] at 1024×1024.

Figure 7: **LSUN generations and reconstructions.** Generations (first row), and reconstructions using StyleALAE trained on LSUN Bedroom [54] at resolution 256×256.

CelebA-HQ. CelebA-HQ [23] is an improved subset of CelebA [33] consisting of 30000 images at resolution 1024×1024. We follow [16, 17, 27, 23] and use CelebA-HQ downscaled to 256×256 with training/testing split of 27000/3000. Table 5 reports the FID and PPL scores, and Figure 8 compares StyleALE reconstructions of unseen faces with two other approaches.

8. Conclusions

We introduced ALAE, a novel autoencoder architecture that is simple, flexible and general, as we have shown to be efective with two very different backbone generator-encoder networks. Differently from previous work it allows learning the probability distribution of the latent space, when the data distribution is learned in adversarial settings. Our experiments confirm that this enables learning representations that are likely less entangled. This allows us to extend StyleGAN to StyleALAE, the first autoencoder capable of generating and manipulating images in ways not possible with SyleGAN alone, while maintaining the same level of visual detail.

Figure 8: **CelebA-HQ reconstructions.** CelebA-HQ reconstructions of unseen samples at resolution 256×256. Top row: real images. Second row: StyleALAE. Third row: Balanced PIONEER [17]. Last row: PIONEER [16]. StyleALAE reconstructions look sharper and less distorted.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grants No. OIA-1920920, and OAC-1761792.

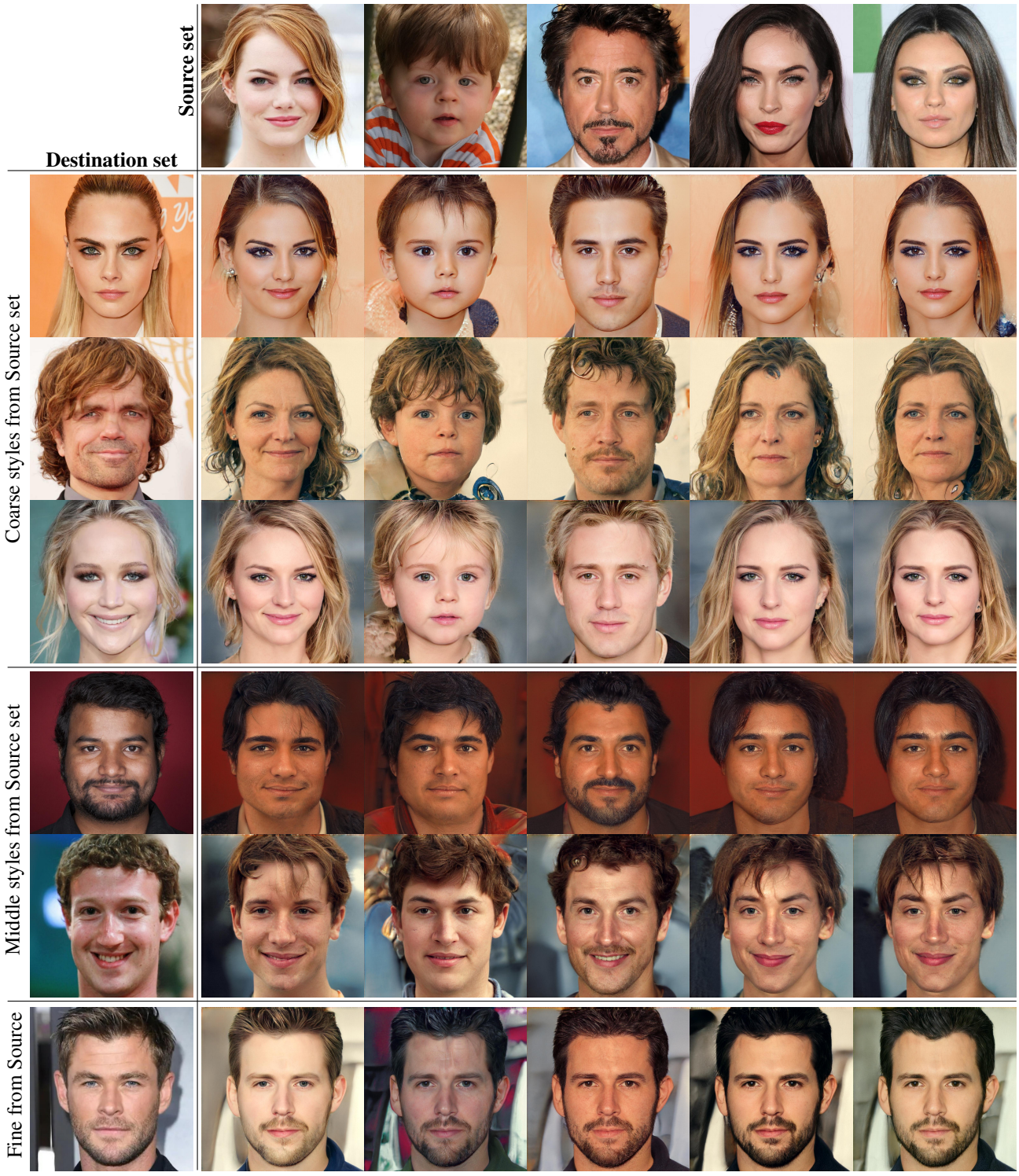

Figure 9: Two sets of real images were picked to form the Source set and the Destination set. The rest of the images were generated by copying specified subset of styles from the Source set into the Destination set. This experiment repeats the one from [24], but with real images. Copying the coarse styles brings high-level aspects such as pose, general hair style, and face shape from Source set, while all colors (eyes, hair, lighting) and finer facial features resemble the Destination set. Instead, if we copy middle styles from the Source set, we inherit smaller scale facial features like hair style, eyes open/closed from Source, while the pose, and general face shape from Destination are preserved. Finally, copying the fine styles from the Source set brings mainly the color scheme and microstructure.

References

[1] Alessandro Achille and Stefano Soatto. Emergence of invariance and disentanglement in deep representations. *The Journal of Machine Learning Research*, 19(1):1947–1980, 2018. 1, 2, 5

[2] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein GAN. In *arXiv:1701.07875*, 2017. 1, 2

[3] A. Brock, J. Donahue, and K. Simonyan. Large scale GAN training for high fidelity natural image synthesis. In *ICLR*, 2019. 1, 2

[4] R. T. Q. Chen, X. Li, R. Grosse, and R. Duvenaud. Isolating sources of disentanglement in variational autoencoders. In *NeurIPS*, 2018. 1, 2

[5] Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *Advances in neural information processing systems (NIPS)*, pages 1486–1494, 2015. 2

[6] J. Donahue, P. Krähenbühl, and T. Darrell. Adversarial feature learning. In *ICLR*, 2016. 1, 2, 3, 4

[7] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *arXiv preprint arXiv:1605.09782*, 2016. 5

[8] Harris Drucker and Yann Le Cun. Improving generalization performance using double backpropagation. *IEEE Transactions on Neural Networks*, 3(6):991–997, 1992. 4

[9] V. Dumoulin, I. Belghazi, B. Poole, O. Mastropietro, A. Lamb, M. Arjovsky, and A. Courville. Adversarially learned inference. In *ICLR*, 2016. 1, 2, 3, 4

[10] Cian Eastwood and Christopher KI Williams. A framework for the quantitative evaluation of disentangled representations. In *ICLR*, 2018. 1, 2

[11] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323, 2011. 4

[12] Ian Goodfellow. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016. 2

[13] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2672–2680, 2014. 1, 2, 4

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1

[15] Ari Heljakka, Arno Solin, and Juho Kannala. Pioneer networks: Progressively growing generative autoencoder. In *Asian Conference on Computer Vision (ACCV)*, pages 22–38. Springer, 2018. 2

[16] Ari Heljakka, Arno Solin, and Juho Kannala. Pioneer networks: Progressively growing generative autoencoder. In *Asian Conference on Computer Vision*, pages 22–38. Springer, 2018. 6, 7

[17] Ari Heljakka, Arno Solin, and Juho Kannala. Towards photographic image manipulation with balanced growing of generative autoencoders. *arXiv preprint arXiv:1904.06145*, 2019. 6, 7

[18] 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. In *Advances in Neural Information Processing Systems*, pages 6626–6637, 2017. 6

[19] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2017. 1, 2

[20] H. Huang, Z. Li, R. He, Z. Sun, and T. Tan. Introvae: Introspective variational autoencoders for photographic image synthesis. In *NIPS*, 2018. 1, 2, 3, 6

[21] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1501–1510, 2017. 4

[22] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 172–189, 2018. 4

[23] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017. 1, 2, 4, 5, 6, 7

[24] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1, 2, 4, 6, 7, 8

[25] H. Kim and A. Mnih. Disentangling by factorising. In *ICML*, 2018. 1, 2

[26] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[27] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*, pages 10215–10224, 2018. 2, 6, 7

[28] D. P. Kingma and W. Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations (ICLR)*, 2014. 1, 2, 3

[29] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *International Conference on Neural Information Processing Systems (NIPS)*, pages 1097–1105, 2012. 1

[30] Karol Kurach, Mario Lucic, Xiaohua Zhai, Marcin Michalski, and Sylvain Gelly. The gan landscape: Losses, architectures, regularization, and normalization. In *arXiv:1807.04720*, 2018. 1

[31] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther. Autoencoding beyond pixels using a learned similarity metric. In *International Conference on Machine Learning (ICML)*, pages 1558–1566, 2016. 1, 2, 3, 4

[32] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 5

[33] 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*, pages 3730–3738, 2015. 1, 7

[34] Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet. Are gans created equal? a large-scale study. In *Advances in neural information processing systems (NeurIPS)*, pages 700–709, 2018. 1

[35] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey. Adversarial autoencoders. In *arXiv:1511.05644*, 2015. 1, 2, 3

[36] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? *arXiv:1801.04406*, 2018. 1, 2, 4, 5

[37] M. Mirza and S. Osindero. Conditional generative adversarial nets. In *arXiv:1411.1784*, 2014. 1

[38] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *arXiv:1802.05957*, 2018. 1, 2

[39] J. Z. Nagarajan, V.and Kolter. Gradient descent GAN optimization is locally stable. In *International Conference on Neural Information Processing Systems (NIPS)*, pages 5591–5600, 2017. 2

[40] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems (NIPS)*, pages 91–99, 2015. 1

[41] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation andapproximate inference in deep generative models. In *International Conference on Machine Learning (ICML)*, 2014. 1, 2, 3

[42] Karl Ridgeway. A survey of inductive biases for factorial representation-learning. *arXiv preprint arXiv:1612.05299*, 2016. 5

[43] Andrew Slavin Ross and Finale Doshi-Velez. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In *Thirty-second AAAI conference on artificial intelligence*, 2018. 4

[44] Kevin Roth, Aurelien Lucchi, Sebastian Nowozin, and Thomas Hofmann. Stabilizing training of generative adversarial networks through regularization. In *Advances in neural information processing systems*, pages 2018–2028, 2017. 2, 5

[45] Jürgen Schmidhuber. Learning factorial codes by predictability minimization. *Neural Computation*, 4(6):863–879, 1992. 5

[46] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, abs/1409.1556, 2015. 1

[47] A. Srivastava, L. Valkov, C. Russell, M. U. Gutmann, and C. Sutton. Veegan: Reducing mode collapse in gans using implicit variational learning. In *NIPS*, 2017. 2, 3

[48] Jakub M Tomczak and Max Welling. VAE with a VampPrior. In *AISTATS*, 2018. 2, 3

[49] D. Ulyanov, A. Vedaldi, and V. Lempitsky. It takes (only) two: Adversarial generator-encoder networks. In *AAAI*, 2018. 2, 3

[50] Aaron Van Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. In *International Conference on Machine Learning*, pages 1747–1756, 2016. 2

[51] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8798–8807, 2018. 2

[52] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module. In *The European Conference on Computer Vision (ECCV)*, September 2018. 1

[53] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao. LSUN: construction of a large-scale image dataset using deep learning with humans in the loop. In *arXiv:1506.03365*, 2015. 1

[54] 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. 6, 7

[55] 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 *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 5907–5915, 2017. 2

[56] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan++: Realistic image synthesis with stacked generative adversarial networks. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1947–1962, 2018. 2

[57] Zizhao Zhang, Yuanpu Xie, and Lin Yang. Photographic text-to-image synthesis with a hierarchically-nested adversarial network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6199–6208, 2018. 2

[58] Jun Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. *IEEE International Conference on Computer Vision (ICCV)*, 2017-Octob:2242–2251, 2017. 1