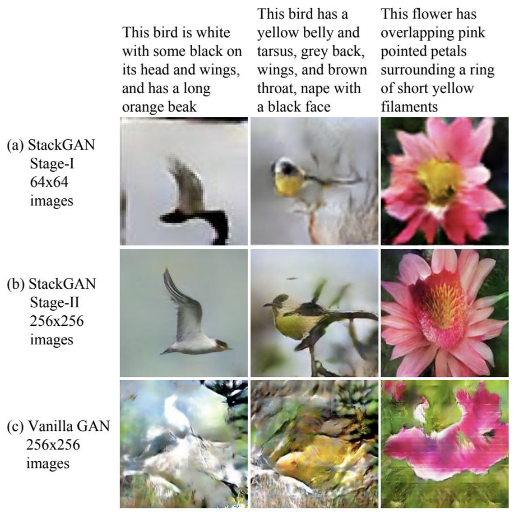
StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks

Problem Statement:
Synthesizing photo-realistic images from text descriptions is a challenging problem in computer vision and has many practical applications like photo editing and computer-aided design. Samples generated by the existing text-to-image approaches fail to contain the necessary details and vivid object parts. The main challenge of this problem is that the space of plausible images given text descriptions is multimodal. There are a large number of images that correctly fit the given text description.

Contribution:
In this paper, the authors propose stacked Generative Adversarial Networks (StackGAN) to generate *photo-realistic* images conditioned on text descriptions. It decomposes the text-to-image generative process into two stages:

- **Stage-I GAN:** It sketches the primitive shape and basic colors of the object conditioned on the given text description, and draws the background layout from a random noise vector, yielding a low resolution image.
- **Stage-II GAN:** It corrects defects in the low resolution image and completes details of the object by reading the text description again, producing a high resolution photo-realistic image.

High resolution 256×256 images can be generated by StackGAN, while state-of-the-art methods have difficulty generating images larger than 64×64.


This bird is white with some black on its head and wings, and has a long orange beak
This bird has a yellow belly and tarsus, grey back, wings, and brown throat, nape with a black face
This flower has overlapping pink pointed petals surrounding a ring of short yellow filaments
(a) StackGAN Stage-I 64x64 images
(b) StackGAN Stage-II 256x256 images
(c) Vanilla GAN 256x256 images

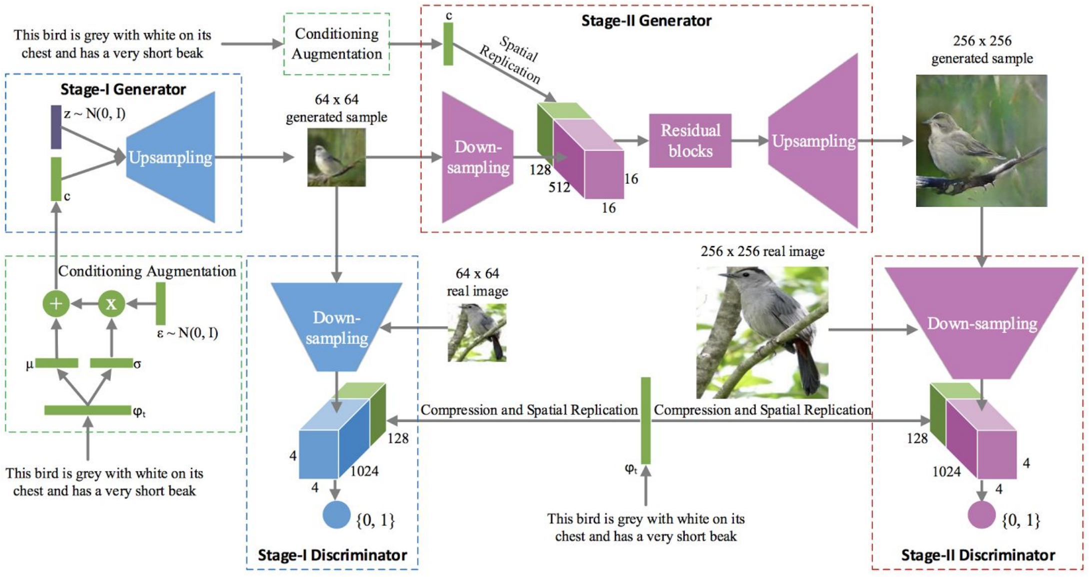
Stacked GAN:

- The conditioning text description t is first encoded by an encoder, yielding a text embedding φ_t.
- Now, randomly sample latent variables from an independent Gaussian distribution $\mathcal{N}(\mu(\varphi_t), \Sigma(\varphi_t))$ where the mean $\mu(\varphi_t)$ and the diagonal covariance matrix $\Sigma(\varphi_t)$ are functions of the text embedding φ_t. This formulation encourages robustness to small perturbations along the conditioning manifold, and thus yields more training pairs given a small number of image-text pairs.
- To further enforce the smoothness over the conditioning manifold and avoid overfitting, we add the following regularization term to the objective of the generator during training:
$$\mathcal{D}_{KL}(\mathcal{N}(\mu(\varphi_t), \sigma(\varphi_t)) \mid\mid \mathcal{N}(0,1))$$
which is the Kullback-Leibler divergence between the standard Gaussian distribution and the conditioning Gaussian distribution.
- Conditioned on the Gaussian latent variables c_0, stacked GAN trains the discriminator D_0 and the generator G_0 by alternatively **maximizing** $\mathcal{L}_{D_0}$ and **minimizing** $\mathcal{L}_{G_0}$:
    - $\mathcal{L}_{D_0} = \mathbb{E}_{(I_0,t)\sim p_{data}}[log D_0(I_0, \varphi_t)] + \mathbb{E}_{z\sim p_z, t\sim p_{data}}[log(1 - D_0(G_0(z, c_0), \varphi_t))]$
    - $\mathcal{L}_{G_0} = \mathbb{E}_{z\sim p_z, t\sim p_{data}}[log(1 - D_0(G_0(z, c_0), \varphi_t))] + \lambda \mathcal{D}_{KL}(\mathcal{N}(\mu(\varphi_t), \sigma(\varphi_t)) \mid\mid \mathcal{N}(0,1))$

where the real image I_0 and the text description t are from the true data distribution p_{data}.
z is a noise vector randomly sampled from a given (Gaussian) distribution p_z.
φ_t is the text embedding generated by a pre-trained encoder.
Gaussian latent variables c_0 are samples from $\mathcal{N}(\mu(\varphi_t), \Sigma(\varphi_t))$ to reflect the text description.

Note: For the Stage-II GAN, the input to generator will be $G_0(z, c_0)$ instead of the random noise vector z.

Architecture:


This bird is grey with white on its chest and has a very short beak
Conditioning Augmentation
Stage-II Generator
c
Spatial Replication
256 x 256 generated sample
Stage-I Generator
z ~ N(0, I)
c
Upsampling
64 x 64 generated sample
Down-sampling
128
512
16
16
Residual blocks
Upsampling
256 x 256 real image
Conditioning Augmentation
ε ~ N(0, I)
μ
σ
φt
64 x 64 real image
Down-sampling
Down-sampling
Compression and Spatial Replication
Compression and Spatial Replication
128
4
1024
4
{0, 1}
φt
128
1024
4
4
{0, 1}
This bird is grey with white on its chest and has a very short beak
This bird is grey with white on its chest and has a very short beak
Stage-I Discriminator
Stage-II Discriminator

For the generator, the text embedding φ_t is fed into a fully connected layer to generate μ_0 and σ_0 (σ_0 are the values in the diagonal of Σ_0) for Gaussian distribution $\mathcal{N}(\mu(\varphi_t), \Sigma(\varphi_t))$. The N_g dimensional conditioning vector c_0 is computed by $c_0 = \mu_0 + \sigma_0 \odot \epsilon$ (where $\odot$ is the element-wise multiplication, $\epsilon \sim \mathcal{N}(0,1)$). Then, c_0 is concatenated with a N_z dimensional noise vector to generate a $W_0 \times H_0$ image by a series of up-sampling blocks.

For the discriminator, the text embedding φ_t is first compressed to N_d dimensions using a fully-connected layer and then spatially replicated to form a $M_d \times M_d \times N_d$ tensor. Meanwhile, the image is fed through a series of down-sampling blocks until it has $M_d \times M_d$ spatial dimension. Then, the image filter map is concatenated along the channel dimension with the text tensor. The resulting tensor is further fed to a 1×1 convolutional layer to jointly learn features across the image and the text. Finally, a fully- connected layer with one node is used to produce the decision score.