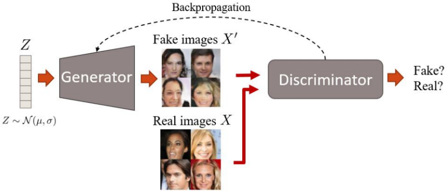
Generative adversarial networks

Victor Kitov

v.v.kitov@yandex.ru

Intuition of adversarial learning

Generative adversarial learning for images:



GAN training overview.

Analogy for bank and a money counterfeiter (having a spy in the bank).

- they compete, until money counterfeiter learns to make perfect money replicas!

Seminal paper on GAN[1]

- 2 multilayer perceptrons:
    - generator $G(z) = G(z|\theta_g)$
        - outputs generated object x
    - discriminator $D(x) = D(x|\theta_d)$
        - probability that x is from training set and not generated by G.

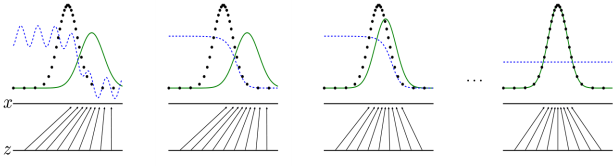
[1] https://arxiv.org/pdf/1406.2661.pdf

Game

D and G play two-player game with minimax function $V(G, D)$

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

Incremental learning:



(a) (b) (c) (d)

black dotted: $p_{data}(x)$; green: $p_{generated}(x)$; blue: $D(x) = p(x \text{ is true}|x)$

Losses

Score for discriminator (for fixed θ_g):

$$\mathbb{E}_{x\sim p_{data}(x)}\left[\log D(x)\right] + \mathbb{E}_{z\sim p_z(z)}\left[\log(1 - D(G(z)))\right] \to \max_{\theta_d}$$

Score for generator (probability of being detected):

$$\mathbb{E}_{z\sim p_z(z)}\left[\log(1 - D(G(z)))\right] \to \min_{\theta_g}$$

- on early iterations generator is very unrealistic
- so $D(G(z)) \approx 0$, gradient of $\log(1 - D(G(z))$ is small.
- better works another score:

$$\mathbb{E}_{z\sim p_z(z)}\left[\log(D(G(z)))\right] \to \max_{\theta_g}$$

Algorithm

for number of training iterations **do**

for k steps **do**

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

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

end for

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

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

end for

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

Optimal value for discriminator

Theorem: For fixed G optimal discriminator is:

$$D^*(x|G) = \frac{p_{data}(x)}{p_{data}(x) + p_g(x)}$$

Proof:

$$V(G, D) = \int_x p_{data}(x) \log(D(x))dx + \int_z p_z(x) \log(1 - D(g(z)))dz =$$
$$= \int_x p_{data}(x) \log(D(x))dx + p_g(x) \log(1 - D(x))dx$$

Since $\arg\max_y \{a \log(y) + b \log(1 - y)\} = \frac{a}{a+b}$ for any a, b =>

$$\arg\max_D V(G, D) = \frac{p_{data}(x)}{p_{data}(x) + p_g(x)}$$

Optimal

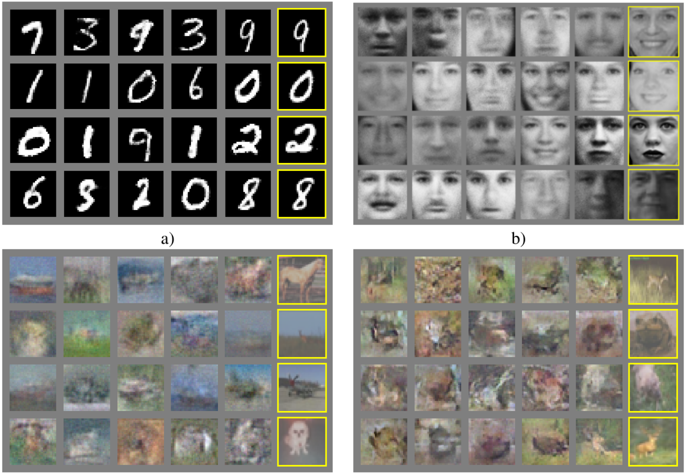
Generator cost function:

$$\begin{aligned} C(G) &= \max_D V(G, D) \\ &= \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}}[\log D_G^*(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}}[\log(1 - D_G^*(G(\boldsymbol{z})))] \\ &= \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}}[\log D_G^*(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x} \sim p_g}[\log(1 - D_G^*(\boldsymbol{x}))] \\ &= \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}}\left[\log \frac{p_{\text{data}}(\boldsymbol{x})}{P_{\text{data}}(\boldsymbol{x}) + p_g(\boldsymbol{x})}\right] + \mathbb{E}_{\boldsymbol{x} \sim p_g}\left[\log \frac{p_g(\boldsymbol{x})}{p_{\text{data}}(\boldsymbol{x}) + p_g(\boldsymbol{x})}\right] \end{aligned}$$

This is maximized for $p_g(x) = p_{data}(x)$:

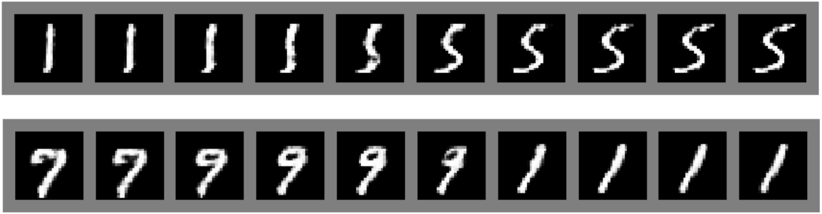
$$C(G) = \mathbb{E} \log \frac{1}{2} + \mathbb{E} \log \frac{1}{2}$$

Generated images



Latent space

Linear interpolation of objects in latent space:



Results

Parzen-window based log-likelihood:

- MNIST - dataset of digit images
- TFD - Toronto faces dataset

Model	MNIST	TFD
DBN [3]	138 ± 2	1909 ± 66
Stacked CAE [3]	121 ± 1.6	$\mathbf{2110 \pm 50}$
Deep GSN [6]	214 ± 1.1	1890 ± 29
Adversarial nets	$\mathbf{225 \pm 2}$	$\mathbf{2057 \pm 26}$

Table of Contents

Experiments[2]

- From transformed image->reconstruct original image
  - denoising, super-resolution, deblurring.
- Quality metric: peak signal-to-noise ratio (PSNR)
- Datasets:
  - Human faces - Large-scale CelebFaces Attributes Dataset
  - Natural scenes - MIT Places Database

[2] From http://stanford.edu/class/ee367/Winter2017/yan_wang_ee367_win17_report.pdf

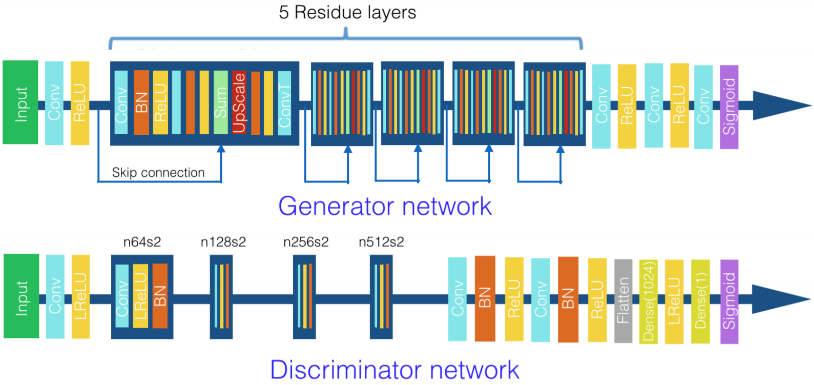
Architecture

- 2 networks: generator, discriminator.
- Discriminator tries to discriminate whether:
    - image came from the training set
    - image came from the generator
- Generator takes **corrupted image as input** and tries to reconstruct original image.

Losses

- **Generator loss:** $0.9\mathcal{L}_{content} + 0.1\mathcal{L}_{G,advers}$
  - $\mathcal{L}_{content} = \left\| I - \widehat{I} \right\|_1$, where I-original and $\widehat{I}$-reconstructed image.
  - $\mathcal{L}_{G,advers}$-standard generator adversarial loss.
- **Discriminator loss:** $\mathcal{L}_{D,advers}$
  - $\mathcal{L}_{D,advers}$-standard discriminator adversarial loss.

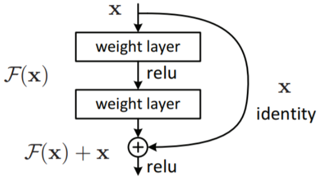
Generator, discriminator structure



Generator details

- Residual networks are used in generator.[3]
- Key idea of residual network:
    - use much more layers
    - layers grouped into groups with similar structure
    - each group learns **small correction** to identity function (to prevent overfitting)

Building block of residual network:



[3] https://arxiv.org/pdf/1512.03385.pdf

Definitions

- I: original image
- K: reconstructed image
- m, n: image dimensions
- Mean squared error (MSE):
  - for grayscale images:

$$MSE = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}[I(i,j) - K(i,j)]^2$$

  - for (r,g,b) images (let c be color channel):

$$MSE = \frac{1}{3mn}\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{c=1}^{3}[I(i,j,c) - K(i,j,c)]^2$$

- MAX: maximum possible pixel value
  - for B-bit image $MAX = 2^B - 1$

Peak signal-to-noise ratio (PSNR)[4]

PSNR measures **quality** of image reconstruction:

$$PSNR = 10\log_{10}\left(\frac{MAX^2}{MSE}\right)$$

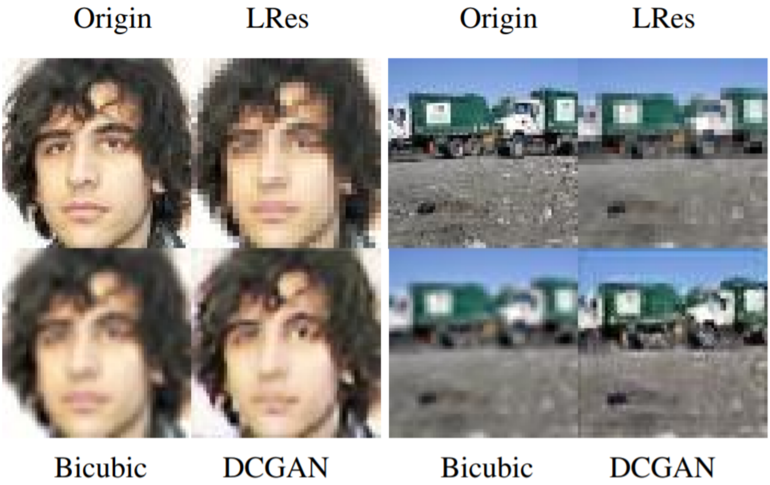
[4] https://en.wikipedia.org/wiki/Peak_signal-to-noise_ratio

1. Application use-case
   - Peak signal-to-noise ratio (PSNR)
   - Experiments

Super-resolution

- **Super-resolution:** recover higher resolution image from its low resolution variant.
  - e.g. from limited device zoom capacity (camera, microscope)
- **Baseline algorithms:**
  - naive scaling (LRes)
  - bicubic interpolation (Bicubic)
- **Results:**
  - PSNR of bicubic is best, but GAN-reconstructed images are more sharp
    and more good-looking for humans (retain high level features).
  - GAN super-resolution for faces works better than for places (which are less typical)

Super-resolution outputs (subsampling=2)



Super-resolution outputs (subsampling=4)

Original LRes Original LRes

Bicubic DCGAN Bicubic DCGAN

Baselines

- **Denoising:** noisy image->clean image
  - e.g. from measurement imperfection.
- **Baseline algorithms:**
  - median filter
  - non-local means
- **Results:**
  - PSNR are comparable, but GAN-reconstructed images are more sharp
    and more good-looking for humans (retain high level features).

Non-local means baseline[5]

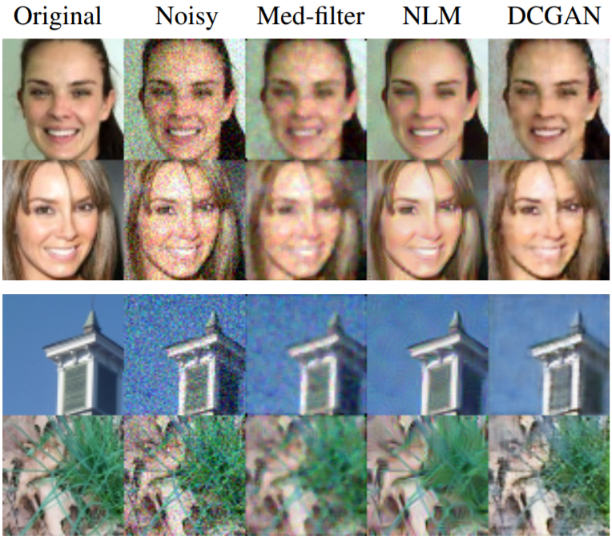
$$u(p) = \frac{1}{C(p)} \sum_{q \in \Omega} v(q) f(p, q)$$

where we used definitions:

- $v(\cdot)$: original image with noise
- $u(\cdot)$: denoised image
- p, q: image locations
- $f(p, q)$: similarity of pixels p, q by their neighborhoods $R(\cdot)$
- $C(p) = \sum_{q \in \Omega} f(p, q)$
- $f(p, q) = e^{-\frac{1}{h^2}|B(q) - B(p)|^2}$
- $B(p) = \frac{1}{|R(p)|} \sum_{i \in R(p)} v(i)$

[5] https://en.wikipedia.org/wiki/Non-local_means

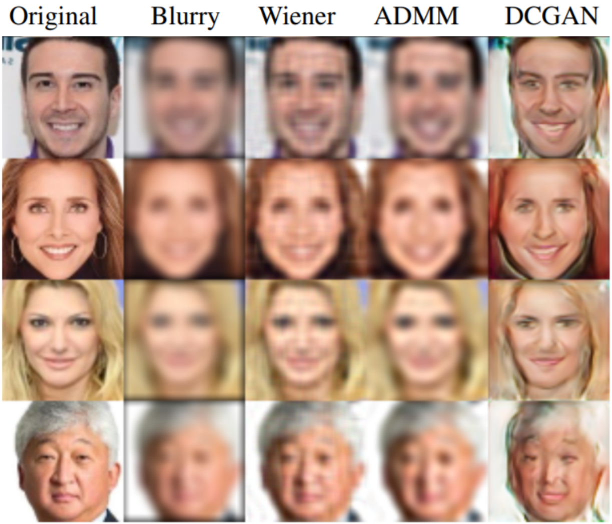
Denoising outputs



Deblurring

- **Deblurring:** images blurred and small Gaussian noise added.
  - e.g. from camera motion.
- **Baseline algorithms:**
  - Wiener filter
  - alternating direction method of multipliers (ADMM)
- **Results:**
  - PSNR of GAN is lower, but GAN-reconstructed images are more sharp
    and more good-looking for humans (retain high level features).
  - GAN super-resolution for faces works better than for places (which are less typical)

Deblurring faces outputs



Deblurring places outputs (not accurate)

Analysis of experiments

- Unequal conditions:
  - Baseline methods use only test image.
  - GAN uses information from the whole training set.
- GANs give smaller PSNR
  - may be attributed to small training set
- GANs give more sharp output
  - to fool "blurry-based" discriminator
  - do not fallback to averaging as standard methods

Analysis of experiments

- GANs reproduce small details on images
  - details learned from other images of the training set.
- GAN performance can be improved by training on specific subsets of objects
  - e.g. train separate face models for different sex, age, nationality, etc.
  - especially important for diverse objects such as places.

Table of Contents

Yet another possible application: impainting

Joining GAN and VAE[6]

- GAN generator learns to produce
  - sharp realistic images
  - **some subset** of objects in training set
    - problem called "*model collapse*"
- Decoder of variational autoencoder (VAE) learns to produce
  - **most training objects**.
  - but generates oversmoothed results
- Combine strong sides of GAN and VAE: train generator on combination of GAN and VAE loss!

[6] https://arxiv.org/pdf/1512.09300.pdf