The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19)

Perceptual Pyramid Adversarial Networks for Text-to-Image Synthesis

Lianli Gao,[1] Daiyuan Chen,[1] Jingkuan Song,[1]* Xing Xu,[1] Dongxiang Zhang,[1] Heng Tao Shen[1]
[1]Center for Future Media and School of Computer Science and Engineering,
University of Electronic Science and Technology of China,
Chengdu 611731, China.
{lianli.gao,xing.xu,zhangdo}@uestc.edu.cn, {yayeochen95,jingkuan.song}@gmail.com, shenhengtao@hotmail.com

Abstract

Generating photo-realistic images conditioned on semantic text descriptions is a challenging task in computer vision field. Due to the nature of hierarchical representations learned in CNN, it is intuitive to utilize richer convolutional features to improve text-to-image synthesis. In this paper, we propose Perceptual Pyramid Adversarial Network (PPAN) to directly synthesize multi-scale images conditioned on texts in an adversarial way. Specifically, we design one pyramid generator and three independent discriminators to synthesize and regularize multi-scale photo-realistic images in one feed-forward process. At each pyramid level, our method takes coarse-resolution features as input, synthesizes high-resolution images, and uses convolutions for up-sampling to finer level. Furthermore, the generator adopts the perceptual loss to enforce semantic similarity between the synthesized image and the ground truth, while a multi-purpose discriminator encourages semantic consistency, image fidelity and class invariance. Experimental results show that our PPAN sets new records for text-to-image synthesis on two benchmark datasets: CUB (i.e., 4.38 Inception Score and .290 Visual-semantic Similarity) and Oxford-102 (i.e., 3.52 Inception Score and .297 Visual-semantic Similarity).

Introduction

Recently, we have witnessed a breakthrough in the application of deep learning to generate textual descriptions conditioned on images/videos (Song et al. 2017; Gao et al. 2017). On the other hand, text-to-image synthesis is the reverse problem: generating photo-realistic images that match the given text descriptions. However, there are very few researches on this task. From a high-level perspective, both tasks are similar. Nevertheless, these problems are entirely different because text-to-image and image-to-text conversions are highly different cross-modal problems (Bodnar 2018). Particularly, text-to-image synthesis requires the synthesized images to be not only photo-realistic but also semantically consistent, and this task has many practical applications such as photo editing or multimedia data creation.

The task of text-to-image synthesis is exactly what generative models attempt to solve, and most of the recent progresses are obtained by Generative Adversarial Networks (GANs) (Goodfellow et al. 2014). GANs consist of one generator and one discriminator. The generator is designed to generate as realistic images as possible to fool the discriminator, and the discriminator is optimized to distinguish between fake generated images and real ground-truth images. Recently, GANs have been applied to various applications, especially in simulating complicated data distributions, like images (Song et al. 2018a; 2018b), music, texts (Gupta et al. 2018) and video (Li et al. 2018). More recently, GANs have been used to generate photo-realistic fine-grained images conditioned on semantic texts. This task demands the network to learn a precise mapping from semantic text distribution to visual image distribution. Meanwhile, the network, as a generative model, is requested to synthesize various and natural images that match text descriptions, not just like the ground-truth images in pixel space.

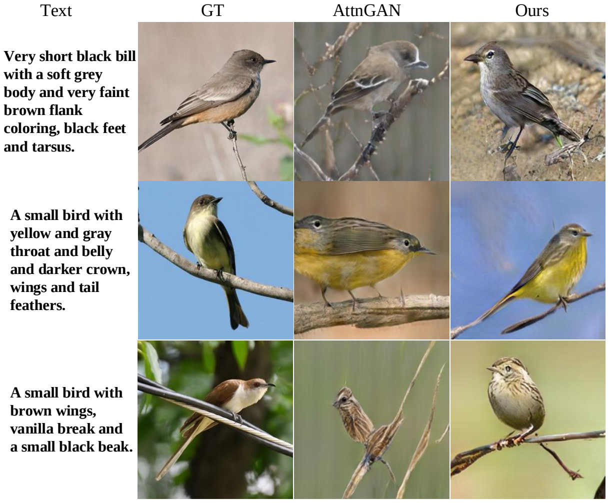

Text
GT
AttnGAN
Ours
Very short black bill with a soft grey body and very faint brown flank coloring, black feet and tarsus.
A small bird with yellow and gray throat and belly and darker crown, wings and tail feathers.
A small bird with brown wings, vanilla break and a small black beak.

Figure 1: Some experimental results of our method (column 3) for text-to-image synthesis on CUB dataset, and the comparison with current state-of-the-art method AttnGAN (column 2). Each sample shows the input text description and generated 256×256 images. Our generated images are more realistic and clear. Zoom-in for better observation.

As a pioneering work on text-to-image synthesis, GAN-INT-CLS (Reed et al. 2016b), firstly introduces a vanilla

*Jingkuan Song is the corresponding author.

GAN to bridge the gap between fine-grained texts space and visual images space. However, this method synthesizes only 64×64 images, which are very blurred not to maintain vivid object details. Based on this work, StackGAN (Zhang et al. 2017a), StackGAN++ (Zhang et al. 2017b) and AttnGAN (Xu et al. 2017) are successively proposed. StackGAN decomposes this task into two sub-problems through a stage-by-stage process. The first-stage GAN in StackGAN generates a blurred 64×64 image with primitive colors and shapes, then, with fixed this stage, the second-stage GAN takes the low-resolution image and text to generate high-resolution images. So StackGAN consists of two separated GANs, which is difficult to train and unstable to evaluate (Huang, Yu, and Wang 2018). Different from StackGAN, StackGAN++ is composed of three pairs of generators and discriminators arranged in a tree-like struture. Furthermore, StackGAN++ can generate 64×64, 128×128 and 256×256 images from different branches of the tree. So StackGAN++ jointly approximate multi-scale image distributions. But three pairs also make StackGAN++ complicated not to converge like StackGAN. Moreover, AttnGAN further extends the architecture of StackGAN++ by adopting attention mechanism over images and texts. And AttnGAN first embeds each sentence into a global sentence feature and multiple local word features, then uses the global feature to generate blurred 64×64 images, lastly takes the local features to progressively generate 128×128 and 256×256 images. Compared with StackGAN++, the structure of AttnGAN is more complex and it is not end-to-end. Different from above methods, which only take text features, TAC-GAN (Dash et al. 2017), as a combination of GAN-INT-CLS (Reed et al. 2016b) and AC-GAN (Odena, Olah, and Shlens 2016), takes additional image class labels to increase image diversity. Compared with all above networks, which generate single-resolution images or multi-resolution images stage-by-stage, our end-to-end PPAN introduces only **one** generator and **three** discriminators to synthesize directly multiple-resolution images. Furthermore, the generator of PPAN applies pyramid framework to enhance multi-scale feature representations and employs a perceptual loss to guarantee the image diversity. And the discriminators of PPAN use matching-aware pair losses (Reed et al. 2016b) and local image losses (Zhang, Xie, and Yang 2018) to assure semantic consistency. Moreover, the discriminator for 256×256 resolution images adopts an additional class information loss (Dash et al. 2017) to attain class invariance.

Leveraging pyramid framework in CNN enriches effectively multi-scale feature representations on computer vision tasks, such as semantic segmentation, object detection (Lin et al. 2017) and super-resolution (Lai et al. 2017). To tackle these tasks, the networks build a down-to-top pathway and lateral connections to combine low-resolution, semantically strong features with high-resolution, semantically weak features. Consequently, the networks have rich semantics at all levels and can be built quickly from a single input image scale. In our method, PPAN combines features from 32×32 to 128×128 via three Cumulative Blocks in Fig. 2(b) to enrich semantics at all scales. However, as the generator structure of PPAN gets deeper and deeper, the features, which are learned at hidden layers, are not always “transparent” in their meaning and show reduced discriminability. To solve these problems and stabilize the training process, PPAN adopts deep supervision information (Zhang, Xie, and Yang 2018) for hidden layers. Specifically, PPAN regularizes multi-scale hidden features with deep adversarial supervision, which is from three independent discriminators, to encourage the generator to model multi-scale image distributions.

To guarantee generated image diversity, we adopt a perceptual loss for the generator to attain perceptual similarity not only pixel-level similarity. Specifically, we train the generator using a perceptual loss (Johnson, Alahi, and Fei-Fei 2016) based on semantic high-level features extracted from a pre-trained VGG16 network (Simonyan and Zisserman 2014), rather than using a pixel-level loss depending only on low-level features.Furthermore, the perceptual loss measures image similarities more robustly than pixel-level losses during training, and runs more quickly during testing (Johnson, Alahi, and Fei-Fei 2016). In this paper, we propose a novel end-to-end GAN network that can generate once multi-resolution photo-realistic images conditioned on text descriptions.

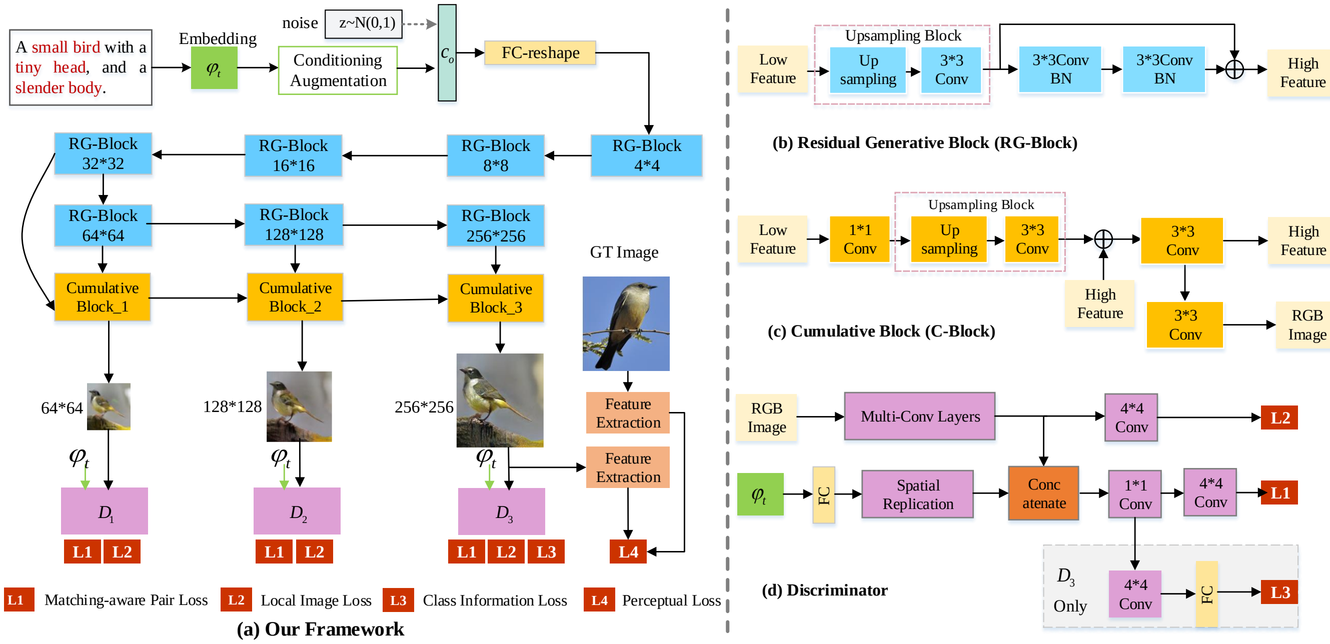
Our major contributions can be summarized as followings: 1) We propose Perceptual Pyramid Adversarial Network (PPAN) for text-to-image synthesis task, by directly generating multi-scale images conditioned on texts in an adversarial way. Instead of using multiple stages or multiple GANs, our PPAN has one generator and three independent discriminators, to synthesize and regularize multi-scale photo-realistic images. At each pyramid level, PPAN utilizes coarse-resolution features to synthesize the high-resolution images and finer-level feature maps. 2) We define perceptual loss on the generator to obtain diverse images and design multi-purpose discriminators to encourage semantic consistency, image fidelity and class invariance. 3) Extensive experimental results are conducted, and our PPAN sets new records for text-to-image synthesis on two benchmark datasets: CUB (i.e., 4.38 Inception Score and .290 Visual-semantic Similarity) and Oxford-102 (i.e., 3.52 Inception Score and .297 Visual-semantic Similarity).

Perceptual Generative Pyramid Network

Network Architecture

As shown in Figure 2, PPAN consists of three components: *Conditioning Augmentation, one Generator and three Discriminators*. Due to the limited training data, we first use Conditioning Augmentation to generate more training pairs. Then a generator is designed to synthesize multi-scale images conditioned on the input text features. Three discriminators are designed to regularize output images of the generator at different pyramid. We describe each of them as well as the loss functions defined on each component in details in the remainder of this section.

Conditioning Augmentation In Fig. 2, a text description t is embedded to an 1024-dim vector φ_t by an encoder (Reed et al. 2016a). However, this high dimension usually causes discontinuity in the latent condition manifold, and this discontinuity is not desirable for training the generator.



(a) Our Framework

(b) Residual Generative Block (RG-Block)

(c) Cumulative Block (C-Block)

(d) Discriminator

Figure 2: The framework of PPAN, which consists of three key components: (1) a conditioning augmentation module, for producing additional condition c_0 (2) a generator consisting of Residual Generative Blocks (RG-Blocks) and Cumulative Blocks (C-Blocks), for synthesizing images with three resolution and (3) three discriminators, for regularizing the generated images.

To mitigate this issue, we use a Conditioning Augmentation(CA) (Zhang et al. 2017a) to properly reduce the text dimension and produce more training pairs given a small amount of image-text training pairs, so CA encourages robustness to small perturbations along the condition manifold. Consequently, CA produces conditioning variable c_0. Instead of using once-sampling variable for training and testing (Zhang et al. 2017a; 2017b; Zhang, Xie, and Yang 2018), we randomly take **once** sampling to get the condition variable c_0 from an independent Gaussian distribution $N(\mu(\varphi_t), \Sigma(\varphi_t))$ during training, where the mean $\mu(\varphi_t)$ and diagonal covariance matrix $\Sigma(\varphi_t)$ are functions of the text embedding φ_t. And we get condition c_0 by computing:

$$c_0 = \mu(\varphi_t) + \delta(\varphi_t) \odot \varepsilon \qquad (1)$$

where $\delta(\varphi_t)$ are the values in the diagonal of $\Sigma(\varphi_t)$, and $\odot$ is the element-wise multiplication and $\varepsilon \sim N(0, I)$. Furthermore, we randomly take **n times** sampling to get condition c_n during testing, which is computed by:

$$c_n = \frac{1}{n}\sum\nolimits_{i=1}^{n} c_i \qquad (2)$$

where c_i indicates i-th sampling.

Meanwhile, to further enforce the smoothness over the condition manifold and avoid over-fitting (Doersch 2016), we add the following regularization to the objective of the generator during training:

$$L_{kl} = D_{KL}(\mu(\varphi_t), \Sigma(\varphi_t)) \,||\, N(0, I)) \qquad (3)$$

where D_{KL} is the Kullback-Leibler divergence (KL divergence) between the standard Gaussian distribution and the conditioning Gaussian distribution.

Generator In Fig. 2, the generator consists of two components: Residual Generative Blocks (RG-Blocks) and Cumulative Blocks (C-Blocks). The RG-Blocks intend to capture the textual features and the C-Blocks are designed to generate the visual features.

As can be seen from Fig. 2, when sampling from $N(\mu(\varphi_t), \Sigma(\varphi_t))$ by CA, we get an 128-dim condition c_0. And we concatenate it to an 100-dim noise vector, then reshape this feature to the shape of (batch-size, 4, 4, 1024). Afterwards this feature is fed consecutively into seven RG-Blocks. As shown in Fig. 2(b), a RG-Block is a combination of an Upsampling Block and a modified residual block (He et al. 2016). The Upsampling Block is composed of an upsampling layer and a 3×3 convolutional (conv) layer instead of a deconv layer to avoid "Checkerboard Artifacts" (Odena, Dumoulin, and Olah 2016). Consequently, A RG-Block consists of an upsampling layer and 3 conv layers with batch normalization layers (BN) (Ioffe and Szegedy 2015) and ReLu to output a high-resolution feature.

To fully use multi-scale features, the outputs of the 5-th to 7-th RG-Block are fed into C-Blocks to synthesize high-resolution images (64×64, 128×128, 256×256) and finer-level features. As shown in Fig. 2(c), each C-Block consists of an 1×1 conv layer, an Upsampling Block and two 3×3 conv layers. Afterwards we take images of three resolution into corresponding discriminators, which regularize the hidden features of the generator with adversarial supervision.

Comparing with previous networks (Zhang et al. 2017a; 2017b; Xu et al. 2017), which use multiple GANs to synthesize images, we play the adversarial game along the depth of the generator $G(z, c_0)$ and jointly train one generator and three discriminators. Consequently, the $G(z, c_0)$ produces three resolution images in a feed-forward pass:

$$X_i = G(z, c_0) \qquad i = 1, 2, 3 \tag{4}$$

where X_i is the i-th synthesized images with the resolution of $64 \times 64, 128 \times 128$ and 256×256, and z is an 100-dim random noise vector.

Perceptual Loss To synthesize better photo-realistic images, the definition of image reconstruction losses is very critical, which steers the whole optimization of generating images. Thus we use a *perceptual loss* (L4 in Fig. 2) to attain perceptual similarity between a generated image and ground-truth image of 256×256 resolution. Specifically, we follow the VGG loss (Ledig et al. 2017) based on the ReLU activation feature of the pre-trained VGG16 network (Simonyan and Zisserman 2014). This feature $\phi_{i,j}(\cdot)$ is extracted from the j-th conv (after activation) before the i-th max-pooling layer in the VGG16. Then we define this loss using the Euclidean distance between a generated image $\phi_{i,j}(X_3)$ and a ground-truth image $\phi_{i,j}(I_3)$:

$$L_4 = \frac{1}{W_{i,j} H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} (\phi_{i,j}(I_3)_{x,y} - \phi_{i,j}(X_3)_{x,y})^2 \tag{5}$$

where $W_{i,j}$ and $H_{i,j}$ indicate the dimensions of the feature within the VGG16 network.

The method (Johnson, Alahi, and Fei-Fei 2016) observes that a generated image minimizing the image reconstruction loss for lower and higher conv layers tends to be visually indistinguishable from the real image. Therefore, we use the intermediate feature $\phi_{2,2}$, which is extracted from the second conv block before the second maxpooling layer in VGG16, to preserve color, texture, and exact shape information.

Multi-purpose Discriminator As shown in Fig. 2, there are three multi-purpose discriminators (i.e., D_1, D_2 and D_3) defined on the synthesized images and text descriptions. For each discriminator, we design different branches to achieve different purposes.

The first branch calculates the *matching-aware pair loss* (L_1 in Fig. 2) (Zhang et al. 2017a) to encourage semantic consistency between the texts and generated images. This branch concatenates an image feature (batch-size, 4, 4, 512) and a text feature (batch-size, 4, 4, 128), spatially replicated from a text embedding φ_t. Then it uses an 1×1 conv to fuse two features together and a 4×4 conv to classify a generated image to be real or fake. The second branch computes the *local image loss* (L_2 in Fig. 2) (Zhang, Xie, and Yang 2018) to guarantee image fidelity. It takes an image feature as input, produces a $H_i \times H_i$ 2D probability map O_i and classifies every location as real or fake. Moreover, we manage H_i accordingly to tune the receptive field of each element in O_i, and we set $H_1 = 1, H_2 = 1$ for 64×64 and 128×128 resolution, $H_3 = 5$ for 256×256 resolution. The third branch calculates the *class information loss* (L_3 in Fig. 2) (Dash et al. 2017) to ensure class invariance. It shares the fused feature after 1×1 conv layer with the first branch. Then this branch uses a 4×4 conv layer and a fully connected layer to produce a feature (batch-size, C) to classify the object in the image, where C indicates the number of object classes in the dataset. And we set $C = 200$ for CUB dataset and $C = 102$ for Oxford-102.

We equip the D_1, D_2 and D_3 with L_1 and L_2 losses, and we further add L_3 loss to D_3. The reason for only applying L_3 loss to D_3 is that higher resolution images have higher potential to compute more accurate classification scores.

Multi-purpose Losses We define the multi-purpose losses of three main categories: 1) matching-aware pair losses L_1; 2) local image losses L_2; and 3) a class information loss L_3, to encourage semantic consistency, image fidelity and class invariance respectively. Furthermore, we adopt the LSGANs (Mao et al. 2017) as our fundamental formulations to remedy the "Vanishing Gradient" problem in training GANs.

The matching-aware pair loss L_1 (Reed et al. 2016b), one of the adversarial losses, is designed to guarantee the global semantic consistency. And during training discriminators, this loss takes three kinds of image-text pairs: 1)the pair of a real image I_i and a matching text description t, which serves as a positive sample pair (I_i, t); 2)a real image I_i' and a mismatching text description t, which serves as a negative sample pair (I_i', t); 3)a generated image X_i and a matching text description t, which serves also as other negative sample pair (X_i, t). And the matching-aware pair loss is defined as followings:

$$L_1 = \sum\nolimits_{i=1}^{3} ((D_i^1(I_i, t) - \mathbb{I})^2 + D_i^1(I_i', t)^2 + D_i^1(X_i, t)^2) \tag{6}$$

where D_i^1 is the matching-aware pair loss function for the i-th discriminator, and $\mathbb{I} \in \mathbb{R}^d$ is a vector of ones.

As the resolution goes higher, it becomes more burdensome for the matching-aware pair losses to capture the local fine-grained details. Furthermore, the matching-aware pair losses may over-emphasize certain biased local features and lead to make artifacts (Shrivastava et al. 2017). So to alleviate these issues and guarantee images smooth and natural, we adopt the local image losses L_2 (Zhang, Xie, and Yang 2018), as the other one of the adversarial losses, to guide the discriminator to differentiate real or fake image patches to focus on local image details. It is defined as:

$$L_2 = \sum\nolimits_{i=1}^{3} ((D_i^2(I_i) - \mathbb{E})^2 + (D_i^2(I_i') - \mathbb{E})^2 + D_i^2(X_i)^2) \tag{7}$$

where D_i^2 is the local image loss function for the i-th discriminator and $\mathbb{E}$ indicates the 2D matrix of ones with the shape of (batch-size, H_i).

Furthermore, to diversify the generated images and improve their structural coherence, we provide the discriminator D_3 with more discriminative information of class labels (Dash et al. 2017). And we optimize the class information loss L_3 based on a sum of the binary cross entropy as a classification loss for 256×256 resolution. This loss is similar to the matching-aware pair loss, but takes three kinds of image-text-class pairs. Inside, the discriminator D_3 takes a pair set $\{(I_3, t, C), (I_3', t, C'), (X_3, t, C)\}$, which uses C and C' to respectively indicate a right and wrong class label according to an image. The pair set consists of three tuples, which contains an image, a corresponding text description and class label. Notice that I_3' is mismatching to the text description t

but matching to the class label C'. And the class information loss is defined as followings:

$$L_3 = log(D_3^3(I_3, t, C)) + log(D_3^3(I'_3, t, C')) + log(D_3^3(X_3, t, C)) \quad (8)$$

where D_3^3 is the class information loss function for the third discriminator.

Optimization

Using the defined loss functions 3, 5, 6, 7 and 8, we train our network PPAN. The forward propagation is as below. First, we embed a text to a vector φ_t using a pre-trained encoder and then adopt CA to output conditioning variable c_0:

$$c_0 = CA(\varphi_t; \theta) \quad (9)$$

where θ stands for the parameters of CA. Then the feature concatenating c_0 and noise z is the input for the generator G to synthesize images X:

$$X = G(z, c_0; \pi) \quad (10)$$

where π stands for the parameters of the generator. Finally, discriminators D calculates the *loss*:

$$loss = D(I, X, \varphi_t; \psi) \quad (11)$$

where I is the ground truth and ψ stands for the parameters of discriminators.

In the network, we have parameters of θ, π, ψ to learn. We use back propagation to learn and Adam (Kingma and Ba 2014) to minimize the loss. Particularly, we initialize network parameters with normal distribution and use forward propagation to obtain the value of (L_{kl}, L_1, L_2, L_3, L_4). In each iteration, we sample a mini-batch of text descriptions from the training set then update each parameter:

$$\theta \leftarrow \theta - \tau(L_{kl}) \quad (12)$$

$$\pi \leftarrow \pi - \tau \bigtriangledown_\pi (\lambda_1 L_1 + \lambda_2 L_2 + \lambda_3 L_3 + \lambda_4 L_{kl}) \quad (13)$$

$$\psi \leftarrow \psi - \tau \bigtriangledown_\psi (\beta_1 L_1 + \beta_2 L_2 + \beta_3 L_3) \quad (14)$$

where τ is the learning rate. We train our network until it converges.

Experiments

We evaluate our Perceptual Pyramid Adversarial Network (PPAN) for the task of text-to-image synthesis on two datasets. Specifically, the experiments are designed to study the following research questions of our methods:

Q1: What is the quantitative performance of PPAN compared with the state-of-the-arts?

Q2: What is the qualitative performance of PPAN compared with the state-of-the-arts?

Q3: How does each component of PPAN affect the performance?

Settings

Datasets Following the experimental setup in (Reed et al. 2016b; Zhang et al. 2017a), we pre-process all images to ensure that bounding boxes of objects have greater-than-0.75 object-image size ratios. And we split CUB (Wah et al. 2011) and Oxford-102 (Nilsback and Zisserman 2008) datasets into class-disjoint training and testing sets. The **CUB** dataset (Wah et al. 2011) contains 8855 and 2933 images for training and testing, totally belonging to 200 categories. The **Oxford-102** dataset (Nilsback and Zisserman 2008) consists of 7034 and 1155 images for training and testing, a total of 102 categories. Each image in both datasets is provided 10 text descriptions by (Reed et al. 2016b). Because both datasets are originally used for the fine-grained classification task, text descriptions are detailed and precise about the single object, not about backgrounds. Moreover, we employ the pre-trained char-RNN text encoder provided by (Reed et al. 2016a) to encode each sentence into an 1024-dim text embedding vector.

Evaluation Metric We use two quantitative metrics to evaluate our method.

1) Inception Score (IS) is a measurement of both objectiveness and diversity of images. The intuition behind IS is that good generative models should synthesize diverse but meaningful images. Therefore, the KL divergence between the marginal distribution $p(y)$ and the conditional distribution $p(y|x)$ should be large. So IS is defined as followings:

$$IS = exp(\mathbb{E}_x(D_{KL}(p(y|x) \,||\, p(y)))) \quad (15)$$

In our experiments, we adopt the fine-tune Inception models (Szegedy et al. 2017) on the training sets of the CUB and Oxford-102 datasets, provided by (Zhang et al. 2017a).

2) Visual-semantic Similarity(VS) is proposed by (Zhang, Xie, and Yang 2018) to measure the alignment between synthesized images and the conditioned text. So VS is computed as followings:

$$VS = \frac{f_t(t) \cdot f_x(x)}{\|f_t(t)\|_2 \cdot \|f_x(x)\|_2} \quad (16)$$

where the text encoder $f_t(\cdot)$ and the image encoder $f_x(\cdot)$ are learned to map both texts t and images x into a common space in $\mathbb{R}^{512}$. Higher score indicates better performance.

However, neither IS nor VS assesses realism of details and intra-class diversity. Therefore we also visualize some synthesized images for qualitative evaluation.

Implementation Details By default, we set $\lambda_1 = \beta_1 = \lambda_2 = \beta_2 = 1$, $\lambda_3 = 1e-07$, and $\beta_3 = 100$ for all datasets, $\lambda_4 = 4$ for CUB and Oxford-102 dataset. Furthermore, when computing IS for the CUB dataset, we take 29,330 randomly selected samples for each resolution, which are from 1 test real image (2,933 in total) to 10 generated images based on 1 text description. When calculating IS for the Oxford-102 dataset, we takes 30,030 randomly selected samples for each image resolution, which are from 1 test real image (1,155 in total) to 26 generated images based on 1 text description.

Quantitative Comparison with the State-of-the-arts (Q1)

To validate our proposed PPAN, in Table 1, we compare our results with state-of-the-arts: GAN-INT-CLS (Reed et al. 2016b), GAWWN (Reed et al. 2016c), StackGAN (Zhang et

Table 1: The Inception Score comparison on two datasets. PPAN outperforms the others on both datasets.

Method	Dataset	
	CUB	Oxford-102
GAN-INT-CLS	2.88±.04	2.66±.03
GAWWN	3.60±.07	-
StackGAN	3.70±.04	3.20±.01
StackGAN++	4.04±.05	3.26±.01
TAC-GAN	-	3.45±.05
HDGAN	4.15±.05	3.45±.07
AttnGAN	4.36±.03	-
PPAN(ours)	**4.38±.05**	**3.52±.02**

Table 2: The Visual-semantic Similarity comparison on two datasets. PPAN outperforms others consistently.

Method	Dataset	
	CUB	Oxford-102
Ground Truth	.302±.151	.336±.138
StackGAN	.228±.162	.278±.134
HDGAN	.246±.157	.296±.131
PPAN(ours)	**.290±.149**	**.297±.136**

al. 2017a), StackGAN++ (Zhang et al. 2017b), TAC-GAN (Dash et al. 2017), HDGAN (Zhang, Xie, and Yang 2018) and AttnGAN (Xu et al. 2017).

As shown in Table 1, PPAN achieves the best performance on IS. Compared with HDGAN, which also adopts one generator and is an end-to-end network, PPAN achieves 0.23 improvement in terms of IS on CUB dataset (from 4.15 to 4.38), and 0.07 improvement on Oxford-102 (from 3.45 to 3.52). This indicates that our PPAN is able to generate more diverse and realistic images conditioned on text descriptions. Compared with AttnGAN, which employs three pairs of generators and discriminators and is not an end-to-end network, PPAN still obtains significant improvements on CUB dataset (from 4.36 to 4.38), with a highly integrated network structure and simple training process.

Table 2 compares VS results on two datasets. And the scores of ground-truth image-text pairs are also computed for reference. PPAN also achieves best performance on both CUB and Oxford-102 datasets, which reflects PPAN can better preserve semantically consistent information between generated images and text descriptions.

Qualitative Comparison with the State-of-the-arts (Q2)

Fig. 3 and Fig. 4 compare visual results with StackGAN, StackGAN++, HDGAN on CUB and Oxford-102 datasets, especially focusing on semantic details, natural color and complex shapes. In Fig. 3, our PPAN generates better image quality and preserves more semantic details than the other images. For example, the semantics of “black bird”, “short, slightly curved bill”, and “long legs” in column 3 are much better represented by our synthesized images, and they are more photo-realistic than others. Particularly, although there is no description about feather, eyes and backgrounds, PPAN still successfully synthesizes smooth feather, vivid eyes and clear backgrounds like the ground-truth image. However, the images synthesized by StackGAN++ or HDGAN lack more or less details, or misrepresent the text description. For example, the image generated by StackGAN++ is too blurred to distinguish eyes, and the image generated by HDGAN misrepresents “black” color information and has an unnatural background.

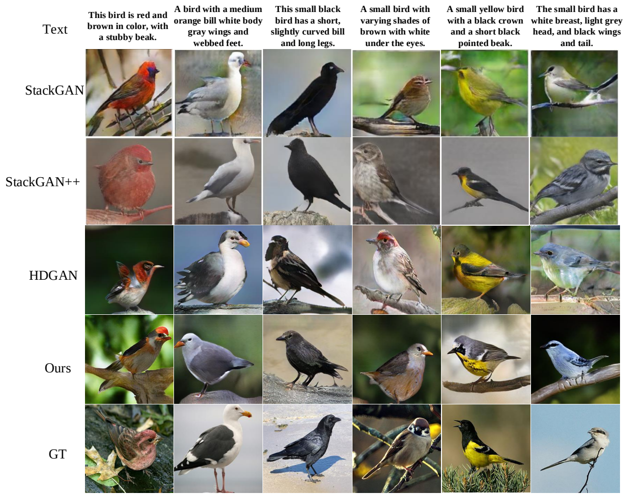

Text
This bird is red and brown in color, with a stubby beak.
A bird with a medium orange bill white body gray wings and webbed feet.
This small black bird has a short, slightly curved bill and long legs.
A small bird with varying shades of brown with white under the eyes.
A small yellow bird with a black crown and a short black pointed beak.
The small bird has a white breast, light grey head, and black wings and tail.
StackGAN
StackGAN++
HDGAN
Ours
GT

Figure 3: Generated images on CUB dataset compares with StackGAN, StackGAN++ and HDGAN. Each sample shows the input text description and generated 256×256 images. Zoom-in for better observation.

Similar observations can be obtained from Fig. 4. In general, the images generated by PPAN are more natural and usually contain complex flower structures. For example, the generated image by PPAN have up to 5 vivid flowers in one image in row 1 of Fig. 4, which provides richer information than the plain ground-truth image. Particularly, the image generated by PPAN also can accurately express the semantic content of the text description, such as “pink, white and yellow in color”, “striped petals”.

By visualizing a large number of images, t-SNE algorithm (Maaten and Hinton 2008) can effectively evaluate the diversity level and semantic consistency of the synthesized images. For each model, a large number of images are synthesized and then embedded into the 2D plane by t-SNE. First, we extract a 2048-dim CNN feature of a generated image using a pre-trained Inception model. Then, t-SNE is applied to embed this feature into a 2D plane, leading to an accurate location for each image in the 2D plane. Fig. 5 shows 50×50 grids with compressed images for CUB dataset. And t-SNE of PPAN has consistent image distribution and rich morphological distribution, so that images generated by PPAN have various image diversity and consistent semantic information. Moreover, t-SNE visualization can easily identify any collapsed modes. We can easily observe HDGAN(left) has one collapsed mode marked with a red rectangle and PPAN (right) has no collapsed mode from Fig. 5.

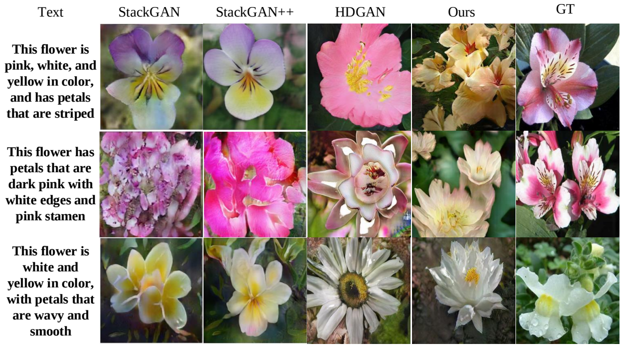

Text
StackGAN
StackGAN++
HDGAN
Ours
GT
This flower is pink, white, and yellow in color, and has petals that are striped
This flower has petals that are dark pink with white edges and pink stamen
This flower is white and yellow in color, with petals that are wavy and smooth

Figure 4: Generated images on Oxford-102 dataset compares with StackGAN, StackGAN++ and HDGAN. Each sample shows the input text description and generated 256 × 256 images. Zoom-in for better observation.

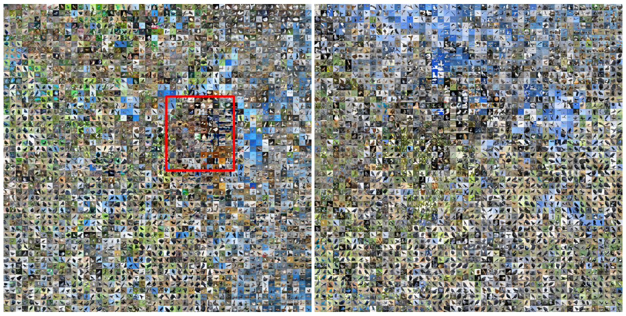


Figure 5: Utilizing t-SNE algorithm to embed images generated by HDGAN(left) and our PPAN(right) on CUB dataset. HDGAN(left) has one collapsed mode marked with a red rectangle and PPAN(right) has no collapsed mode.

Component Analysis (Q3)

Our PPAN consists of several major components, e.g., the Cumulative Blocks (CB) which progressively synthesize multi-scale images, the perceptual loss (L_4) and the class information loss (L_3). In this sub-section, we study the effect of each component on the performance. We start from the basic model (PPAN-CB-L_3-L_4) by removing CB, L_3 and L_4 from PPAN, and then progressively add each component to see their performance. Due to the space limit, we only report the results on the CUB dataset in Table 3.

As can be observed, with CB, our method (PPAN-L_3-L_4) increases the IS scores from 3.27 to 3.62, 3.87 to 4.00, and 4.00 to 4.03 on 64 × 64, 128 × 128 and 256 × 256 resolution. Also, the VS score is improved from 0.239 to 0.257 on 256 × 256 images. This indicates that CB enriching multi-scale features can help match visual images and semantic texts. Furthermore, StackGAN has claimed the difficulty of generating high-resolution images by GANs, because natural image distribution and implied model distribution may not overlap in high dimensional pixel space. But our PPAN has solved this problem by one highly structured generator.

Table 3: Component analysis of Cumulative Blocks on CUB dataset. The IS of 64 × 64, 128 × 128, and 256 × 256 resolution are computed. The VS of 256 × 256 resolution are computed.

Model	IS			VS
	64×64	128×128	256×256	
PPAN-CB-L_3-L_4	3.27±.04	3.87±.05	4.00±.06	.239±.158
PPAN-L_3-L_4	3.62±.04	4.00±.04	4.03±.06	.257±.160
PPAN-L_3	**3.80±.04**	4.27±.04	4.15±.06	.283±.151
PPAN	3.74±.04	**4.27±.04**	**4.38±.05**	**.290±.149**

By further adding the perceptual loss L_4, the IS scores are improved from 3.62 to 3.80, 4.00 to 4.27 and 4.03 to 4.15, and the VS score is improved from 0.257 to 0.283. This indicates that the perceptual loss plays an important role in improving the quality of image synthesizing. By further adding the class information loss L_3 for D_3, it becomes our final model PPAN. The IS score of 256 × 256 images are significantly improved from 4.15 to 4.38. This demonstrates the effectiveness of L_3 for 256 × 256 images generation. We have tried to add L_4 and L_3 on each output but the performance was not improved. One possible reason is that the gradient passed by high resolution 256 × 256 images is sufficient to update the networks with lower resolutions.

Conclusion

In this work, we propose an end-to-end Perceptual Pyramid Adversarial Network (PPAN) to address the problem of generating photo-realistic images conditioned on text descriptions. Instead of using multiple generators and discriminators in previous works, we explore a new perspective to play one-vs-three adversarial games along the depth of one generator using three independent discriminators to synthesize and regularize multi-scale photo-realistic images. The perceptual loss defined on the generator enforces the semantic similarity between the synthesized image and the ground truth, while multi-purpose discriminators encourages semantic consistency, image fidelity and class invariance. Extensive experiments on evaluation scores and visual results demonstrate that PPAN can generate photo-realistic fine-grained images of three resolutions, and perform significantly better than state-of-the-arts on two public datasets.

Acknowledgements

This work is supported by the Fundamental Research Funds for the Central Universities (Grant No. ZYGX2014J063, No. ZYGX2016J085), the National Natural Science Foundation of China (Grant No. 61772116, No. 61872064, No. 61632007, No. 61602049).

References

Bodnar, C. 2018. Text to image synthesis using generative adversarial networks. *arXiv preprint arXiv:1805.00676*.

Dash, A.; Gamboa, J. C. B.; Ahmed, S.; Liwicki, M.; and Afzal, M. Z. 2017. Tac-gan-text conditioned auxiliary classifier generative adversarial network. *arXiv preprint arXiv:1703.06412*.

Doersch, C. 2016. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*.

Gao, L.; Guo, Z.; Zhang, H.; Xu, X.; and Shen, H. T. 2017. Video captioning with attention-based LSTM and semantic consistency. *IEEE Trans. Multimedia* 19(9):2045–2055.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative adversarial nets. In *NIPS*, 2672–2680.

Gupta, A.; Agarwal, A.; Singh, P.; and Rai, P. 2018. A deep generative framework for paraphrase generation. In *AAAI*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Identity mappings in deep residual networks. In *ECCV*, 630–645.

Huang, H.; Yu, P. S.; and Wang, C. 2018. An introduction to image synthesis with generative adversarial nets. *CoRR* abs/1803.04469.

Ioffe, S., and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

Johnson, J.; Alahi, A.; and Fei-Fei, L. 2016. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 694–711.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lai, W.-S.; Huang, J.-B.; Ahuja, N.; and Yang, M.-H. 2017. Deep laplacian pyramid networks for fast and accurate superresolution. In *CVPR*, 5835–5843.

Ledig, C.; Theis, L.; Huszár, F.; Caballero, J.; Cunningham, A.; Acosta, A.; Aitken, A. P.; Tejani, A.; Totz, J.; Wang, Z.; et al. 2017. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, 105–114.

Li, Y.; Min, M. R.; Shen, D.; Carlson, D. E.; and Carin, L. 2018. Video generation from text. In *AAAI*.

Lin, T.-Y.; Dollár, P.; Girshick, R. B.; He, K.; Hariharan, B.; and Belongie, S. J. 2017. Feature pyramid networks for object detection. In *CVPR*, 936–944.

Maaten, L. v. d., and Hinton, G. 2008. Visualizing data using t-sne. *Journal of machine learning research* 9(Nov):2579–2605.

Mao, X.; Li, Q.; Xie, H.; Lau, R. Y.; Wang, Z.; and Smolley, S. P. 2017. Least squares generative adversarial networks. In *ICCV*, 2813–2821.

Nilsback, M.-E., and Zisserman, A. 2008. Automated flower classification over a large number of classes. In *ICVGIP*, 722–729.

Odena, A.; Dumoulin, V.; and Olah, C. 2016. Deconvolution and checkerboard artifacts. *Distill*.

Odena, A.; Olah, C.; and Shlens, J. 2016. Conditional image synthesis with auxiliary classifier gans. *arXiv preprint arXiv:1610.09585*.

Reed, S.; Akata, Z.; Lee, H.; and Schiele, B. 2016a. Learning deep representations of fine-grained visual descriptions. In *CVPR*, 49–58.

Reed, S.; Akata, Z.; Yan, X.; Logeswaran, L.; Schiele, B.; and Lee, H. 2016b. Generative adversarial text to image synthesis. *arXiv preprint arXiv:1605.05396*.

Reed, S. E.; Akata, Z.; Mohan, S.; Tenka, S.; Schiele, B.; and Lee, H. 2016c. Learning what and where to draw. In *NIPS*, 217–225.

Shrivastava, A.; Pfister, T.; Tuzel, O.; Susskind, J.; Wang, W.; and Webb, R. 2017. Learning from simulated and unsupervised images through adversarial training. In *CVPR*, 2242–2251.

Simonyan, K., and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Song, J.; Gao, L.; Guo, Z.; Liu, W.; Zhang, D.; and Shen, H. T. 2017. Hierarchical LSTM with adjusted temporal attention for video captioning. In *IJCAI*.

Song, J.; He, T.; Gao, L.; Xu, X.; Hanjalic, A.; and Shen, H. T. 2018a. Binary generative adversarial networks for image retrieval. In *AAAI*, 394–401.

Song, J.; Zhang, J.; Gao, L.; Liu, X.; and Shen, H. T. 2018b. Dual conditional gans for face aging and rejuvenation. In *IJCAI*, 899–905.

Szegedy, C.; Ioffe, S.; Vanhoucke, V.; and Alemi, A. A. 2017. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, 4278–4284.

Wah, C.; Branson, S.; Welinder, P.; Perona, P.; and Belongie, S. 2011. The caltech-ucsd birds-200-2011 dataset.

Xu, T.; Zhang, P.; Huang, Q.; Zhang, H.; Gan, Z.; Huang, X.; and He, X. 2017. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. *arXiv preprint*.

Zhang, H.; Xu, T.; Li, H.; Zhang, S.; Huang, X.; Wang, X.; and Metaxas, D. 2017a. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. *arXiv preprint*.

Zhang, H.; Xu, T.; Li, H.; Zhang, S.; Wang, X.; Huang, X.; and Metaxas, D. 2017b. Stackgan++: Realistic image synthesis with stacked generative adversarial networks. *arXiv preprint arXiv:1710.10916*.

Zhang, Z.; Xie, Y.; and Yang, L. 2018. Photographic text-to-image synthesis with a hierarchically-nested adversarial network. In *CVPR*.