

Photographic Text-to-Image Synthesis with a Hierarchically-nested Adversarial Network

Zizhao Zhang*, Yuanpu Xie*, Lin Yang

University of Florida

*equal contribution

Abstract

This paper presents a novel method to deal with the challenging task of generating photographic images conditioned on semantic image descriptions. Our method introduces accompanying hierarchical-nested adversarial objectives inside the network hierarchies, which regularize mid-level representations and assist generator training to capture the complex image statistics. We present an extensible single-stream generator architecture to better adapt the jointed discriminators and push generated images up to high resolutions. We adopt a multi-purpose adversarial loss to encourage more effective image and text information usage in order to improve the semantic consistency and image fidelity simultaneously. Furthermore, we introduce a new visual-semantic similarity measure to evaluate the semantic consistency of generated images. With extensive experimental validation on three public datasets, our method significantly improves previous state of the arts on all datasets over different evaluation metrics.

1. Introduction

Photographic text-to-image synthesis is a significant problem in generative model research [34], which aims to learn a mapping from a semantic text space to a complex RGB image space. This task requires the generated images to be not only realistic but also *semantically consistent*, i.e., the generated images should preserve specific object sketches and semantic details described in text.

Recently, Generative adversarial networks (GANs) have become the main solution to this task. Reed *et al.* [34] address this task through a GAN based framework. But this method only generates 64^2 images and can barely generate vivid object details. Based on this method, StackGAN [47] proposes to stack another low-to-high resolution GAN to generate 256^2 images. But this method requires training two separate GANs. Later on, [8] proposes to bypass the difficulty of learning mappings from text to RGB images

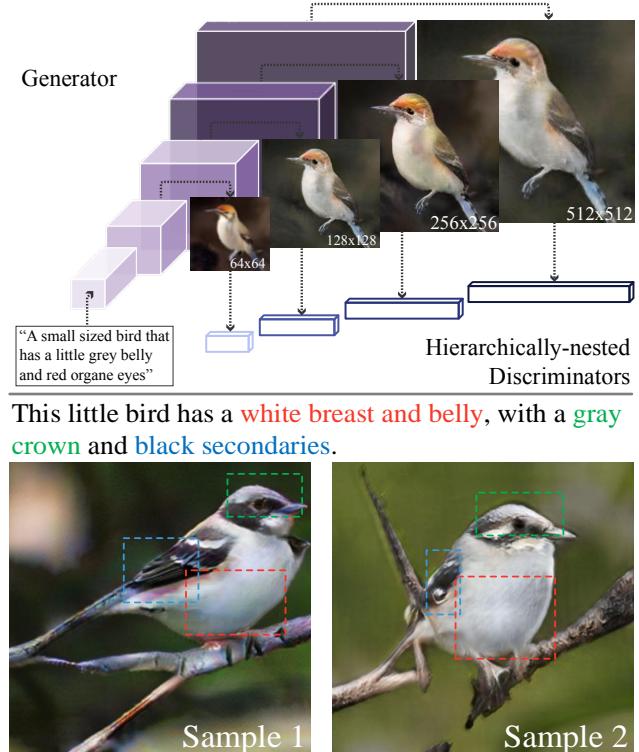


Figure 1: Top: Overview of our hierarchically-nested adversarial network, which produces side output images with growing resolutions. Each side output is associated with a discriminator. Bottom: Two test sample results where fine-grained details are highlighted.

and treat it as a pixel-to-pixel translation problem [15]. It works by re-rendering an arbitrary-style 128^2 training image conditioned on a targeting description. However, its high-resolution synthesis capability is unclear. At present, training a generative model to map from a low-dimensional text space to a high-resolution image space in a fully end-to-end manner still remains unsolved.

This paper pays attention to two major difficulties for text-to-image synthesis with GANs. The first is balancing the convergence between generators and discriminators

[11, 37], which is a common problem in GANs. The second is stably modeling the huge pixel space in high-resolution images and guaranteeing semantic consistency [47]. An effective strategy to regularize generators is critical to stabilize the training and help capture the complex image statistics [13].

In this paper, we propose a novel end-to-end method that can directly model high-resolution image statistics and generate photographic images (see Figure 1 bottom). The contributions are described as follows.

Our generator resembles a simple vanilla GAN, without requiring multi-stage training and multiple internal text conditioning like [47] or additional class label supervision like [6]. To tackle the problem of the big leap from the text space to the image space, our insight is to leverage and regularize hierarchical representations with additional ‘deep’ adversarial constraints (see Figure 1 top). We introduce accompanying hierarchically-nested discriminators at multi-scale intermediate layers to play adversarial games and jointly encourage the generator to approach the real training data distribution. We also propose a new convolutional neural network (CNN) design for the generator to support accompanying discriminators more effectively. To guarantee the image diversity and semantic consistency, we enforce discriminators at multiple side outputs of the generator to simultaneously differentiate real-and-fake image-text pairs as well as real-and-fake local image patches.

We validate our proposed method on three datasets, CUB birds [41], Oxford-102 flowers [30], and large-scale MSCOCO [24]. In complement of existing evaluation metrics (e.g. Inception score [37]) for generative models, we also introduce a new visual-semantic similarity metric to evaluate the alignment between generated images and conditioned text. It alleviates the issue of the expensive human evaluation. Extensive experimental results and analysis demonstrate the effectiveness of our method and significantly improved performance compared against previous state of the arts on all three evaluation metrics. All source code will be released.

2. Related Work

Deep generative models have attracted wide interests recently, including GANs [11, 33], Variational Auto-encoders (VAE) [19], etc [32]. There are substantial existing methods investigating the better usage of GANs for different applications, such as image synthesis [33, 38], (unpaired) pixel-to-pixel translation [15, 52], medical applications [5, 50], etc [22, 45, 13, 46].

Text-to-image synthesis is an interesting application of GANs. Reed *et al.* [34] is the first to introduce a method that can generate 64^2 resolution images. This method also presents a new strategy for image-text matching aware adversarial training. Reed *et al.* [35] propose a generative

adversarial what-where network (GAWWN) to enable location and content instructions in text-to-image synthesis. Zhang *et al.* [47] propose a two-stage training strategy that is able to generate 256^2 compelling images. Recently, Dong *et al.* [8] propose to learn a joint embedding of images and text so as to re-render a prototype image conditioned on a targeting description. Cha *et al.* [3] explore the usage of the perceptual loss [16] with a CNN pretrained on ImageNet and Dash *et al.* [6] make use of auxiliary classifiers (similar with [31]) to assist GAN training for text-to-image synthesis. Xu *et al.* [43] shows an attention-driven method to improve fine-grained details.

Learning a continuous mapping from a low-dimensional manifold to a complex real image distribution is a long-standing problem. Although GANs have made significant progress, there are still many unsolved difficulties, e.g., training instability and high-resolution generation. Wide methods have been proposed to address the training instability, such as various training techniques [36, 2, 38, 31], regularization using extra knowledge (e.g. image labels, ImageNet CNNs) [9, 22, 6], or different generator and discriminator combinations [28, 10, 13]. *While our method shows a new way to unite generators and discriminators and does not require any extra knowledge apart from training paired text and images.* In addition, it is easy to see the training difficulty increases significantly as the targeting image resolution increases.

To synthesize high-resolution images, cascade networks are effective to decompose originally difficult tasks to multiple subtasks (Figure 2 A). Denton *et al.* [7] train a cascade of GANs in a Laplacian pyramid framework (LAPGAN) and use each to synthesize and refine image details and push up the output resolution stage-by-stage. StackGAN also shares similar ideas with LAPGAN. Inspired by this strategy, Chen *et al.* [4] present a cascaded refinement network to synthesize high-resolution scenes from semantic maps. Recently, Karras *et al.* [17] propose a progressive training of GANs. The idea is to add symmetric generator and discriminator layers gradually for high-resolution image generation (Figure 2 C). *Compared with these strategies that train low-to-high resolution GANs stage-by-stage or progressively, our method has the advantage of leveraging mid-level representations to encourage the integration of multiple subtasks, which makes end-to-end high-resolution image synthesis in a single vanilla-like GAN possible.*

Leveraging hierarchical representations of CNNs is an effective way to enhance implicit multi-scaling and ensembling for tasks such as image recognition [23, 49] and pixel or object classification [42, 26, 51]. Particularly, using deep supervision [23] at intermediate convolutional layers provides short error paths and increases the discriminativeness of feature representations. *Our hierarchically-nested adversarial objective is inspired by the family of deeply-*

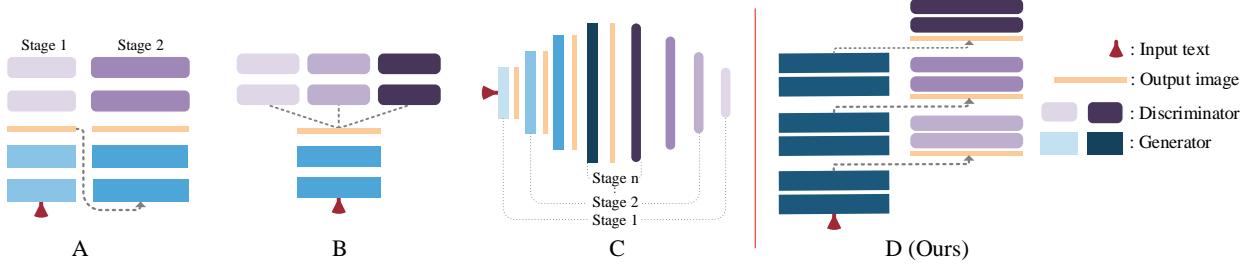


Figure 2: Overviews of some typical GAN frameworks. **A** uses multi-stage GANs [47, 7]. **B** uses multiple discriminators with one generator [10, 29]. **C** progressively trains symmetric discriminators and generators [17, 13]. **A** and **C** can be viewed as decomposing high-resolution tasks to multi-stage low-to-high resolution tasks. **D** is our proposed framework that uses a single-stream generator with hierarchically-nested discriminators trained end-to-end.

supervised CNNs.

3. Method

3.1. Adversarial objective basics

In brief, a GAN [11] consists of a generator G and a discriminator D , which are alternatively trained to compete with each other. D is optimized to distinguish synthesized images from real images, meanwhile, G is trained to fool D by synthesizing fake images. Concretely, the optimal G and D can be obtained by playing the following two-player min-max game,

$$G^*, D^* = \arg \min_G \max_D \mathcal{V}(D, G, Y, z), \quad (1)$$

where Y and $z \sim \mathcal{N}(0, 1)$ denote training images and random noises, respectively. \mathcal{V} is the overall GAN objective, which usually takes the form $\mathbb{E}_{Y \sim p_{data}} [\log D(Y)] + \mathbb{E}_{z \sim p_z} [\log(1 - D(G(z)))]$ (the cross-entropy loss) or other variations [27, 2].

3.2. Hierarchical-nested adversarial objectives

Numerous GAN methods have demonstrated ways to unite generators and discriminators for image synthesis. Figure 2 and Section 2 discuss some typical frameworks. Our method actually explores a new dimension of playing this adversarial game along the depth of a generator (Figure 2 D), which integrates additional hierarchical-nested discriminators at intermediate layers of the generator. The proposed objectives act as regularizers to the hidden space of G , which also offer a short path for error signal flows and help reduce the training instability.

The proposed \mathcal{G} is a CNN (defined in Section 3.4), which produces multiple side outputs:

$$X_1, \dots, X_s = \mathcal{G}(t, z), \quad (2)$$

where $t \sim p_{data}$ denotes a sentence embedding (generated by a pre-trained char-RNN text encoder [34]). $\{X_1, \dots, X_{s-1}\}$ are images with gradually growing resolutions and X_s is the final output with the highest resolution.

For each side output X_i from the generator, a distinct discriminator D_i is used to compete with it. Therefore, our full min-max objective is defined as

$$\mathcal{G}^*, \mathcal{D}^* = \arg \min_{\mathcal{G}} \max_{\mathcal{D}} \mathcal{V}(\mathcal{G}, \mathcal{D}, \mathcal{Y}, t, z), \quad (3)$$

where $\mathcal{D} = \{D_1, \dots, D_s\}$ and $\mathcal{Y} = \{Y_1, \dots, Y_s\}$ denotes training images at multi-scales, $\{1, \dots, s\}$. Compared with Eq. (1), our generator competes with multiple discriminators $\{D_i\}$ at different hierarchies (Figure 2 D), which jointly learn discriminative features in different contextual scales.

In principle, the lower-resolution side output is used to learn semantic consistent image structures (e.g. object sketches, colors, and background), and the subsequent higher-resolution side outputs are used to render fine-grained details. Since our method is trained in an end-to-end fashion, the lower-resolution outputs can also fully utilize top-down knowledge from discriminators at higher resolutions. As a result, we can observe consistent image structures, color and styles in the outputs of both low and high resolution images. The experiment demonstrates this advantage compared with StackGAN.

3.3. Multi-purpose adversarial losses

Our generator produces resolution-growing side outputs composing an image pyramid. We leverage this hierarchy property and allow adversarial losses to capture hierarchical image statistics, with a goal to guarantee both semantic consistency and image fidelity.

In order to guarantee semantic consistency, we adopt the matching-aware pair loss proposed by [34]. The discriminator is designed to take image-text pairs as inputs and be trained to identify two types of errors: a real image with mismatched text and a fake image with conditioned text.

The pair loss is designed to guarantee the global semantic consistency. However, there is no explicit loss to guide the discriminator to differentiate real images from fake images. And combining both tasks (generating realistic images and matching image styles with text) into one network output complicates the already challenging learning tasks.

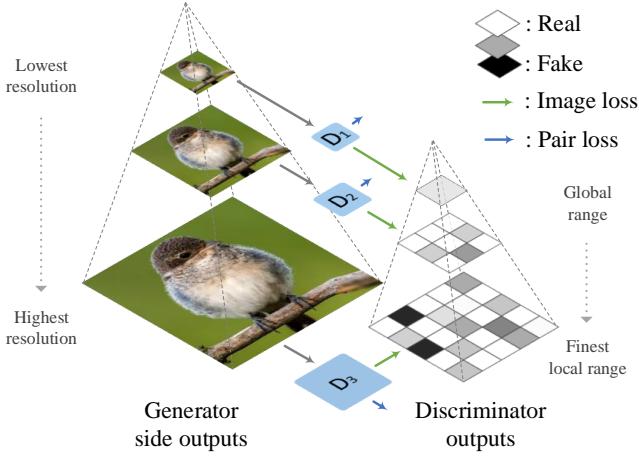


Figure 3: For each side out image in the pyramid from the generator, the corresponding discriminator D_i computes the matching-aware pair loss and the local image loss (outputting a $R_i \times R_i$ probability map to classify real or fake image patches).

Moreover, as the image resolution goes higher, it might be challenging for a global pair-loss discriminator to capture the local fine-grained details (results are validated in experiments). In addition, as pointed in [38], a single global discriminator may over-emphasize certain biased local features and lead to artifacts.

To alleviate these issues and guarantee image fidelity, our solution is to add local adversarial image losses. We expect the low-resolution discriminators to focus on global structures, while the high-resolution discriminators to focus on local image details. Each discriminator D_i consists of two branches (see Section 3.4), one computes a single scalar value for the pair loss and another branch computes a $R_i \times R_i$ 2D probability map O_i for the local image loss. For each D_i , we control R_i accordingly to tune the receptive field of each element in O_i , which differentiates whether a corresponding local image patch is real or fake. The local GAN loss is also used for pixel-to-pixel translation tasks [38, 52, 15]. Figure 3 illustrates how hierarchically-nested discriminators compute the two losses on the generated images in the pyramid.

Full Objective Overall, our full min-max adversarial objective can be defined as

$$\begin{aligned} \mathcal{V}(\mathcal{G}, \mathcal{D}, \mathcal{Y}, \mathbf{t}, \mathbf{z}) = \sum_{i=1}^s & \left(L_2(D_i(Y_i)) + L_2(D_i(Y_i, \mathbf{t}_Y)) + \right. \\ & \left. \overline{L}_2(D_i(X_i)) + \overline{L}_2(D_i(X_i, \mathbf{t}_{X_i})) + \overline{L}_2(D_i(Y_i, \mathbf{t}_{\overline{Y}})) \right), \end{aligned} \quad (4)$$

where $L_2(x) = \mathbb{E}[(x - \mathbb{I})^2]$ is the mean-square loss (instead of the conventional cross-entropy loss) and $\overline{L}_2(x) = \mathbb{E}[x^2]$. This objective is minimized by \mathcal{D} . While in practice, \mathcal{G} minimizes $\sum_{i=1}^s (L_2(D_i(\mathcal{G}(\mathbf{t}, \mathbf{z}), i)) + L_2(D_i(\mathcal{G}(\mathbf{t}, \mathbf{z}), i, \mathbf{t}_{X_i})))$

instead. For the local image loss, the shape of $x, \mathbb{I} \in \mathbb{R}^{R_i \times R_i}$ varies accordingly (see Figure 3). $R_i = 1$ refers to the (largest local) global range. $D_i(X_i)$ is the image loss branch and $D_i(X_i, \mathbf{t}_{X_i})$ is the pair loss branch (conditioned on \mathbf{t}_{X_i}). $\{Y_i, \mathbf{t}_Y\}$ denotes a matched image-text pair and $\{Y_i, \mathbf{t}_{\overline{Y}}\}$ denotes a mismatched image-text pair.

In the spirit of variational auto-encoder [20] and the practice of StackGAN [47] (termed conditioning augmentation (CA)), instead of directly using the deterministic text embedding, we sample a stochastic vector from a Gaussian distribution $\mathcal{N}(\mu(\mathbf{t}), \Sigma(\mathbf{t}))$, where μ and Σ are parameterized functions of \mathbf{t} . We add the Kullback-Leibler divergence regularization term, $D_{KL}(\mathcal{N}(\mu(\mathbf{t}), \Sigma(\mathbf{t})) || \mathcal{N}(0, \mathbf{I}))$, to the GAN objective to prevent over-fitting and force smooth sampling over the text embedding distribution.

3.4. Architecture Design

Generator The generator is simply composed of three kinds of modules, termed K -repeat res-blocks, stretching layers, and linear compression layers. A single res-block in the K -repeat res-block is a modified¹ residual block [12], which contains two convolutional (conv) layers (with batch normalization (BN) [14] and ReLU). The stretching layer serves to change the feature map size and dimension. It simply contains a scale-2 nearest up-sampling layer followed by a conv layer with BN+ReLU. The linear compression layer is a single conv layer followed by a Tanh to directly compress feature maps to the RGB space. We prevent any non-linear function in the compression layer that could impede the gradient signals. Starting from a $1024 \times 4 \times 4$ embedding, which is computed by CA and a trained embedding matrix, the generator simply uses M K -repeat res-blocks connected by $M-1$ in-between stretching layers until the feature maps reach to the targeting resolution. For example, for 256×256 resolution and $K=1$, there are $M=6$ 1-repeat res-blocks and 5 stretching layers. At pre-defined side-output positions at scales $\{1, \dots, s\}$, we apply the compression layer to generate side output images, for the inputs of discriminators.

Discriminator The discriminator simply contains consecutive stride-2 conv layers with BN+LeakyReLU. There are two branches added to the upper layer of the discriminator. One branch is a direct fully convolutional layers to produce a $R_i \times R_i$ probability map (see Figure 3) and classify each location as real or fake. Another branch first concatenates a $512 \times 4 \times 4$ feature map and a $128 \times 4 \times 4$ text embedding (replicated by a reduced 128-d text embedding). Then we use an 1×1 conv to fuse text and image features and a 4×4 conv layer to classify an image-text pair to real or fake.

The optimization is similar to the standard alternative

¹We remove ReLU after the skip-addition of each residual block, with an intention to reduce sparse gradients.

training strategy in GANs. Please refer to the supplementary material for more training and network details.

4. Experiments

We denote our method as **HDGAN**, referring to High-Definition results and the idea of Hierarchically-nested Discriminators.

Dataset We evaluate our model on three widely used datasets. The CUB dataset [41] contains 11,788 bird images belonging to 200 categories. The Oxford-102 dataset [30] contains 8,189 flow images in 102 categories. Each image in both datasets is annotated with 10 descriptions provided by [34]. We pre-process and split the images of CUB and Oxford-102 following the same pipeline in [34, 47]. The COCO dataset [24] contains 82,783 training images and 40,504 validation images. Each image has 5 text annotations. We use the pre-trained char-RNN text encoder provided by [34] to encode each sentence into a 1024-d text embedding vector.

Evaluation metric We use three kinds of quantitative metrics to evaluate our method. 1) Inception score [37] is a measurement of both objectiveness and diversity of generated images. Evaluating this score needs a pre-trained Inception model [39] on ImageNet. For CUB and Oxford-102, we use the fine-tuned Inception models on the training sets of the two datasets, respectively, provided by StackGAN. 2) Multi-scale structural similarity (MS-SSIM) metric [37] is used for further validation. It tests pair-wise similarity of generated images and can identity mode collapses reliably [31]. Lower score indicates higher diversity of generated images (i.e. less model collapses).

3) **Visual-semantic similarity** The aforementioned metrics are widely used for evaluating standard GANs. However, they can not measure the alignment between generated images and the conditioned text, i.e., semantic consistency. [47] resorts to human evaluation, but this procedure is expensive and difficult to conduct. To tackle this issue, we introduce a new measurement inspired by [21], namely visual-semantic similarity (VS similarity). The insight is to train a visual-semantic embedding model and use it to measure the distance between synthesized images and input text. Denote \mathbf{v} as an image feature vector extracted by an Inception model f_{cnn} . We define a scoring function $c(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x} \cdot \mathbf{y}}{\|\mathbf{x}\|_2 \cdot \|\mathbf{y}\|_2}$. Then, we train two mapping functions f_v and f_t , which map both real images and paired text embeddings into a common space in \mathbb{R}^{512} , by minimizing the following bi-directional ranking loss:

$$\begin{aligned} & \sum_{\mathbf{v}} \sum_{\mathbf{t}_{\bar{v}}} \max(0, \delta - c(f_v(\mathbf{v}), f_t(\mathbf{t}_v)) + c(f_v(\mathbf{v}), f_t(\mathbf{t}_{\bar{v}}))) + \\ & \sum_{\mathbf{t}} \sum_{\mathbf{v}_{\bar{t}}} \max(0, \delta - c(f_t(\mathbf{t}), f_v(\mathbf{v}_v)) + c(f_t(\mathbf{t}), f_v(\mathbf{v}_{\bar{t}}))), \end{aligned} \quad (5)$$

Method	Dataset		
	CUB	Oxford-102	COCO
GAN-INT-CLS	2.88±.04	2.66±.03	7.88±.07
GAWWN	3.60±.07	-	-
StackGAN	3.70±.04	3.20±.01	8.45±.03*
StackGAN++	3.84±.06	-	-
TAC-GAN	-	3.45±.05	-
HDGAN	4.15±.05	3.45±.07	11.86±.18

*Recently, it updated to 10.62±.19 in its source code.

Table 1: The Inception-score comparison on the three datasets. HDGAN outperforms others significantly.

Method	Dataset		
	CUB	Oxford-102	COCO
Ground Truth	.302±.151	.336±.138	.426±.157
StackGAN	.228±.162	.278±.134	-
HDGAN	.246±.157	.296±.131	.199±.183

Table 2: The VS similarity evaluation on the three datasets. The higher score represents higher semantic consistency between the generated images and conditioned text. The groundtruth score is shown in the first row.

where δ is the margin, which is set as 0.2. $\{\mathbf{v}, \mathbf{t}\}$ is a ground truth image-text pair, and $\{\mathbf{v}, \mathbf{t}_{\bar{v}}\}$ and $\{\mathbf{v}_{\bar{t}}, \mathbf{t}\}$ denote mismatched image-text pairs. In the testing stage, given an text embedding \mathbf{t} , and the generated image \mathbf{x} , the VS score can be calculated as $c(f_{cnn}(\mathbf{x}), \mathbf{t})$. Higher score indicates better semantic consistency.

4.1. Comparative Results

To validate our proposed HDGAN, we compare our results with GAN-INT-CLS [34], GAWWN [35], TAC-GAN [6], Progressive GAN [17], StackGAN [47] and also its improved version StackGAN++ [48]². We especially compare with StackGAN in details (results are obtained from its provided models).

Table 1 compares the Inception score. We follow the experiment settings of StackGAN to sample ~30,000 256² images for computing the score. HDGAN achieves significant improvement compared against other methods. For example, it improves StackGAN by .45 and StackGAN++ by .31 on CUB. HDGAN achieves competitive results with TAC-GAN on Oxford-102. TAC-GAN uses image labels to increase the discriminability of generators, while we do not use any extra knowledge. Figure 4 and Figure 5 compare the qualitative results with StackGAN on CUB and Oxford-102, respectively, by demonstrating more, semantic details, natural color, and complex object structures. Moreover, we qualitatively compare the diversity of samples conditioned on the same text (with random input noises) in Figure 7 left.

²StackGAN++ and Prog.GAN are two very recently released preprints we noticed. We acknowledge them as they also target at generating high-resolution images.



Figure 4: Generated images on CUB compared with StackGAN. Each sample shows the input text and generated 64^2 (left) and 256^2 (right) images. Our results have significantly higher quality and preserve more semantic details, for example, “the brown and white feathers and read head” in the last column is much better reflected in our images. Moreover, we observed our birds exhibit nicer poses (e.g. the frontal/back views in the second/forth columns). Zoom-in for better observation.



Figure 5: Generated images on Oxford-102 compared with StackGAN. Our generated images perform more natural satisfiability and higher contrast and can generate complex flower structures (e.g. spiked petals in the third example).

HDGAN can generate substantially more compelling samples.

Different from CUB and Oxford-102, COCO is a much more challenging dataset and contains largely diverse natural scenes. Our method significantly outperforms StackGAN as well (Table 1). Figure 15 also shows some generated samples in several different scenes. Please refer to the supplementary material for more results.

Furthermore, the right figure compares the multi-resolution Inception score on CUB. Our results are from the side outputs of a single model. As can be observed, our 64^2 results outperform the 128^2 results of StackGAN and our 128^2 results also outperform 256^2 results of StackGAN substantially. It demonstrates that our HDGAN better preserves semantically consistent information

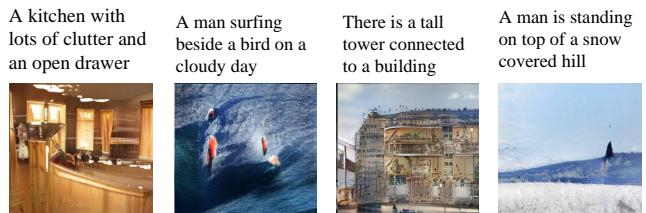
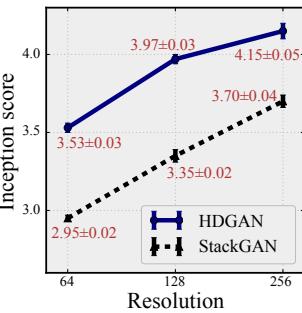


Figure 6: Samples on the COCO validation set, which contain descriptions across different scenes.

mation in all resolutions (as stated in Section 3.2). Figure 7 right validates this property qualitatively. On the other hand, we observe that, in StackGAN, the low-resolution images and high-resolution images sometimes are visually inconsistent (see examples in Figure 4 and 5).

Table 2 compares the proposed visual-semantic similarity (VS) results on three datasets. The scores of the groundtruth image-text pair are also shown for reference. HDGAN achieves consistently better performance on both CUB and Oxford-102. These results demonstrate that

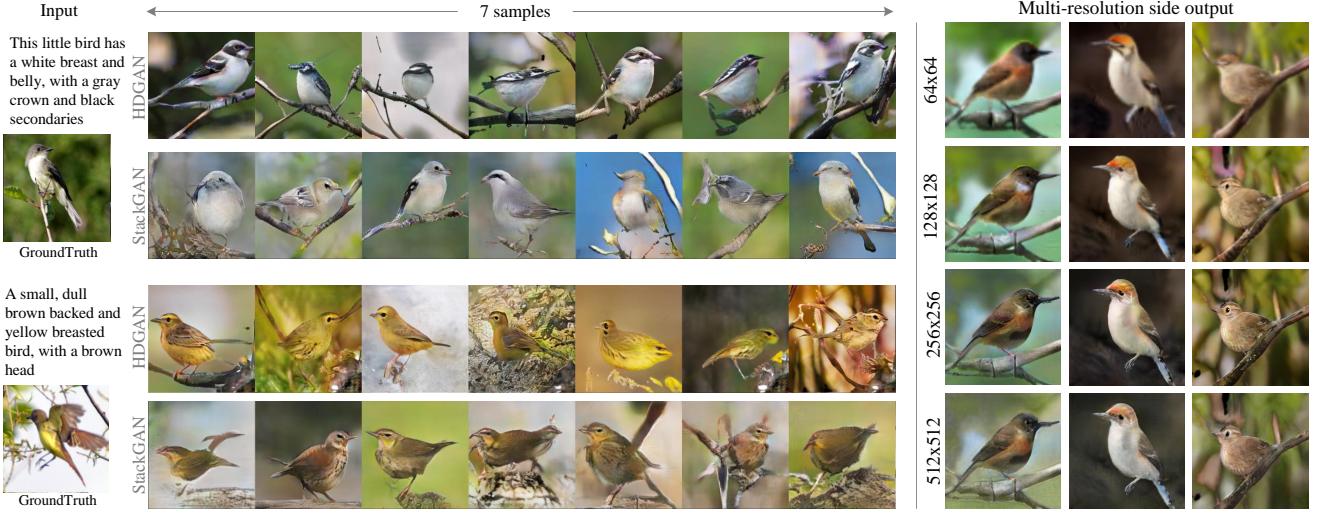


Figure 7: Left: Multiple samples are shown given a single input text. The proposed HDGAN (top) show obviously more fine-grained details. Right: Side outputs of HDGAN with increasing resolutions. Different resolutions are semantically consistent and semantic details appear as the resolution increases.

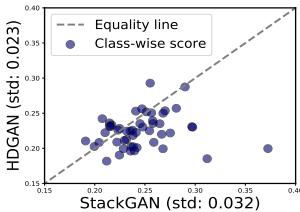


Table 3: Left: Class-wise MS-SSIM evaluation. Lower score indicates higher intraclass dissimilarity. The points below the equality line represent classes our HDGAN wins. The inter-class std is shown in axis text. Right: Overall (not class-wised) MS-SSIM evaluation.

HDGAN can better capture the visual semantic information in generated images.

Table 3 compares the MS-SSIM score with StackGAN and Prog.GAN for bird image generation. StackGAN and our HDGAN use text as input so the generated images are separable in class. We randomly sample $\sim 20,000$ image pairs (400 per class) and compare the class-wise score in the left figure. HDGAN outperforms StackGAN in majority of classes and also has a lower standard deviation (.023 vs. .032). Note that Prog.GAN uses a noise input rather than text. We can compare with it for a general measure of the image diversity. Following the procedure of Prog.GAN, we randomly sample $\sim 10,000$ image pairs from all generated samples³ and show the results in Table 3 right. HDGAN outperforms both methods.

³We use 256^2 bird images provided by Prog.GAN at https://github.com/tkarras/progressive_growing_of_gans. Note that Prog.GAN is trained on the LSUN [44] bird set, which contains ~ 2 million bird images.

4.2. Style Transfer Using Sentence Interpolation

Ideally, a well-trained model can generalize to a smooth linear latent data manifold. To demonstrate this capability, we generate images using the linearly interpolated embeddings between two source sentences. As shown in Figure 8, our generated images show smooth style transformation and well reflect the semantic details in sentences. For example, in the second row, complicated sentences with detailed appearance descriptions (e.g. pointy peaks and black wings) are used, our model could still successfully capture these subtle features and tune the bird’s appearance smoothly.

4.3. Ablation Study and Discussion

Hierarchically-nested adversarial training Our hierarchically-nested discriminators play a role of regularizing the layer representations (at scale $\{64, 128, 256\}$). In Table 4, we demonstrate their effectiveness and show the performance by removing parts of discriminators on both CUB and COCO datasets. As can be seen, increasing the usage of discriminators at different scales have positive effects. And using discriminators at 64^2 is critical (by comparing the 64-256 and 128-256 cases). For now, it is uncertain if adding more discriminators and even on lower resolutions would be helpful. Further validation will be conducted. StackGAN emphasizes the importance of using text embeddings not only at the input but also with intermediate features of the generator, by showing a large drop from 3.7 to 3.45 without doing so. While our method only uses text embeddings at the input. Our results strongly demonstrate the effectiveness of our hierarchically-nested adversarial training to maintain such semantic information and a high Inception score.



Figure 8: Text embedding interpolation from the source to target sentence results in smooth image style changes to match the targeting sentence.

Discriminators			Inception score	
64	128	256	CUB	COCO
		✓	3.52±.04	-
✓	✓		3.99±.04	-
✓		✓	4.14±.03	11.29±.18
✓	✓	✓	4.15±.05	11.86±.18

Table 4: Ablation study of hierarchically-nested adversarial discriminators on CUB and COCO. ✓ indicates whether a discriminator at a certain scale is used. See text for detailed explanations.

The local image loss We analyze the effectiveness of the proposed local adversarial image loss. Table 4 compares the case without using it (denoted as ‘w/o local image loss’). The local image loss helps improve the visual-semantic matching evidenced by a higher VS score. We hypothesize that it is because adding the separate local image loss can offer the pair loss more “focus” on learning the semantic consistency. Furthermore, the local image loss helps generate more vivid image details. As demonstrated in Figure 9, although both models can successfully capture the semantic details in the text, the ‘w/ local’ model generates complex object structures described in the conditioned text more precisely.

Design principles StackGAN claims the failure of directly training a vanilla 256^2 GAN to generate meaningful images. We test this extreme case using our method by removing all nested discriminators without the last one. Our method still generates fairly meaningful results (the first row of Table 4), which demonstrate the effectiveness of our proposed framework (see Section 3.4).

Initially, we tried to share the top layers of the hierarchical-nested discriminators of HDGAN inspired by [25]. The intuition is that all discriminators have a common goal to differentiate real and fake despite difficult scales, and such sharing would reduce their inter-variances. However, we did not observe benefits from this mechanism and our independent discriminators can be trained fairly stably.

HDGAN has a very succinct framework, compared most

	Inc. score	VS
w/o local image loss	$3.12 \pm .02$	$.263 \pm .130$
w/ local image loss	$3.45 \pm .07$	$.296 \pm .130$

Table 5: Ablation study of the local image loss on Oxford-102. See text for detailed explanations.



Figure 9: Qualitative evaluation of the local image loss. The two images w/ the local image loss more precisely exhibit the complex flower petal structures described in the (colored) text.

existing methods, as they [43, 3] adds extra supervision on output images to ‘inject’ semantic information, which is shown helpful for improving the inception score. However, it is not clear that whether these strategies can substantially improve the visual quality, which is worth further study.

5. Conclusion

In this paper, we present a novel and effective method to tackle the problem of generating images conditioned on text descriptions. We explore a new dimension of playing adversarial games along the depth of the generator using the hierarchical-nested adversarial objectives. A multi-purpose adversarial loss is adopted to help render fine-grained image details. We also introduce a new evaluation metric to evaluate the semantic consistency between generated images and conditioned text. Extensive experiment results demonstrate that our method, namely HDGAN, can generate high-resolution photographic images and performs significantly better than existing state of the arts on three public datasets.

References

- [1] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. **10**
- [2] D. Berthelot, T. Schumm, and L. Metz. Began: Boundary equilibrium generative adversarial networks. *arXiv preprint arXiv:1703.10717*, 2017. **2, 3**
- [3] M. Cha, Y. Gwon, and H. T. Kung. Adversarial nets with perceptual losses for text-to-image synthesis. *arXiv preprint arXiv:1708.09321*, 2017. **2, 8**

- [4] Q. Chen and V. Koltun. Photographic image synthesis with cascaded refinement networks. *ICCV*, 2017. 2, 10
- [5] P. Costa, A. Galdran, M. I. Meyer, M. D. Abràmoff, M. Niemeijer, A. M. Mendonça, and A. Campilho. Towards adversarial retinal image synthesis. *arXiv preprint arXiv:1701.08974*, 2017. 2
- [6] A. Dash, J. C. B. Gamboa, S. Ahmed, M. Z. Afzal, and M. Liwicki. Tac-gan-text conditioned auxiliary classifier generative adversarial network. *arXiv preprint arXiv:1703.06412*, 2017. 2, 5
- [7] E. L. Denton, S. Chintala, R. Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *NIPS*, pages 1486–1494, 2015. 2, 3
- [8] H. Dong, S. Yu, C. Wu, and Y. Guo. Semantic image synthesis via adversarial learning. *ICCV*, 2017. 1, 2
- [9] A. Dosovitskiy and T. Brox. Generating images with perceptual similarity metrics based on deep networks. In *NIPS*, 2016. 2
- [10] I. Durugkar, I. Gemp, and S. Mahadevan. Generative multi-adversarial networks. *arXiv preprint arXiv:1611.01673*, 2016. 2, 3
- [11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS*, 2014. 2, 3
- [12] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *ECCV*, 2016. 4
- [13] X. Huang, Y. Li, O. Poursaeed, J. Hopcroft, and S. Belongie. Stacked generative adversarial networks. *CVPR*, 2017. 2, 3
- [14] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 4
- [15] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *CVPR*, 2017. 1, 2, 4
- [16] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 2016. 2
- [17] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *arXiv preprint arXiv:1710.10196*, 2016. 2, 3, 5
- [18] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 10
- [19] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2
- [20] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *ICLR*, 2013. 4
- [21] R. Kiros, R. Salakhutdinov, and R. S. Zemel. Unifying visual-semantic embeddings with multimodal neural language models. *arXiv preprint arXiv:1411.2539*, 2014. 5
- [22] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. *CVPR*, 2017. 2
- [23] C.-Y. Lee, S. Xie, P. Gallagher, Z. Zhang, and Z. Tu. Deeply-supervised nets. In *AIS*, 2015. 2
- [24] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 2, 5
- [25] M.-Y. Liu, T. Breuel, and J. Kautz. Unsupervised image-to-image translation networks. *arXiv preprint arXiv:1703.00848*, 2017. 8
- [26] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 2
- [27] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. P. Smolley. Least squares generative adversarial networks. *arXiv preprint ArXiv:1611.04076*, 2016. 3
- [28] L. Metz, B. Poole, D. Pfau, and J. Sohl-Dickstein. Unrolled generative adversarial networks. *arXiv preprint arXiv:1611.02163*, 2016. 2
- [29] T. D. Nguyen, T. Le, H. Vu, and D. Phung. Dual discriminator generative adversarial nets. In *NIPS*, 2017. 3
- [30] M.-E. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In *ICCVGIP*, 2008. 2, 5
- [31] A. Odena, C. Olah, and J. Shlens. Conditional image synthesis with auxiliary classifier gans. *arXiv preprint arXiv:1610.09585*, 2016. 2, 5
- [32] A. v. d. Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel recurrent neural networks. *ICML*, 2016. 2
- [33] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 2
- [34] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee. Generative adversarial text to image synthesis. *ICML*, 2016. 1, 2, 3, 5, 10
- [35] S. E. Reed, Z. Akata, S. Mohan, S. Tenka, B. Schiele, and H. Lee. Learning what and where to draw. In *NIPS*, 2016. 2, 5
- [36] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training gans. In *NIPS*, 2016. 2
- [37] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen, and X. Chen. Improved techniques for training gans. In *NIPS*. 2016. 2, 5
- [38] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb. Learning from simulated and unsupervised images through adversarial training. *CVPR*, 2017. 2, 4
- [39] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, June 2015. 5
- [40] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016. 10
- [41] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-ucsd birds 200. 2010. 2, 5
- [42] S. Xie and Z. Tu. Holistically-nested edge detection. In *ICCV*, 2015. 2
- [43] T. Xu, P. Zhang, Q. Huang, H. Zhang, Z. Gan, X. Huang, and X. He. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. *arXiv preprint arXiv:1711.10485*, 2017. 2, 8
- [44] 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. *arXiv preprint arXiv:1506.03365*, 2015. 7

- [45] H. Zhang and K. Dana. Multi-style generative network for real-time transfer. *arXiv preprint arXiv:1703.06953*, 2017. 2
- [46] H. Zhang, V. Sindagi, and V. M. Patel. Image de-raining using a conditional generative adversarial network. *arXiv preprint arXiv:1701.05957*, 2017. 2
- [47] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *ICCV*, 2017. 1, 2, 3, 4, 5, 10
- [48] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. Metaxas. Stackgan++: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *arXiv preprint arXiv:1710.10916*, 2017. 5
- [49] Z. Zhang, Y. Xie, F. Xing, M. McGough, and L. Yang. Mdnet: A semantically and visually interpretable medical image diagnosis network. In *CVPR*, 2017. 2
- [50] Z. Zhang, L. Yang, and Y. Zheng. Translating and segmenting multimodal medical volumes with cycle- and shape-consistency generative adversarial network. In *CVPR*, 2018. 2
- [51] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *CVPR*, 2017. 2
- [52] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. *ICCV*, 2017. 2, 4, 10

6. Supplementary Material

6.1. Training and Architecture Details

The training procedure is similar to the one used in standard GANs, which alternatively updates the generator and discriminators until converge.

The Adam optimizer [18] is used. The initial learning rate is set as 0.0002 and decreased by half for every 100 epochs (50 for COCO). The model is trained for 500 epochs in total (200 epochs for COCO). We configure the side outputs at 4 different scales where the feature map resolution is equal to 64^2 , 128^2 , 256^2 , and 512^2 , respectively. For the local image loss of these 4 side outputs, we set $R_1 = 1$, $R_2 = 1$, $R_3 = 5$, and $R_4 = 5$. For example, R_1 refers to 64^2 . These numbers are not fine-tuned but are set empirically. We believe there exists better configurations to be explored.

All intermediate conv layers, except from the specified ones in Section 3.4, use 3×3 kernels (with reflection padding). Some other normalization also layers are experimented (i.e. instance normalization [40] and layer normalization [1]) since they are used by recent advances [52, 4]. But the results are not satisfactory.

With respect to the generator, we use 1-repeat residual blocks for the generator till the 256^2 resolution. The input of the generator is a $1024 \times 4 \times 4$ tensor. As the feature map resolution increases by 2, the number of feature maps is halved at $8, 32, 128, 256$ sizes. To generate 512^2 images, we pre-train the generator to 256^2 due to the limitation of the GPU memory. We use a 3-repeat res-block followed by a stretching layer to upscale the feature map size to $32 \times 512 \times 512$, and a linear compression layer to generate images. Since the 256^2 image already captures the overall semantics and details, to boost the training and encourage the 512^2 maintain this information, we use a 11 reconstruction loss to ‘self-regularize’ the generator.

6.2. More Qualitative Results and Analysis

In this section, we demonstrate more sample results for the three datasets.

Figure 10 compares our results with StackGAN. For each input, 6 images are randomly sampled. Furthermore, we visualize zoomed-in samples compared with StackGAN in Figure 11. Our results demonstrate obviously better quality, less artifacts, and less sharp pixel transitions.

Figure 13 shows the results on the CUB bird dataset. All the outputs of a model with different resolutions are also shown. As can be observed in this two figures, our method can generate fairly vivid images with different poses, shape, background, etc. Moreover, the images with different resolutions, which are side outputs of a single model, have very consistent information. More and more image details can be observed as the resolution increases. Figure 14 shows the results on the Oxford-102 flower dataset. Very detailed petals can be generated with photographic colors and saturation.

Figure 15 shows some sampled results on the COCO dataset. COCO is much more challenging than the other two datasets since it contains natural images from a wide variety of scenes containing hundreds of different objects. As can be observed in the shown samples, our method can still generate semantically consistent images.

However, it is worth to notice that, although our method significantly improves existing methods [47, 34] on COCO, we realize that generating fine-grained details of complex natural scenes with various objects is still challenging. Based on this study, we expect to further address this problem as the future study.

Failure cases: Although we observed that the majority of test data can result in successful outputs (at least one sample of a single input text), there are still observable failure cases. The major problems include obvious artifacts, minor semantic inconsistency (compared with groundtruth), loss of object basic shapes. Figure 12 shows these mentioned failure cases. To compare with StackGAN category by category, please refer to Table 3 left (in the main paper).

This is a tiny bird with a large protruding tan chest and a short black beak that has grey wings

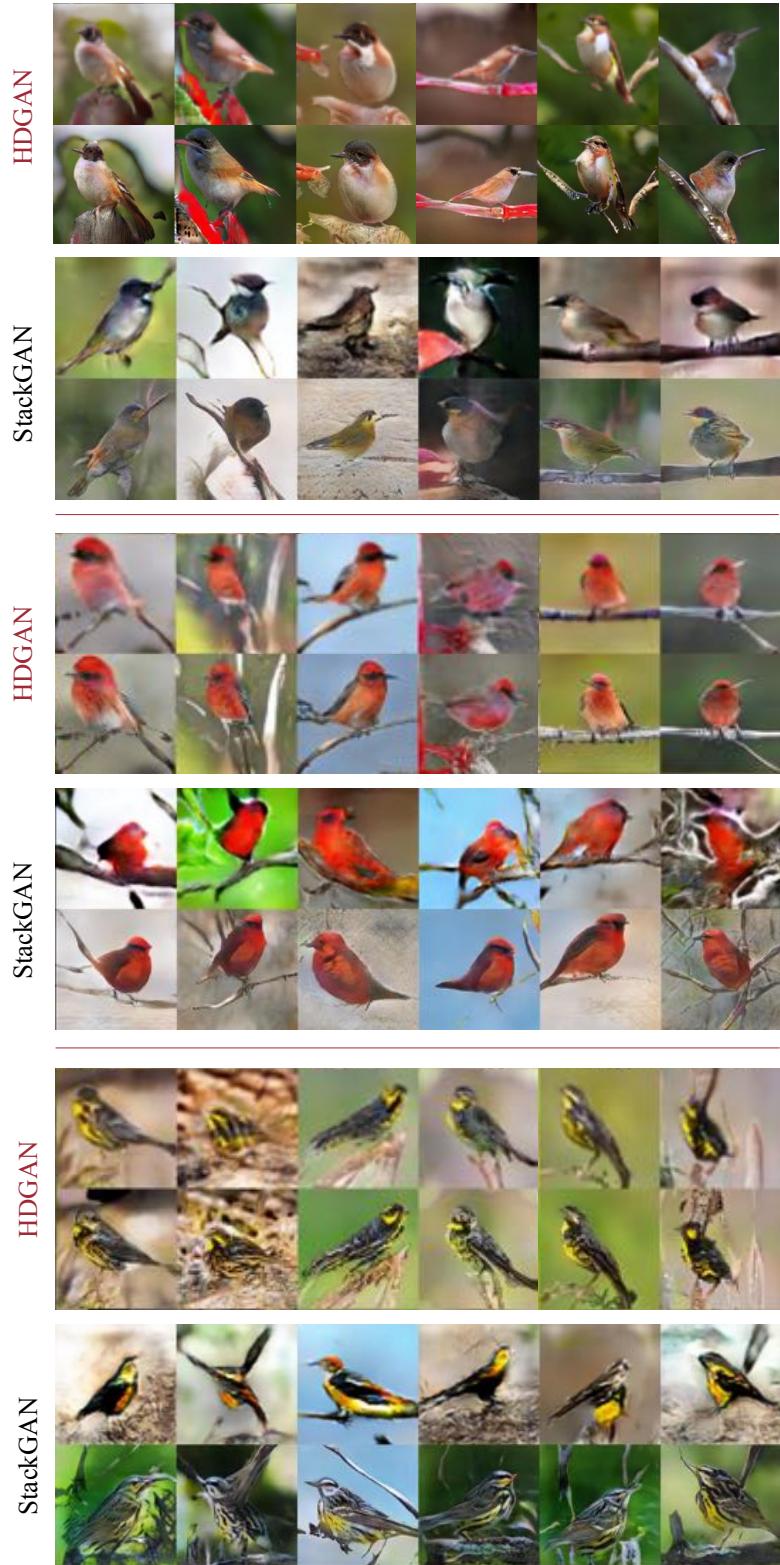


Figure 10: Sample results on CUB compared with StackGAN. For each input, 6 samples are shown with resolutions of 64^2 and 256^2 , respectively. As can be obviously seen, our HDGAN can generate very consistent content in images of different resolutions. Moreover, our generated images show more photographic color and contrast.

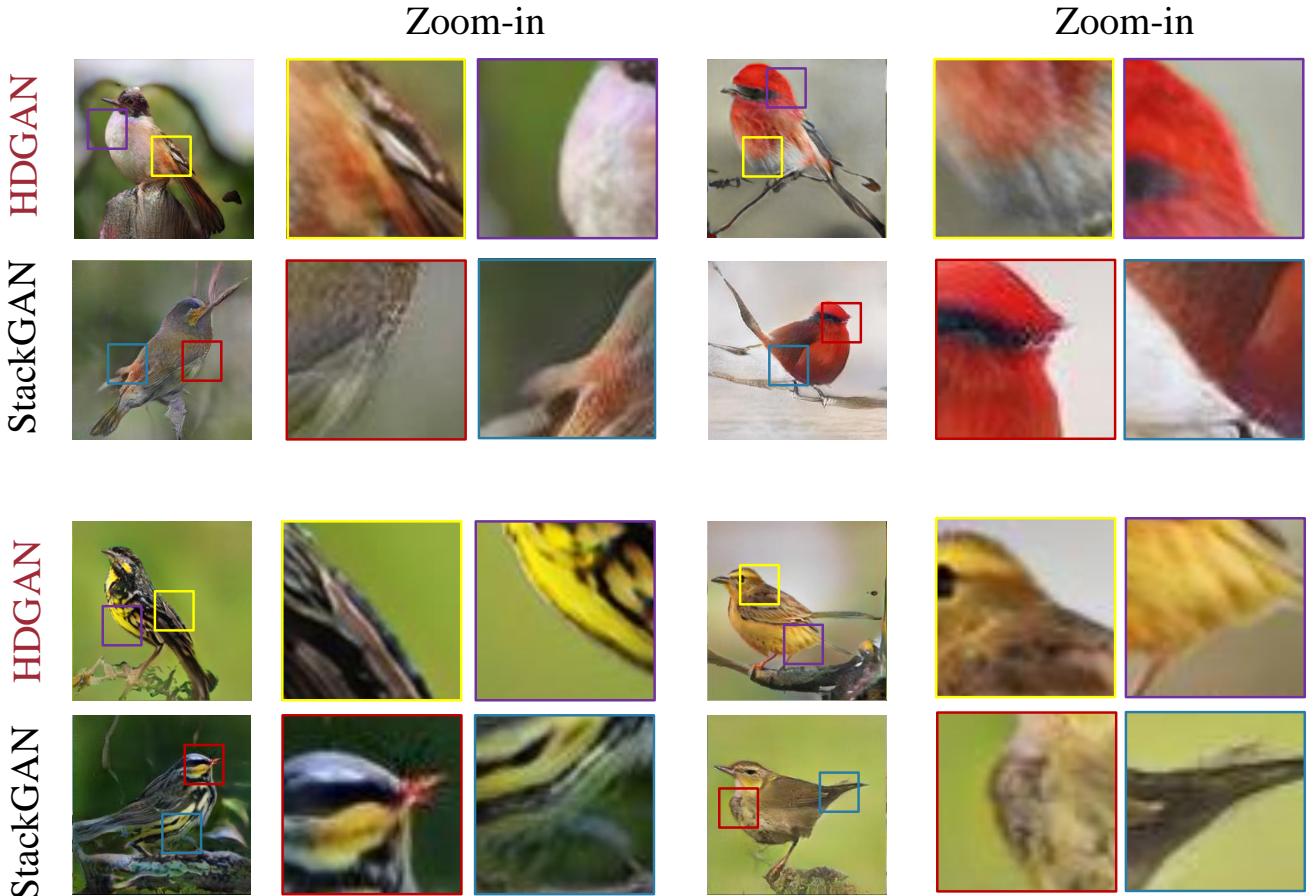


Figure 11: Zoomed-in samples compared with StackGAN. The best sample among 6 samples given an input text is selected. It can be clearly observed that our results show more smooth visual results. Especially, much less sharp pixel transitions exist in our results.

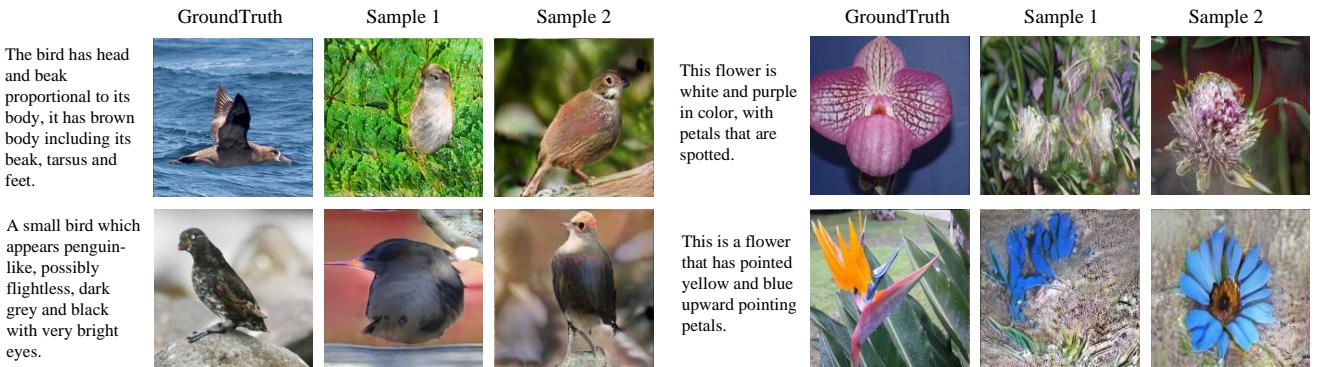
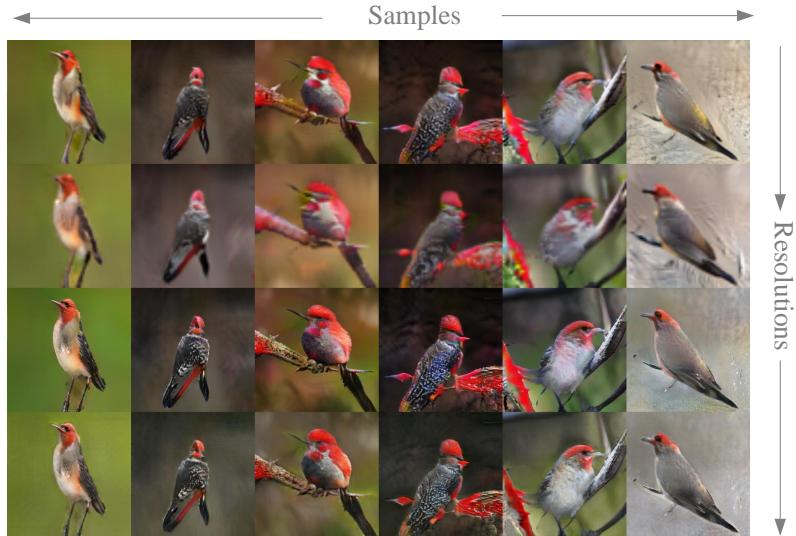


Figure 12: Illustration of failure cases due to artifacts, minor semantic inconsistency (compared with groundtruth), loss of object basic shapes. For each input text, we sampled ten samples. The figure shows a worst failed image (Sample 1) and a relative better image (Sample 2).

A small bird with a red crown and straight bill sits perched atop a branch



A small yellow/green bird with black wings and white wingbars

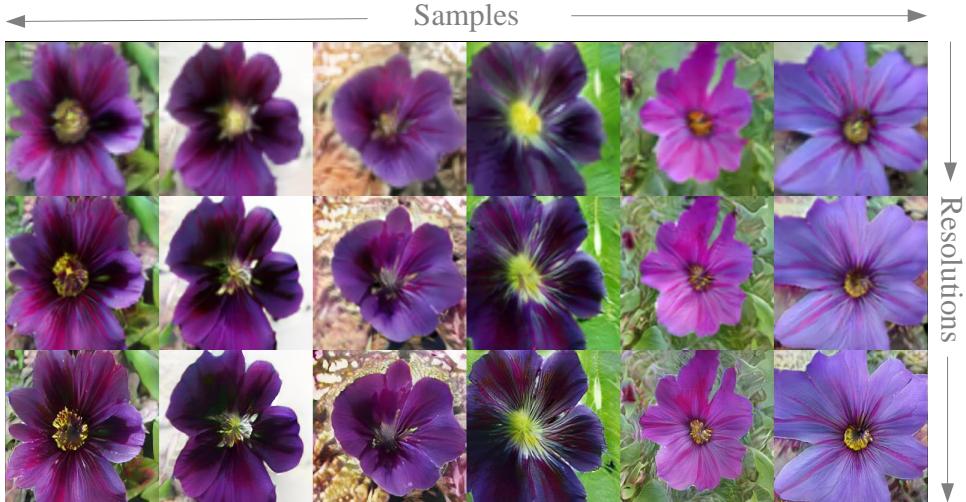


The bird has sharp pointed beak with grayish yellow throat, with white eyering



Figure 13: Sample results on CUB. For each input, 6 samples with resolutions of 64^2 , 128^2 , 256^2 , and 512^2 are shown in 4 rows, respectively.

This flower is purple and red in color, and has petals that are striped



This flower is pink and yellow in color, with petals that are skinny and oval.



Flower has petals that are pale pink with yellow stamen

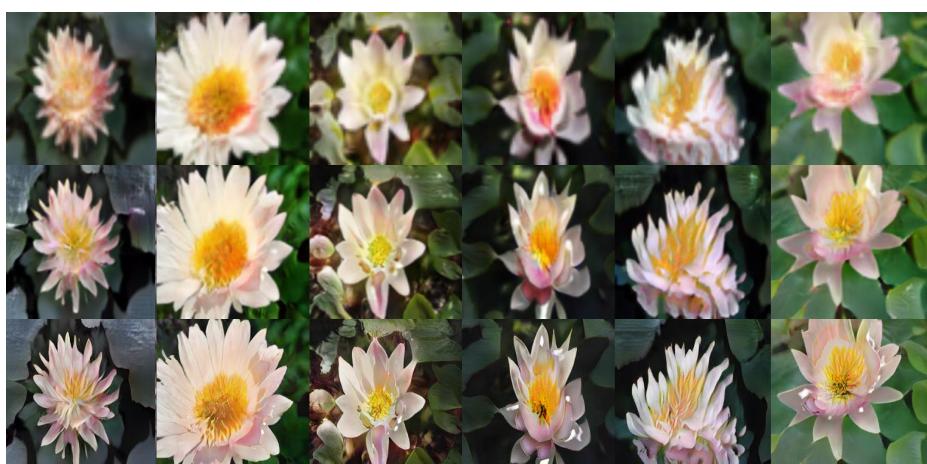


Figure 14: Sample results on Oxford-102. For each input, 6 samples with resolutions of 64^2 , 128^2 , and 256^2 are shown in 3 rows, respectively.

A close up of a plate of food containing broccoli



A photo of a train on a bridge above a river



A beach on a sunny day with a bunch of people on it.



A living room filled with lots of furniture



A man riding a kiteboard on top of the ocean



A very tall cathedral towering over a city



A group of people carrying ski equipment while walking on snow covered ground.



People playing with kites outside in the desert



Figure 15: Sample results on COCO. We show 8 256^2 samples in very different scenes.