

Generative Adversarial Networks

Sahin Olut

Department of Computer Engineering
Istanbul Technical University

November 4, 2017

Outline

- 1 Motivation
 - Why should we study generative models?
 - Some results from recent GAN works
- 2 How does GAN work?
 - GAN Architecture
 - Formulation of GAN
 - Training procedure of GANs
- 3 Applications of GANs
 - Computer Vision
 - Reinforcement Learning

Outline

- 1 Motivation
 - Why should we study generative models?
 - Some results from recent GAN works
- 2 How does GAN work?
 - GAN Architecture
 - Formulation of GAN
 - Training procedure of GANs
- 3 Applications of GANs
 - Computer Vision
 - Reinforcement Learning

Motivation

- We can restore the missing data by generative models. (Image inpainting, super-resolution)

Motivation

- We can restore the missing data by generative models. (Image inpainting, super-resolution)
- We can generate new examples to enhance our classifier networks. (Data augmentation strategy)

Motivation

- We can restore the missing data by generative models. (Image inpainting, super-resolution)
- We can generate new examples to enhance our classifier networks. (Data augmentation strategy)
- If we want our computers to understand, we have to teach them to create. (I do not understand what I cannot create. – Richard Feynman)

Outline

1 Motivation

- Why should we study generative models?
- Some results from recent GAN works

2 How does GAN work?

- GAN Architecture
- Formulation of GAN
- Training procedure of GANs

3 Applications of GANs

- Computer Vision
- Reinforcement Learning

Recent works

- Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network [Ledig et al., 2016]

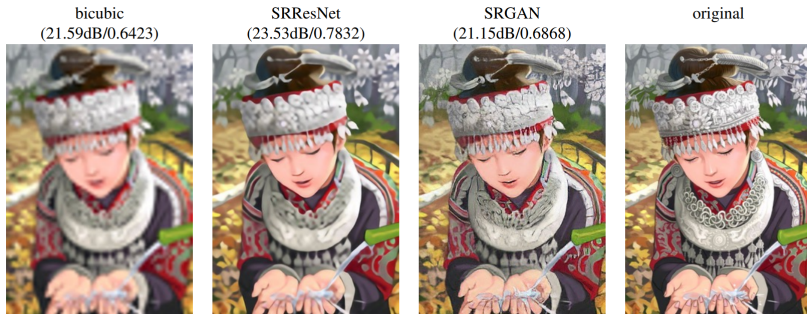


Figure: Work by Ledig et al., 2016

Recent works

- Plug & Play Generative Networks: Conditional Iterative Generation of Images in Latent Space [Nguyen et al., 2017]

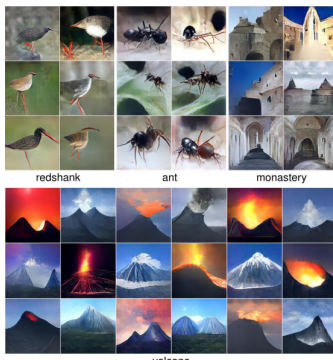


Figure: Synthetic images generated from ImageNet classes.

Outline

- 1 Motivation
 - Why should we study generative models?
 - Some results from recent GAN works
- 2 How does GAN work?
 - GAN Architecture
 - Formulation of GAN
 - Training procedure of GANs
- 3 Applications of GANs
 - Computer Vision
 - Reinforcement Learning

Discriminator and Generator Networks

- What is a generative model?

Discriminator and Generator Networks

- What is a generative model?
- Discriminator's role in GAN is to predict whether the input is generated or sampled from training data.
- The aim of generator is to capture the distribution of the training data.
- According to Goodfellow et al., it is a minimax game between generator and discriminator where generator tries to fool discriminator.(There are some debates about it.)

Outline

- 1 Motivation
 - Why should we study generative models?
 - Some results from recent GAN works
- 2 How does GAN work?
 - GAN Architecture
 - **Formulation of GAN**
 - Training procedure of GANs
- 3 Applications of GANs
 - Computer Vision
 - Reinforcement Learning

- The description of GANs leads us to formulation for loss:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{\text{data}}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))] \quad (1)$$

- Where both networks rely on gradients flowing through discriminator.

Outline

- 1 Motivation
 - Why should we study generative models?
 - Some results from recent GAN works
- 2 How does GAN work?
 - GAN Architecture
 - Formulation of GAN
 - Training procedure of GANs
- 3 Applications of GANs
 - Computer Vision
 - Reinforcement Learning

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k , is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

for k steps **do**

- Sample minibatch of m noise samples $\{z^{(1)}, \dots, z^{(m)}\}$ from noise prior $p_g(z)$.
- Sample minibatch of m examples $\{x^{(1)}, \dots, x^{(m)}\}$ from data generating distribution $p_{\text{data}}(x)$.
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^m \left[\log D(x^{(i)}) + \log (1 - D(G(z^{(i)}))) \right].$$

end for

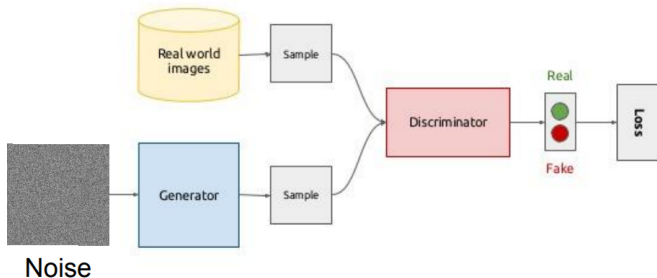
- Sample minibatch of m noise samples $\{z^{(1)}, \dots, z^{(m)}\}$ from noise prior $p_g(z)$.
- Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^m \log (1 - D(G(z^{(i)}))).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

GAN Framework



From now and on, we have a basic grasp of GAN, therefore we can code our own GAN! The starter code can be found in my GitHub Profile: github.com/norveclibalicki

Outline

- 1 Motivation
 - Why should we study generative models?
 - Some results from recent GAN works
- 2 How does GAN work?
 - GAN Architecture
 - Formulation of GAN
 - Training procedure of GANs
- 3 Applications of GANs
 - Computer Vision
 - Reinforcement Learning

Computer Vision Applications

- Image generation(Plug and Play GAN)
- Style transfer(CycleGAN)
- Image inpainting, super-resolution(SRGAN)
- Image to text (Image captioning - Generative Adversarial Text to Image Synthesis)

Computer Vision Applications

- Image generation(Plug and Play GAN)
- Style transfer(CycleGAN)
- Image inpainting, super-resolution(SRGAN)
- Image to text (Image captioning - Generative Adversarial Text to Image Synthesis)
- There are many applications which are not covered above.

Outline

- 1 Motivation
 - Why should we study generative models?
 - Some results from recent GAN works
- 2 How does GAN work?
 - GAN Architecture
 - Formulation of GAN
 - Training procedure of GANs
- 3 Applications of GANs
 - Computer Vision
 - Reinforcement Learning

Applications to real life

- Discriminator can be rewarded for labeling correctly and a new loss can be defined by that way. (Still on research)
- Generating environment and test for reinforcement learning applications.
- Simulating particle experiments like they do in CERN.

Many things can be done with GANs however, GANs have limitations as well.

- There is no training procedure that has been proven to be successful.

- There is no training procedure that has been proven to be successful.
- Sometimes discriminator learns faster than generator(predicts everything correctly), which leads a gradient problem to generator. In some cases, it is vice-versa as well.

- There is no training procedure that has been proven to be successful.
- Sometimes discriminator learns faster than generator(predicts everything correctly), which leads a gradient problem to generator. In some cases, it is vice-versa as well.
- After from some point, generator keeps generating similar examples.

- There is no training procedure that has been proven to be successful.
- Sometimes discriminator learns faster than generator(predicts everything correctly), which leads a gradient problem to generator. In some cases, it is vice-versa as well.
- After from some point, generator keeps generating similar examples.
- Losses of models are not meaningful as classifier's. It just keep oscillating back and forth.

Loss graphic of a GAN



Figure: Taken from <http://www.rricard.me>

Summary

- GANs are **very** powerful tools for generating new samples from data yet it has serious issues.
- GAN uses **semi-supervised** approach therefore, there is no need for data-labeling.
- **Unsupervised learning is the cake of true AI.** (Yann LeCun, Head of Facebook Research)

For Further Reading I



Chintala et al.

GAN Hacks

github.com/soumith/ganhacks



Goodfellow et al.

Generative Adversarial Networks

arXiv: 1406.2661, 2014.