Paired-D GAN for Semantic Image Synthesis

Duc Minh Vo[1(✉)] and Akihiro Sugimoto[2]

[1] Department of Informatics,
The Graduate University for Advanced Studies (SOKENDAI), Tokyo, Japan
[2] National Institute of Informatics, Tokyo, Japan
{vmduc,sugimoto}@nii.ac.jp

Abstract. Semantic image synthesis is to render foreground (object) given as a text description into a given source image. This has a wide range of applications such as intelligent image manipulation, and is helpful to those who are not good at painting. We propose a generative adversarial network having a pair of discriminators with different architectures, called *Paired-D GAN*, for semantic image synthesis where the two discriminators make different judgments: one for foreground synthesis and the other for background synthesis. The generator of paired-D GAN has the encoder-decoder architecture with skip-connections and synthesizes an image matching the given text description while preserving other parts of the source image. The two discriminators judge foreground and background of the synthesized image separately to meet an input text description and a source image. The paired-D GAN is trained using the effective adversarial learning process in a simultaneous three-player minimax game. Experimental results on the Caltech-200 bird dataset and the Oxford-102 flower dataset show that Paired-GAN is capable of semantically synthesizing images to match an input text description while retaining the background in a source image against the state-of-the-art methods.

1 Introduction

Very recently proposed semantic image synthesis [1] is to manipulate a given source image semantically with given text descriptions, while still maintain features that are irrelevant to what text descriptions. Text descriptions are usually on foreground (objects), and thus the task is to render foreground given as a text description into a given source image. Since text descriptions [2] is easier and more natural for us than image descriptions such as attributes [3], textures [4] or styles [5], semantic image synthesis is promising to widen the range of applications of image synthesis.

Generative Adversarial Network (GAN) [6] is capable of synthesizing images, and work has been proposed that conditions GAN on either text descriptions [2, 7] or images [8–10] to synthesize images for various tasks. Almost all work on image synthesis [1,2,7,11] follows the original GAN architecture where a single

C. V. Jawahar et al. (Eds.): ACCV 2018, LNCS 11364, pp. 468–484, 2019.
https://doi.org/10.1007/978-3-030-20870-7_29

discriminator judges whether a synthesized image is realistic. Despite obtaining remarkable results, synthesizing realistic images directly from text descriptions is still difficult. This is due to the gap between the semantic levels of images and text descriptions.

Semantic image synthesis requires to disentangle the semantics contained in image and text information and then combine the disentangled semantics to synthesize realistic images. This suggests to separately deal with text descriptions and images with different semantic levels. We thus design a GAN with a pair of discriminators, called *Paired-D GAN*, to separately condition text descriptions and images. Indeed, dual discriminator GAN [12] showed that having two discriminators is more effective than GANs with one discriminator for image synthesis. Different from dual discriminator GAN, we design different architectures for two discriminators to deal with different levels of semantics of text descriptions and images. The two discriminators separately judge foreground and background of the synthesized image to meet an input text description and a source image. Furthermore, we employ the skip-connection in the generator to more precisely retain background information in the source image. We also introduce a training process for adversarial learning in the three-player minimax game of the generator and two discriminators. In this way, Paired-D GAN improves the quality of synthesized images. Experiments on the Caltech-200 bird dataset [13] and the Oxford-102 flower dataset [14] demonstrate outperformances of Paired-D GAN against [1,15]. Figure 1 shows an example of our results.

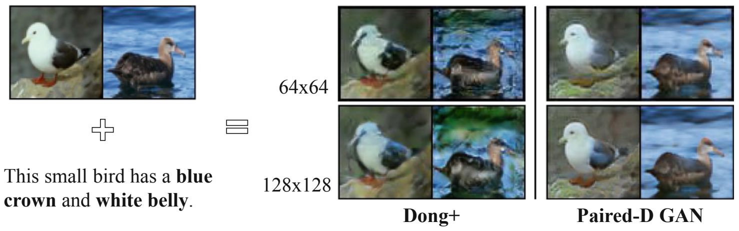


Fig. 1. Examples of synthesized images. Our results match the text description more precisely than [1] while successfully retaining background of the source image. The performance of our method does not change for different sizes of images (64×64 and 128×128 images).

2 Related Work

With the rapid development of deep learning, many models for image synthesis have been proposed to achieve highly realistic images. They include variational auto-encoder [16,17], auto-regressive models [18,19], and GAN [1,2,6,15]. Among them, GAN and its variants show remarkably realistic results.

GAN [6] consists of a generator and a discriminator. The generator maps the latent variable into the data space while the discriminator judges whether the output of the generator is real or fake. The generator and the discriminator are simultaneously trained in a minimax game. Interestingly, GAN can be constrained on various conditions not only to generate plausible images but also to meet the conditions. Some work conditions GANs on the attribute label [3,20] or images [10,21–24] for domain transfer [21,22], photo editing [23], image super-resolution [10], and style transfer [24].

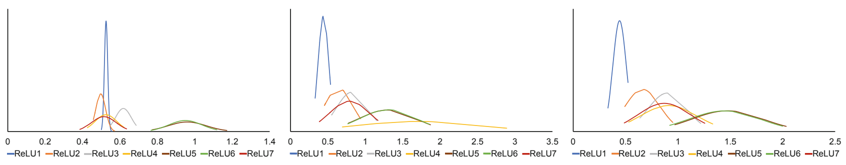
Among various conditions on GAN, text descriptions make image synthesis easier and more friendly to us. Reed et al. [2] proposed an end-to-end GAN using the text condition. They employed a pre-trained text encoder [25] to extract text features from an input text, and then combined text features with a vector representing random noise to produce the input of the generator. They also employed the combination of text features and image features in the discriminator to discriminate real images and generated images. Their proposed model [2] became the baseline of the GAN framework for generating images from text descriptions.

As an extension, a model conditioned on texts and location information was proposed [26]. Models with two stages of GAN, Stack-GAN [7] (and Stack-GAN++ [11]), were also proposed, showing successfully generated higher resolution images (256×256), compared to [2] (64×64). These models [2,7,11,15] condition on GAN only texts or a pair of texts and location information [26].

Addressing the background problem in image synthesis, Yang et al. [15] proposed to decompose the image synthesis into two phases using foreground and background generators. They fed random noise vectors to a long short-term memory (LSTM) network to obtain hidden states for the foreground generator and used the first hidden state to generate background. They then combined foreground and background by a compositor operator. However, decomposing foreground and background may cause less realistic images.

The model proposed by Dong et al. [1] is most related with ours. It also conditions text and source image on GAN. The architecture of the model is, however, similar to [2] and has a single discriminator: the noise vector in [2] is replaced by image features from the image encoder. Though it generates images that match the semantic meaning of the input text description while maintaining other parts of a source image, it does not preserve background precisely because the discriminator is used only for foreground; synthesized images are less realistic images.

Different from the above mentioned models, we fully take into account each role of foreground and background in synthesized images. More precisely, our proposed Paired-D GAN is conditioned on both text descriptions and images, has skip-connections in its generator to preserve background information as much as possible, and has two discriminators with different architectures for synthesizing realistic images. Paired-D GAN generates simultaneously foreground and background.



(a) Same background (with different foregrounds). (b) Different backgrounds (with one foreground). (c) Different backgrounds (with another foreground).

Fig. 2. Distribution of the mean values of the first 7 ReLU layers in VGG-16.

3 Semantic Levels of Image Features for Foreground/Background

Convolution Neural Network (CNN) has proved the effectiveness in many tasks. Along with the depth, CNN extracts different semantic levels of image features in layers. Gatys et al. [5] pointed out that features in early layers reflect color or texture of images while features in latter layers convey foreground information. The work [27] also found that features in early layers address background while foreground is obtained in latter layers. As [28] learned the statistic of image features, we experimentally exploit semantic levels of image features in VGG-16 [29].

We randomly prepare 10 foreground images and 8 background ones. We then generated 100 images for each pair of foreground-background images with randomly localizing foreground (we have 8000 images in total). We feed these generated images into the VGG-16 [29] pre-trained on ImageNet dataset [30] without any fine-tuning to compute the mean activation at each Rectified Linear Unit (ReLU) layer [31].

The distribution of the mean activations in all 13 ReLU layers shows that the first 7 ReLU layers are more sensitive to background and foreground than the other ReLU layers. In the case where background is the same (Fig. 2a), the distribution of the mean values is small at the 1st – 3rd ReLU layers and becomes larger from the 4th ReLU layer. This suggests that VGG-16 recognizes the similarity of images at the 1st – 3rd ReLU layers and starts to learn differences of images from the 4th ReLU layer (though not strictly clear at the 4th layer).

On the other hand, in the case where backgrounds are different (Fig. 2b, c), the values at the 1st – 3rd ReLU layers are larger and similar with each other even if foregrounds are different (compared to the same background case). This observation is in good harmony with the same background case. If we change foreground (Fig. 2b, c), the distribution of mean values is completely different from the layer 4th to the 7th layer (at each layer respectively).

Combining insights given by [5,27], we may thus conclude that VGG-16 weights background in early layers and foreground at latter layers. More precisely, from the 1st to the 3rd ReLU layers capture background while from the 5th to the 7th ReLU layers do foreground, and the 4th ReLU layer seems to be in-between as a transition.

Using appropriate semantic levels of image features for discriminators is crucial. We use above observation for employing appropriate semantic levels of image features for foreground and background. Namely, we use features from the 1st to the 3rd ReLU layers for background and those from the 5th to the 7th ReLU layers for foreground. We remark that more deeply exploring background-foreground relation is preferable.

4 Proposed Method

4.1 Network Design

Our network follows the GAN architecture [6] for image synthesis [1,2,7,11]. Like [1], we condition GAN on both text descriptions and a source image. As seen in Sect. 3, we use different semantic levels of features depending on foreground and background. Namely, we design the network in which a text description on foreground matches features in latter layers while features of a source image in early layers are preserved as much background information as possible. This appropriate-level selection allows our model to synthesize realistic images that meet both a text description and a source image.

Nguyen et al. [12] argued that dual discriminators in GAN generate better images in quality than a single discriminator, though the two discriminators has the same architecture. To deal with foreground and background separately and more precisely, we employ a pair of discriminators where each of them independently judges foreground/background of synthesized images. For different semantic levels of foreground and background, we design our discriminators with different architectures and make each play a different role. Namely, we design one discriminator to evaluate matching foreground between a text description and a synthesized image following [1,2,7] and the other discriminator to evaluate whether background of a source image is retained in the synthesized image. We also introduce an effective training strategy for adversarial learning in a three-player minimax game.

4.2 Network Architecture

We build our network, called Paired-D GAN, upon the GAN architecture with one generator G and a pair of discriminators, foreground discriminator D_{FG} and background discriminator D_{BG} (Fig. 3). We employ the end-to-end encoder-decoder architecture for our generator G following [1]. The generator G receives a source image and a text description where the source image is with the size of $n \times n \times 3$ (n can be 64, 128 or 256; 3 are for RGB channels) and the text description is with maximum of 50 words. G synthesizes an image of $n \times n \times 3$ that adaptively changes foreground to match the text description while retaining background of the source image.

Two discriminators D_{FG}, D_{BG} evaluate whether the synthesized image is real or generated. D_{FG} receives the generated image and the ground-truth foreground image with the text feature extracted from the text description to focus

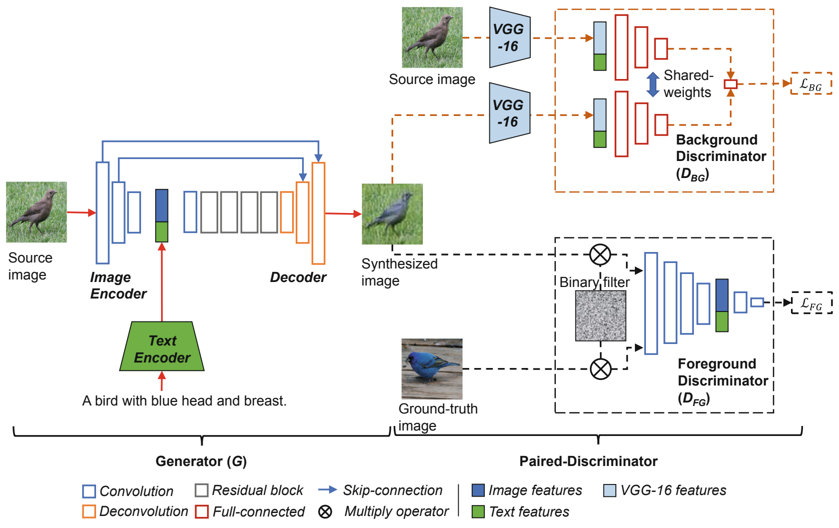


Fig. 3. Framework of our proposed Paired-D GAN.

on foreground evaluation. D_{BG}, on the other hand, receives the image feature extracted from the source image and the text feature extracted from the text description to focus on background evaluation. We use the pre-trained VGG-16 to extract image features from input images for D_{BG} as mentioned in Sect. 3. We remark that the two discriminators do not share their parameters.

We train G, D_{FG}, and D_{BG} simultaneously in a three-player minimax game using adaptive loss functions. This adversarial learning process enables our generator G to generate plausible images that mach text descriptions while preserving background information of the source image.

4.2.1 Generator

Our generator G consists of an image encoder, a text encoder, and a decoder.

The image encoder is a stack of three convolution layers that receives the source image size of $n \times n \times 3$ to produce an image feature with the size of $m \times m \times 512$ (m can be 16, 32 or 64 depending on n) at the top. We adopt the pre-trained text encoder [25] for our text encoder and use the text embedding augmentation [7] to produce a text feature with the size of 1×128. The channel of the text feature is duplicated to the size of $m \times m \times 128$ to be consistent with that of the image feature.

The image feature and the text feature are then concatenated to produce an image-text feature as the input of the decoder.

The decoder in our generator consists of one convolution layer, four residual blocks [32], and two deconvolution layers. The convolution layer reduces the channel of the image-text feature, and the four residual blocks enrich feature maps. The two deconvolution layers, on the other hand, upscale the feature maps.

We remark that each of the convolution and deconvolution layers in the image encoder and the decoder is followed by a batch normalization (BN) layer [33] and a ReLU layer. The only exception is the last deconvolution layer in the decoder where it uses the tanh activation to guarantee that the range of the output can be normalized to be $[0, 255]$ (in the test step). We remark that we use images with the range $[-1, 1]$ in the training step.

To reflect the features at early layers weighting background information into a synthesized image, we employ the skip-connection from the image encoder to the decoder. More precisely, the first layer in the image encoder is connected to the last layer in the decoder while the second layer in the image encoder is paired with the second last layer in the decoder.

4.2.2 Foreground Discriminator

The foreground discriminator should be able to discriminate foreground of real images and that of generated images. We employ the foreground-text matching in the foreground discriminator. Following previous work [1,2,7,11], we design our foreground discriminator D_{FG} as a classification task that rewards high probability scores to real images and low ones to generated images in the adversarial learning phase.

Our D_{FG} is a stack of six convolution layers.

Each of the first four convolution layers uses the filter size of 4×4, the reflection-padding size of 1×1, and the stride size of 2×2, producing 64, 128, 256, 512 output channels, respectively. These convolution layers encode an input to produce the high-level semantic image features containing mostly foreground information (cf. Sect. 3). These image features are then concatenated with the text feature obtained from the input text description using the text encoder to produce a image-text feature.

Next, the image-text feature is fed into the last two convolution layers, each of which is with the filter size of 1×1, and 4×4, respectively, no padding, the stride size of 1×1, outputting 512, 4 channels respectively. The output of the last convolution layer indicates how realistic the image input to D_{FG} is using the similarity probability.

We remark that each of the all convolution layers except for the last one is followed by a BN layer and a ReLU layer. We follow Reed et al. [2] to train D_{FG} (Eq. 1).

D_{FG} need not access all image information but focuses on foreground image information. To enhance the performance of D_{FG}, we introduce a processing before feeding an input image to D_{FG}. Namely, we create a binary filter where 0 at each pixel is generated with the probability of p. We then apply the binary filter to the image input to D_{FG}, and feed the filtered image to D_{FG}. This processing brings two benefits: (1) D_{FG} has more chance to focus on only foreground information, helping to extract semantic image features of foreground, and (2) this operation prevents quick convergence [34].

4.2.3 Background Discriminator

The background discriminator evaluates how real and generated images are different in background. We therefore design the background discriminator as a verification task with the limited number of samples in each category. This is because each image in a dataset has different background in general, and the number of samples with the same (very similar) background is limited. To this end, we follow the idea of the Siamese network [35] because it shows the effectiveness for the verification task.

Our D_{BG} consists of four fully-connected layers in which the first three layers are two shared-parameter layers and the last one is the joint layer, producing 512, 100, 10, 1 outputs, respectively. D_{BG} receives two input features (one from the source image with the text description and the other from the generated image with the text description) and passes them to the two shared-parameter layers separately before being jointly trained at the last layer.

In order to create the input of D_{BG}, we feed the input image into the pre-trained VGG-16 to compute the mean and variance at the first four ReLU layers (cf. Sect. 3), and then concatenate them with the text feature extracted from the input text description using the pre-trained text encoder [25] (without using the text embedding augmentation [7]). The text feature is useful to disentangle background and foreground information (e.g. images with the same background and different foreground information can be positive samples for the background verification task). We remark that the size of the input is 1×1068 where the image feature is with the size of 1×768 and the text feature is with the size of 1×300.

We propose a new training strategy for D_{BG}, which is based on the contrastive loss function [35] that fully uses a source image and a text description.

4.3 Adversarial Learning for Paired-GAN

Training the generator G, and a pair of discriminators D_{FG} and D_{BG} becomes a three-player minimax game conditioned on images and text descriptions. Using positive/negative training samples, we first update the parameters of D_{FG} with fixing the parameters of D_{BG} and G, and then update the parameters of D_{BG} with fixing the parameters of D_{FG} and G, and finally update the parameters of G with fixing the parameters of the two discriminators. We iterate this adversarial training to minimize each loss function separately.

For the adversarial training for Paired-D GAN, we use positive and negative samples whose definitions depend on D_{FG} and D_{BG}. A positive sample of D_{FG} is a sample in which foreground is the ground-truth and its text description is matching. A sample is negative if (1) foreground is the ground-truth but its text description is mismatching or (2) foreground is generated even if its text description is matching. A positive sample of D_{BG}, on the other hand, is the one where the background of the source image used in training the generator and discriminators for each iteration is the same regardless of whether text descriptions are matching or mismatching. A sample is negative if background is generated even if the text descriptions match foreground.

Table 1. Types of input pairs used in the adversarial leaning process.

	D_{FG}	D_{BG}
Positive	$\{g,\, \varphi(t)\}$	$\{(s,t),\, (s,\bar{t})\}$
Negative	$\{g,\, \varphi(\bar{t})\}$, $\{G(s,\varphi(t)),\, \varphi(t)\}$	$\{(G(s,\varphi(t)),t),\, (G(\bar{s},\varphi(t)),t)\}$

Let s be an image in a dataset and t be a text description. Then, we let g be an image in the dataset whose foreground is the ground-truth to t (t is thus a matching text description to g). We denote by $\bar{s}$ a randomly selected image (from the dataset) having different background from s, and by $\bar{t}$ a different text description from t (a mismatching text description to g). We also denote by $\varphi(\cdot)$ the text embedding augmentation [7]. Then, positive/negative samples of D_{FG} and D_{BG} can be classified as in Table 1.

Let $D(\cdot)$ denote the discriminators (D_{FG} and D_{BG}). At each iteration in training $D(\cdot)$, we randomly select all the types of samples in Table 1 from the training dataset, and feed them one by one to $D(\cdot)$ to obtain the probability whether the sample is positive or negative. We train the two discriminators to reward a high score to a positive sample and a low score to a negative sample. Through the training, we maximize the ability of $D(\cdot)$ to assign relevant scores to the samples. The loss functions for $D(\cdot)$ are defined as follows:

$$\begin{aligned}\mathcal{L}_{\mathrm{FG}} = {} & \mathbb{E}_{(g,t)\sim p_{\mathrm{data}}}[\log D_{\mathrm{FG}}(g,\varphi(t))] + \frac{1}{2}\mathbb{E}_{(g,\bar{t})\sim p_{\mathrm{data}}}[\log(1 - D_{\mathrm{FG}}(g,\varphi(\bar{t})))] \\ & + \frac{1}{2}\mathbb{E}_{(s,t)\sim p_{\mathrm{data}}}[\log(1 - D_{\mathrm{FG}}(G(s,\varphi(t)),\varphi(t)))]. \end{aligned} \tag{1}$$

$$\begin{aligned}\mathcal{L}_{\mathrm{BG}} = {} & \mathbb{E}_{(s,t,s,\bar{t})\sim p_{\mathrm{data}}}[\log D_{\mathrm{BG}}((s,t),(s,\bar{t}))] \\ & + \mathbb{E}_{(s,t,\bar{s},t)\sim p_{\mathrm{data}}}[\log(1 - D_{\mathrm{BG}}((G(s,\varphi(t)),t),(G(\bar{s},\varphi(t)),t)))], \end{aligned} \tag{2}$$

where p_{data} denotes the all the training data and $\mathbb{E}_{(\cdot)\sim p_{\mathrm{data}}}$ means the expectation over p_{data}. Each term in Eqs. 1 and 2 corresponds to the type of samples: $\log(D(\cdot))$ for positive samples and $\log(1 - D(\cdot))$ for negative samples. Note that Eq. 1 follows [2].

Since our adversarial learning process is a three-player minimax game, we also train the generator G in which we minimize the terms of $\log(1 - D(\cdot))$ in Eqs. 1 and 2. In practice, however, maximizing $\log(D(\cdot))$ is known to be better than minimizing $\log(1 - D(\cdot))$ in training G [6]. We also introduce the reconstruction loss to keep the structure of the input source image. Now the loss function for G is:

$$\begin{aligned}\mathcal{L}_{\mathrm{G}} = {} & \mathbb{E}_{(s,t)\sim p_{\mathrm{data}}}[\log(D_{\mathrm{FG}}(G(s,\varphi(t)),\varphi(t))] \\ & + \mathbb{E}_{(s,t,s,\bar{t})\sim p_{\mathrm{data}}}[\log(D_{\mathrm{BG}}((G(s,\varphi(t)),t)),(G(s,\varphi(\bar{t})),\bar{t})))] \\ & + \lambda\mathbb{E}_{(s,t)\sim p_{\mathrm{data}}}\left\|s - G(s,\varphi(t))\right\|_2, \end{aligned} \tag{3}$$

where λ is the hyperparameter, and $\|.\|_2$ is the Euclidean distance. To train G, we randomly select an image s, and two text descriptions t and $\bar{t}$ to generate the synthesized images. We then feed them to the D_{FG} and D_{BG} to receive feedback signals for updating parameters of G. We remark that since our aim is not to reconstruct the source image, λ can be small (we set $\lambda = 0.0001$ in our experiments).

As discussed in [2], training D_{FG} with match and mismatching text descriptions enables D_{FG} to feedback stronger image-text matching signals, allowing G to generate plausible images that match text descriptions. Our usage of a pair of image and a text description in training D_{BG}, on the other hand, enables D_{BG} to generate stronger signals as well, leading to the capability of G of retaining background information (though at the beginning, D_{BG} spends more time to verify background, D_{BG} gradually need not concern foreground thanks to text descriptions, and has ability of easily judging whether the image is real or generated). Accordingly, the above adversarial learning brings to Paired-D GAN the capability of generating realistic images that match text descriptions in foreground and precisely retain background of source images.

5 Experiments

5.1 Dataset and Compared Methods

Dataset. We used the Caltech-200 bird dataset [13] and the Oxford-102 flower dataset [14]. The Caltech-200 bird dataset contains 11,788 images belonging to one of 200 different bird classes. The Oxford-102 flower dataset has 8,189 images with 102 classes of the flower. Each image in the datasets has 10 captions collected by Reed et al. [25]. Following previous work [1,2], we split the Caltech-200 dataset into 150 training classes and 50 testing classes, and the Oxford-102 dataset into 82 training classes and 20 testing classes. We remark that we resized the images used in our experiments to ones with 64×64.

Compared Methods. We employed the model proposed by Dong+ [1] as the baseline. We also compared our method with Yang+ [15] that generates image foreground and background separately and recursively from input text descriptions (we chose this though the task is different because it generates realistic images). For Dong+ [1], we used the re-implementation by Seonghyeon [36] (as recommended by the authors of Dong+ [1]). For Yang+ [15], we used the publicly available source codes with the parameters recommended by the authors [37]. We remark that we used the combination of a noise vector and a text feature [2] as an input for Yang+ [15].

5.2 Implementation and Training Details

We implemented our model in PyTorch. We adopted the pre-trained text encoder [25] without any fine-tuning. To extract image features for the background discriminator input, we employed the VGG-16 [29] pre-trained on ImageNet dataset [30] without any fine-tuning. Like [1], we also used the image augmentation technique (e.g., flipping, rotating, zooming and cropping). We conducted all the experiments using a PC with CPU 6-cores Xeon 3.7 GHz, 64 GB of RAM, and GTX1080 Titan GPU (11 GB of VRAM).

We optimized the adaptive loss functions (Sect. 4.3) using Adam optimizer [38] with the learning rate of 2×10^{-3}, the learning rate decay of 0.5 performed every 100 epochs, the momentum $\beta_1 = 0.9$ and $\beta_2 = 0.999$, and the division from zero parameter $\epsilon = 10^{-8}$. We did not use the weight decay. We trained our model with the batch size of 48 for 600 epochs.

5.3 Evaluation Metrics

We use the inception score (IS) [39] to evaluate the overall quality of synthesized images. We also use two metrics, foreground score (FGS) and background score (BGS) for evaluating foreground and background of synthesized images separately.

IS is widely used for the generative model evaluation through the output of the Inception-v3 network [40]: $IS(G) \approx \exp(\frac{1}{N}\sum_{i=1}^{N} D_{\mathrm{KL}}(p(y|\hat{x}^{(i)}||\hat{p}(y))))$, where $\hat{x}$ is a synthesized image by the generator G, N is the number of generated images, D_{KL} is the Kullback–Leibler divergence, y indicates an instance of all classes given in the dataset, $p(y|\hat{x})$ is the conditional class distribution, and $\hat{p}(y) = \frac{1}{N}\sum_{i=1}^{N} p(y|\hat{x}^{(i)})$ is the empirical marginal-class distribution.

We employ the visual-text shared-space [25] and compute the matching between text descriptions and foreground for the foreground evaluation: $FGS = \|f_{\mathrm{img}} - f_{\mathrm{text}}\|_2$ where f_{img} and f_{text} are the features from the image encoder and the text encoder.

For background evaluation, we use $BGS = \|\hat{x} \odot \overline{x_{\mathrm{seg}}} - x \odot \overline{x_{\mathrm{seg}}}\|_2$ where x is the source image, and $\odot$ is the element-wise multiplication. $\overline{x_{\mathrm{seg}}}$ is the inverse map of x_{seg} where x_{seg} is the binary segmentation map of x provided from the dataset. We use $\overline{x_{\mathrm{seg}}}$ to mask foreground for x and $\hat{x}$.

5.4 Qualitative Evaluation

Figures 4 and 5 illustrate some examples of the results obtained by our method (with $p = 0.8$) and Dong+ [1]. They show that the synthesized images by our method match the text descriptions more precisely than Dong+ [1] while successfully retaining background of the source images.

On the Caltech-200 dataset, we see that the results by our method are clearer in foreground and background with less noise than Dong+ [1] (Fig. 4). Though foreground of the results by Dong+ [1] also matches the text descriptions (not always though), we observe that background is not preserved well.

On the Oxford-102 dataset, on the other hand, we see that our method and Dong+ [1] both have some failures in synthesizing images (red rectangles in Fig. 5). This is because images in the dataset are too complex; for example, the detail of flowers such as a stamen is too small. Nevertheless, we still observe that our method outperforms Dong+ [1]. We note that Dong+ [1] generated different flowers from the source images (blue rectangles in Fig. 5).

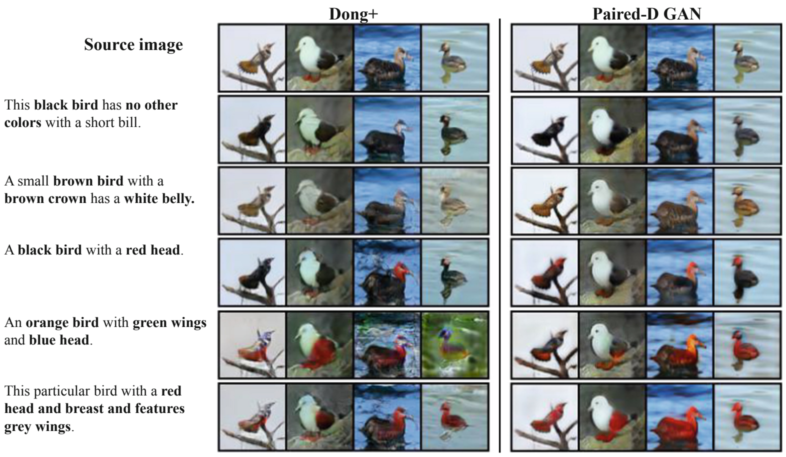

Dong+
Paired-D GAN
Source image
This black bird has no other colors with a short bill.
A small brown bird with a brown crown has a white belly.
A black bird with a red head.
An orange bird with green wings and blue head.
This particular bird with a red head and breast and features grey wings.

Fig. 4. Visual comparison of our method against Dong+ [1] on the Caltech-200 bird dataset [13]. First row: source images, most left column: text descriptions. Each image is generated using a source image and a text description.

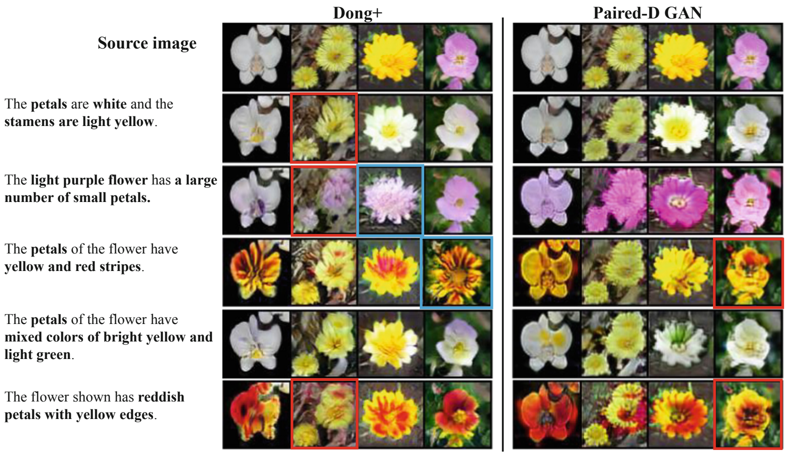

Dong+
Paired-D GAN
Source image
The petals are white and the stamens are light yellow.
The light purple flower has a large number of small petals.
The petals of the flower have yellow and red stripes.
The petals of the flower have mixed colors of bright yellow and light green.
The flower shown has reddish petals with yellow edges.

Fig. 5. Visual comparison of our method against Dong+ [1] on Oxford-102 flower dataset [14]. First row: source image, most left column: text descriptions. Each image is generated using its source image and text. The red rectangles indicate the failure synthesized images in both Dong+ [1] and ours. The blue rectangles indicate the generated images different from their source images. (Color figure online)

5.5 Quantitative Evaluation

For the quantitative evaluation, we computed IS, FGS, and BGS of the synthesized images, which are shown in Table 2. To compute IS, we iterated 10 times the experiment that we synthesize 8000 images, and computed the average and the standard deviation of the resulting scores, as recommended in [39]. For FGS

Table 2. Quantitative comparison using IS (larger is better), FGS, and BGS (smaller is better). The best results are given in blue.

Dataset	Caltech-200 [13]			Oxford-102 [14]		
Metric	$IS \Uparrow$	$FGS \Downarrow$	$BGS \Downarrow$	$IS \Uparrow$	$FGS \Downarrow$	$BGS \Downarrow$
Paired-D GAN	6.39 ± 0.18	17.26 ± 0.21	9.03 ± 0.06	4.41 ± 0.08	8.81 ± 0.08	8.87 ± 0.04
Dong+ [1]	5.56 ± 0.14	18.60 ± 0.09	11.83 ± 0.06	4.03 ± 0.11	9.71 ± 0.11	9.47 ± 0.14
Yang+ [15]	5.92 ± 1.04	18.34 ± 0.14	–	3.49 ± 0.04	10.32 ± 0.09	–

and BGS, we iterated 5 times the experiment that we synthesize 600 images, and computed the average and the standard deviation of the resulting scores. Note that we cannot compute BGS for Yang+ [15] because no ground-truths of background images exit for Yang+ [15]. We also remark that we used the visual-text shared-space model [25] pre-trained on the Caltech-200 (or Oxford-102) dataset to compute features for FGS.

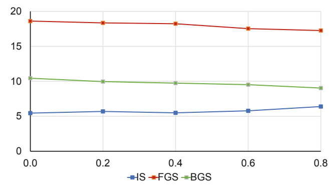
Table 2 shows that our method achieves the best performances in all the metrics, meaning that the images synthesized by our method are superior not only in the overall quality (IS) but also in foreground-text matching (FGS) and in background preservation (BGS). The outperformance of our method against Dong+ [1] in all the metrics confirms that evaluating foreground and background separately in the training phase is effective. Compared to Yang+ [15], we see that our method and Dong+ [1] generate more realistic image, suggesting that for semantic image synthesis, generating foreground and background at the same time is better than separately and recursively generating foreground and background.

5.6 Detailed Analysis

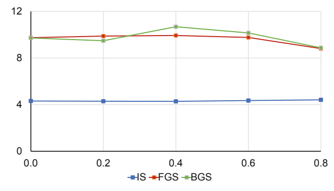
First of all, we evaluated the effectiveness of employing D_{BG} through comparing our complete model with models using D_{FG} only or D_{BG} only. As shown in Table 3, the method using D_{FG} only achieves FGS best, and the method using D_{BG} only achieves BGS best. This means that the method using D_{FG} is correctly tuned to the foreground while the method using D_{BG} is correctly tuned to the background, and that D_{FG} and D_{BG} properly work for foreground and background each. Our completed method, on the other hand, balances foreground and background well as it achieves IS best.

Table 3. Evaluation on the effectiveness of employing D_{BG}.

Dataset	Caltech-200			Oxford-102		
Metric	$IS \Uparrow$	$FGS \Downarrow$	$BGS \Downarrow$	$IS \Uparrow$	$FGS \Downarrow$	$BGS \Downarrow$
Complete model ($D_{FG} + D_{BG}$)	6.39 ± 0.18	17.26 ± 0.21	9.03 ± 0.06	4.41 ± 0.08	8.81 ± 0.08	8.87 ± 0.04
Model with D_{FG} only	5.83 ± 0.19	16.74 ± 0.12	11.89 ± 0.08	4.21 ± 0.07	8.52 ± 0.13	10.02 ± 0.08
Model with D_{BG} only	6.02 ± 0.15	20.33 ± 0.11	7.63 ± 0.08	4.24 ± 0.10	10.68 ± 0.14	8.32 ± 0.15



(a) Caltech-200 dataset.



(b) Oxford-102 dataset.

Fig. 6. Quantitative comparison by changing p by 0.2 from 0.0 to 0.8.

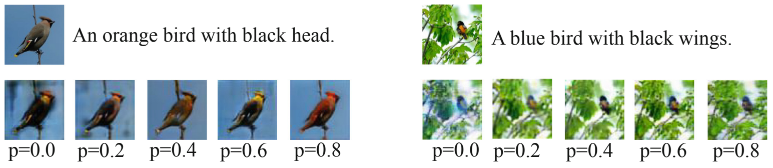

An orange bird with black head.
A blue bird with black wings.
p=0.0
p=0.2
p=0.4
p=0.6
p=0.8
p=0.0
p=0.2
p=0.4
p=0.6
p=0.8

Fig. 7. Zero-shot results by changing p by 0.2 from 0.0 to 0.8. The source image has simple background (right) or complex background (left).

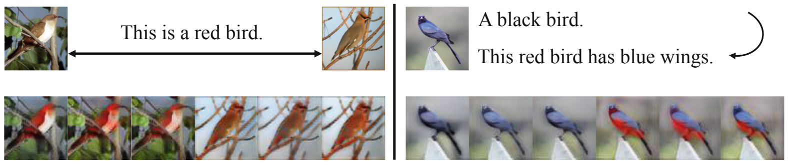


Fig. 8. Zero-shot results of interpolation. Left: interpolation between two source images with the same target text description. Right: interpolation between two target text descriptions for the same source image.

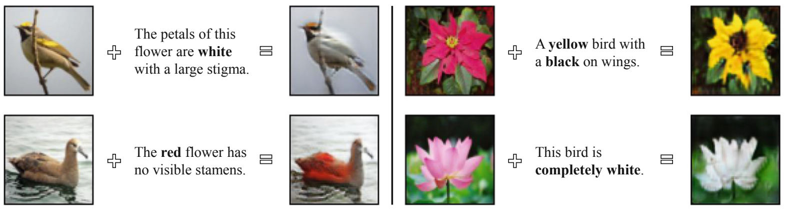

The petals of this flower are **white** with a large stigma.
The **red** flower has no visible stamens.
A **yellow** bird with a **black** on wings.
This bird is **completely white**.

Fig. 9. Zero-shot results from a source image and text descriptions that are not related to each other, showing the effectiveness of foreground and background discriminators.

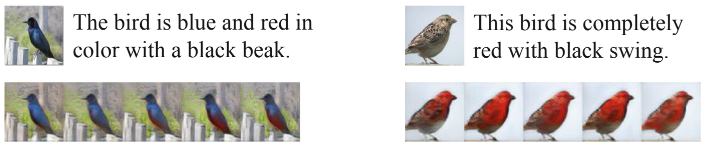

The bird is blue and red in color with a black beak.
This bird is completely red with black swing.

Fig. 10. Zero-shot results from the same source image and text descriptions, showing variety of foregrounds.

Then, we evaluated the impact on the results by different p's (the probability of generating zero at each pixel) used in creating the binary filter for the foreground discriminator D_{FG}. We changed p by 0.2 from 0.0 (no mask) to 0.8 and computed IS, BGS, FGS at each p (Fig. 6). Visual comparison with different p's are illustrated in Fig. 7 (two examples with simple/complex background). Figure 6 indicates that all the metrics become better at $p = 0.8$ (80% in probability of a source image are masked to focus on foreground). The explanation for this can be as follows, which is also supported by Fig. 7. When $p = 0.0$ (no mask), D_{FG} accesses the whole source image in the training phase, affecting background of generated images. By increasing p, D_{FG} is likely to focus on only foreground, leading to improving the quality of generated images. We note that background discriminator D_{BG} also succeeds in maintaining background of the source image (we can see that the background is kept well in most cases).

We next demonstrated the smooth interpolation between the source image and the target image. Figure 8 show synthesized images obtained by the linear interpolation between the source and the target images. In Fig. 8 (left), we interpolated two source images with a fixed text description. In contrast, we keep the source image fixed while changing text descriptions in Fig. 8 (right). These results indicate that our method is capable of independently interpolating between source images and text descriptions. We remark that our method preserve background well regardless of interpolation.

Figure 9 shows the generated images obtained using source images from the Caltech-200 [13] dataset with text descriptions from the Oxford-102 [14] dataset (not used in training phase), and vice verse. Figure 9 shows that our model retains background of source images and changes only foreground to match text descriptions (e.g. color) even if they are not used in the training (regardless of untrained text descriptions). This illustrates the flexible capability of our model to disentangle foreground and background.

We also show in Fig. 10 the effectiveness of text embedding augmentation [7] in our method to synthesize various images using the same source image and text descriptions.

6 Conclusion

We proposed Paired-D GAN conditioned on both text descriptions and images for semantic image synthesis. Paired-D GAN consists of one generator and two discriminators with different architectures where one discriminator is used for judging foreground and the other is for judging background. Our method is able to synthesize a realistic image where an input text description matches its corresponding part (foreground) of the image while preserving background of a given source image. Experimental results on the Caltech-200 and the Oxford-102 datasets demonstrate the effectiveness of our method.

References

1. Dong, H., Yu, S., Wu, C., Guo, Y.: Semantic image synthesis via adversarial learning. In: ICCV (2017)
2. Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., Lee, H.: Generative adversarial text-to-image synthesis. In: ICML (2016)
3. Yan, X., Yang, J., Sohn, K., Lee, H.: Attribute2Image: conditional image generation from visual attributes. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9908, pp. 776–791. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46493-0_47
4. Efros, A.A., Freeman, W.T.: Image quilting for texture synthesis and transfer. In: SIGGRAPH (2001)
5. Gatys, L.A., Ecker, A.S., Bethge, M.: Image style transfer using convolutional neural networks. In: CVPR (2016)
6. Goodfellow, I., et al.: Generative adversarial nets. In: NIPS (2014)
7. Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., Metaxas, D.: StackGAN: text to photo-realistic image synthesis with stacked generative adversarial networks. In: ICCV (2017)
8. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: CVPR (2017)
9. Wang, X., Gupta, A.: Generative image modeling using style and structure adversarial networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9908, pp. 318–335. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46493-0_20
10. Ledig, C., et al.: Photo-realistic single image super-resolution using a generative adversarial network. In: CVPR (2017)
11. Zhang, H., et al.: StackGAN++: realistic image synthesis with stacked generative adversarial networks. arXiv: 1710.10916 (2017). (IEEE TPAMI, to appear)
12. Nguyen, T., Le, T., Vu, H., Phung, D.: Dual discriminator generative adversarial nets. In: NIPS (2017)
13. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The Caltech-UCSD Birds-200-2011 Dataset. California Institute of Technology, Technical report CNS-TR-2011-001 (2011)
14. Nilsback, M.-E., Zisserman, A.: Automated flower classification over a large number of classes. In: Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing (2008)
15. Yang, J., Kannan, A., Batra, D., Parikh, D.: LR-GAN: layered recursive generative adversarial networks for image generation. In: ICLR (2017)
16. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. In: ICLR (2014)
17. Rezende, D.J., Mohamed, S., Wierstra, D.: Stochastic backpropagation and approximate inference in deep generative models. In: ICML (2014)
18. Van Den Oord, A., Kalchbrenner, N., Kavukcuoglu, K.: Pixel recurrent neural networks. In: ICML (2016)
19. Van den Oord, A., Kalchbrenner, N., Vinyals, O., Espeholt, L., Graves, A., Kavukcuoglu, K.: Conditional image generation with pixelcnn decoders. In: NIPS (2016)
20. Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., Abbeel, P.: InfoGAN: interpretable representation learning by information maximizing generative adversarial nets. In: NIPS (2016)

21. Taigman, Y., Polyak, A., Wolf, L.: Unsupervised cross-domain image generation. In: ICLR (2017)
22. Zhu, J.-Y., et al.: Toward multimodal image-to-image translation. In: NIPS (2017)
23. Perarnau, G., van de Weijer, J., Raducanu, B., Álvarez, J.M.: Invertible conditional GANs for image editing. In: NIPS Workshop on Adversarial Training (2016)
24. Li, C., Wand, M.: Precomputed real-time texture synthesis with Markovian generative adversarial networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9907, pp. 702–716. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46487-9_43
25. Reed, S., Akata, Z., Lee, H., Schiele, B.: Learning deep representations of fine-grained visual descriptions. In: CVPR (2016)
26. Reed, S., Akata, Z., Mohan, S., Tenka, S., Schiele, B., Lee, H.: Learning what and where to draw. In: NIPS (2016)
27. Ha, M.L., Franchi, G., Moller, M., Kolb, A., Blanz, V.: Segmentation and shape extraction from convolutional neural networks. In: WACV (2018)
28. Wu, R., Li, X., Yang, B.: Identifying computer generated graphics via histogram features. In: ICIP (2011)
29. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR (2015)
30. Russakovsky, O., et al.: ImageNet large scale visual recognition challenge. Int. J. Comput. Vis. **115**, 211–252 (2015)
31. Nair, V., Hinton, G.E.: Rectified linear units improve restricted Boltzmann machines. In: ICML (2010)
32. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
33. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: ICML (2015)
34. Ashish Bora, A.D., Price, E.: AmbientGAN: generative models from lossy measurements. In: ICLR (2018)
35. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: CVPR (2005)
36. https://github.com/woozzu/dong_iccv_2017. Accessed 01 June 2018
37. https://github.com/jwyang/lr-gan.pytorch. Accessed 01 June 2018
38. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: ICLR (2015)
39. Salimans, T., et al.: Improved techniques for training GANs. In: NIPS (2016)
40. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: CVPR (2016)