Generating Images Part by Part with Composite Generative Adversarial Networks

Hanock Kwak and Byoung-Tak Zhang
School of Computer Science and Engineering, Seoul National University
{hnkwak, btzhang}@bi.snu.ac.kr

Abstract

Image generation remains a fundamental problem in artificial intelligence in general and deep learning in specific. The generative adversarial network (GAN) was successful in generating high quality samples of natural images. We propose a model called composite generative adversarial network, that reveals the complex structure of images with multiple generators in which each generator generates some part of the image. Those parts are combined by alpha blending process to create a new single image. It can generate, for example, background and face sequentially with two generators, after training on face dataset. Training was done in an unsupervised way without any labels about what each generator should generate. We found possibilities of learning the structure by using this generative model empirically.

1 Introduction

Images are composed of several different objects forming a hierarchical structure with various styles and shapes. Recently, deep learning models are used to implicitly disentangle those complex underlying patterns [Reed et al., 2014], form distributed feature representations [Hinton and Salakhutdinov, 2006], and solve classification [Krizhevsky et al., 2012] and generation [Radford et al., 2015] problems using large dataset. While lots of classification problems have focused on abstraction for finite number of labels, generation problems need a concretizing process starting from latent variables. In other words, the latent variables of generative models must contain concrete information about the data, which is not the case for the features of discriminative models. This gives us interesting challenges in generation task which is a fundamental problem of artificial intelligence. Even though we can easily imagine a scene by combining and mixing semantic parts, current generative models are far from reaching our abilities.

Generative adversarial networks (GAN) [Goodfellow et al., 2014], based on deep neural networks, are successful unsupervised learning models that can generate samples of natural images generalized from the training data. It provides an alternative to intractable maximum likelihood estimation and pixel-wise loss functions. GAN simultaneously trains two models: a generator G that tries to generate real images, and a discriminator D that classifies between the real images that came from the training data and the fake images that came from G. The D alleviates lack of semantic consideration in pixel-wise loss function used in most auto-encoder models.

It is proven that if the GAN has enough capacity, data distribution formed by G can converge to the distribution over real data [Goodfellow et al., 2014]. In practice, however, the convergence is intractable and it is easy to overfit due to exponential complexity of images in which multiple objects exist in any position with noisy features like shape, color, and size. To solve this issue, we propose a composite generative adversarial network (CGAN) that can generate images part by part instead of generating whole images directly. CGAN differs from other recurrent generative models [Mansimov et al., 2015; Im et al., 2016], which simply add the sequence of generated images blurring each image in the overlapping areas. In order to address this problem, CGAN uses an alpha channel for opacity along with RGB channels to stack images iteratively with alpha blending process. The alpha blending process maintains previous image in some areas and overlap the new image perfectly in other areas. For instance, given a transparent image the model may put a snowy background first, then later add trees and characters sequentially as shown in Figure 1.

Figure 1: Examples of generated images from CGAN with three generators. $C_1, C_2, C_3$ are images generated from three generators, respectively, and $O^{(3)}$ is final output. Similar real images are shown for comparison. Black and white checkerboard is the default background for transparent images.
Since VAE optimizes pixel-wise loss functions, images generated by VAE tend to be blurred.

Similar to the DRAF, a recurrent adversarial network [Im et al., 2016] adds generated images from multiple generators sequentially and puts the sigmoid function at the end. Simply adding the images based on RGB channels results blurring effects. Our proposed CGAN uses the additional alpha channel to avoid this issue.

Some of the variants of GAN, such as LAPGAN [Denton et al., 2015], DCGAN [Radford et al., 2015], and recurrent adversarial network [Im et al., 2016], improved the quality of generated images. VAE/GAN [Larsen et al., 2015] replaced pixel-wise loss function of VAE with feature-wise loss function where the features come from the discriminator of GAN. [Ulyanov et al., 2016] have used two GAN: the Structure-GAN generates structures; the Style-GAN puts styles on the structures.

3 Model

In this section, we first review GAN in detail and describe how we combined GAN with alpha blending process. In addition to the basic idea, several recent deep learning techniques (batch normalization [Ioffe and Szegedy, 2015], ADAM [Kingma and Ba, 2015], LSTM [Hochreiter and Schmidhuber, 1997], etc.), critical to the performance of CGAN, were utilized.

3.1 Generative Adversarial Networks

A GAN has two networks: a generator $G$ that tries to generate real data given a noise $z \sim p_z(z)$, and a discriminator $D \in [0, 1]$ that classifies the real data $x \sim p_{data}(x)$ and the fake data $G(z)$ generated from $G$. The objective of $G$ is to fit the true data distribution deceiving $D$ by playing following minimax game:

$$\min_{\theta_G} \max_{\theta_D} \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log (1-D(G(z)))]$$

(1)

where $\theta_G$ and $\theta_D$ are parameters of $G$ and $D$, respectively. Given mini-batch of $\{x_1, x_2, ..., x_m\}$ and $\{z_1, z_2, ..., z_m\}$, $\theta_G$ and $\theta_D$ are updated for each iteration as following:

$$\theta_G \leftarrow \theta_G - \gamma \nabla_{\theta_G} \sum_{i=1}^{m} \log (1 - D(G(z_i)))$$

$$\theta_D \leftarrow \theta_D + \gamma \nabla_{\theta_D} \sum_{i=1}^{m} \log D(x_i) + \log (1 - D(G(z_i)))$$

(2)

where $\gamma$ is a learning rate which could be different for $G$ and $D$.

3.2 Alpha Blending

The alpha blending combines two translucent images, producing a new blended image. The value of alpha is between 0.0 and 1.0, where a pixel is fully transparent if 0.0, and fully opaque if 1.0. We denote $C_{iRGB}$ as a 3-dimensional vector.
of RGB values in position \((i, j)\) and \(C_{ijA}\) as a scalar alpha value of the same position. A CGAN uses alpha blending which covers the previous image \(C^{(\text{prev})}\) with the next image \(C^{(\text{next})}\) to make the new image \(C^{(\text{new})}\):

\[
C^{(\text{new})}_{ijRGB} = C^{(\text{prev})}_{ijRGB} C^{(\text{prev})}_{ijA} (1 - C^{(\text{next})}_{ijA}) + C^{(\text{next})}_{ijRGB} C^{(\text{next})}_{ijA}.
\]

(3)

Assuming that the new image is opaque, \(C^{(\text{new})}_{ijA}\) is always 1. This process maintains colors of the next image where \(C^{(\text{next})}_{ijA}\) is nearly one, and that of the previous image where \(C^{(\text{next})}_{ijA}\) is nearly zero.

### 3.3 Composite Generative Adversarial Networks

A CGAN, an extension of GAN, consists of multiple generators connected with a recurrent neural network (RNN) as shown in Figure 2. The generators in CGANs are different from that of GANs as there are additional alpha channels in the output. The images are then combined sequentially with alpha blending to form a final image.

Given a input noise \(z\) of CGAN, the RNN produces inputs \(h_1, h_2, ..., h_n\) of each generator sequentially as following:

\[
z \sim \rho_z(z)\]

(4)

\[
h_0 = z\]

(5)

\[
h_t = RNN(h_{t-1})\]

(6)

The RNN preserves the consistency between the generators, so that the generated images are all related. We denote the generators and the generated images as \(G_1, G_2, ..., G_n\), and \(C^{(1)}, C^{(2)}, ..., C^{(n)}\) respectively. Note that \(C^{(t)}\) is a RGBA image in which all pixels are four dimensional vectors. To illustrate how a final output image \(O^{(n)}\) is made, we denote intermediate images as \(O^{(1)}, O^{(2)}, ..., O^{(n-1)}\). Then, the following explains how \(O^{(n)}\) is formed:

\[
C^{(t)} = G_i(h_t)\]

(7)

\[
O^{(t)}_{ijRGB} = \begin{cases} C^{(1)}_{ijRGB} C^{(1)}_{ijA} & \text{if } t = 1 \\ O^{(t-1)}_{ijRGB} (1 - C^{(t)}_{ijA}) + C^{(t)}_{ijRGB} C^{(t)}_{ijA} & \text{if } t > 1 \end{cases}
\]

(8)

The objective of generators in whole is same as that of GAN, and the algorithm of CGAN is similar to that of GAN as illustrated in algorithm 1.

### 4 Experiments

Evaluation of generative models is problematic [Theis et al., 2016] due to various objectives (density estimator, feature learning, clustering, etc.) of unsupervised learning. Since our objective is to generate images part by part, qualitative analysis takes most part in assessment of CGAN.

All images are resized to \(64 \times 64\) with antialiasing. We used long short-term memory (LSTM) [Hochreiter and Schmidhuber, 1997] architecture for RNN. The structures of each generator are identical and similar to that of DCGAN [Radford et al., 2015] which has series of four fractionally strided convolutions (or transposed convolution).
Algorithm 1: The algorithm of CGAN

Input: dataset $X$, the number of generators $n$, mini-batch size $m$
Initialize $\theta_D, \theta_{G_1}, \theta_{G_2}, \ldots, \theta_{G_n}$.
for number of training iterations do
    Select $m$ data $\{x_1, x_2, \ldots, x_m\} \subset X$ randomly
    Draw $m$ noises $\{z_1, z_2, \ldots, z_m\} \sim p_z(z)$
    $\theta_D \leftarrow \theta_D + \gamma \nabla_{\theta_D} \sum_{i=1}^m \log D(x_i) + \log(1 - D(G(z_i)))$
    for $i = 1$ to $n$ do
        $\theta_G \leftarrow \theta_G - \gamma \nabla_{\theta_G} \sum_{i=1}^m \log(1 - D(G(z_i)))$
    end for
end for

4.1 CelebA face images

CelebA dataset contains 202,599 face images, 11,177 number of identities. The images in this dataset cover large pose variations and background clutter. We used two generators on this dataset and the result is shown in Figure 3 (a). The first generator generated backgrounds and the second one generated faces mostly, yet sometime second one only generates hairs and other parts are supported by the first one. Even though overall process is stochastic and behavior of CGAN is unpredictable due to its unsupervised setting, CGAN successfully generates full images while all generators share their efforts fairly.

4.2 Pororo cartoon video

Pororo is a cartoon video that consists of 183 episodes with 1,232 minutes of playing time. The dataset has large diversities of poses, sizes, and positions of characters, yet most of the backgrounds come from snowy mountains, glaciers, forests and wooden houses. We captured frames for each second from the video and shuffled to avoid bias. As shown in Figure 3 (b), some final images are generated combining backgrounds, trees and characters separately. The ways of generation are diverse due to complexity of the real images.

5 Discussion

Disentanglement of high dimensional data has been thoroughly studied, yet it has not been sufficiently solved using unsupervised learning. Compared to humans who see the video of real life for numerous years, 100K ~ 1M of images might be too small for learning tasks like distinguishing objects from scenes. Yet we found implicit possibilities of structure learning from images without any labels by constructing the hierarchical structures of the images. With a larger dataset, future models could understand images in more detail and might be capable of changing arrangement of characters dynamically without touching the backgrounds.

Our model could be extended to other domains such as video, text, audio, or a combination of them. One of the main goal of unsupervised learning is any modality to any modality mapping. Since most of the data has hierarchical structures, studies on decomposing the combined data are essential to finding correlation between multimodal data.

6 Acknowlegements

This work was partly supported by the Institute for Information & Communications Technology Promotion (R0126-16-1072-SW.StarLab), Korea Evaluation Institute of Industrial Technology (10044009-HRI.MESSI, 10060086-RISF), and Agency for Defense Development (UD130070ID-BMRR) grant funded by the Korea government (MSIP, DAPA).

References


