

Deep Generative Models

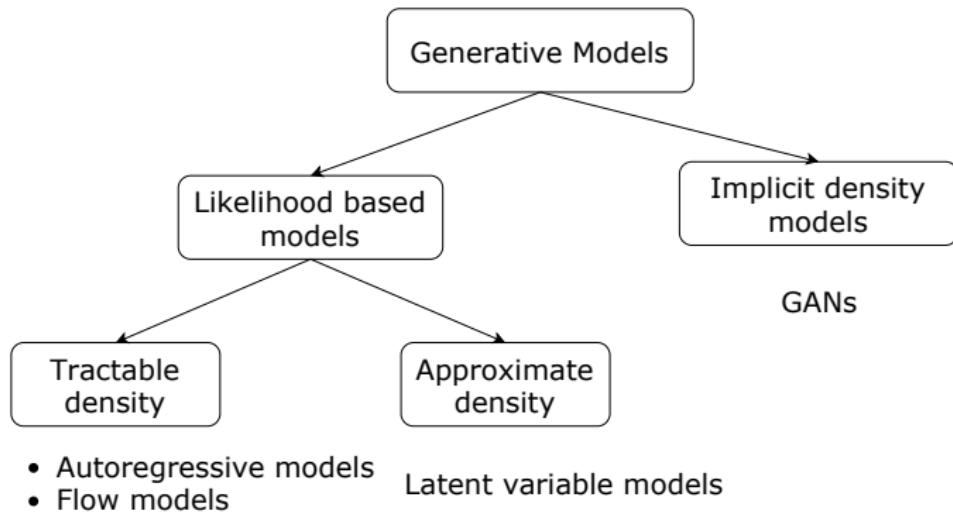
Lecture 8

Roman Isachenko

Moscow Institute of Physics and Technology

2020

Generative models zoo



Likelihood based models

Is likelihood a good measure of model quality?

Poor likelihood
Great samples

$$p_1(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^n \mathcal{N}(\mathbf{x}|\mathbf{x}_i, \epsilon \mathbf{I})$$

For small ϵ this model will generate samples with great quality, but likelihood will be very poor.

Great likelihood
Poor samples

$$p_2(\mathbf{x}) = 0.01p(\mathbf{x}) + 0.99p_{\text{noise}}(\mathbf{x})$$

$$\begin{aligned} \log [0.01p(\mathbf{x}) + 0.99p_{\text{noise}}(\mathbf{x})] &\geq \\ \geq \log [0.01p(\mathbf{x})] &= \log p(\mathbf{x}) - \log 100 \end{aligned}$$

Noisy irrelevant samples, but for high dimensions $\log p(\mathbf{x})$ becomes larger.

Likelihood-free learning

- ▶ Likelihood is not a perfect measure for quality measure for generative model.
- ▶ Likelihood could be intractable.

Where did we start

We would like to approximate true data distribution $\pi(\mathbf{x})$. Instead of searching true $\pi(\mathbf{x})$ over all probability distributions, learn function approximation $p(\mathbf{x}|\theta) \approx \pi(\mathbf{x})$.

Imagine we have two sets of samples

- ▶ $\mathcal{S}_1 = \{\mathbf{x}_i\}_{i=1}^{n_1} \sim \pi(\mathbf{x})$
- ▶ $\mathcal{S}_2 = \{\mathbf{x}_i\}_{i=1}^{n_2} \sim p(\mathbf{x}|\theta)$

Two sample test

$$H_0 : \pi(\mathbf{x}) = p(\mathbf{x}|\theta), \quad H_1 : \pi(\mathbf{x}) \neq p(\mathbf{x}|\theta)$$

Likelihood-free learning

Two sample test

$$H_0 : \pi(\mathbf{x}) = p(\mathbf{x}|\boldsymbol{\theta}), \quad H_1 : \pi(\mathbf{x}) \neq p(\mathbf{x}|\boldsymbol{\theta})$$

Define test statistic $T(\mathcal{S}_1, \mathcal{S}_2)$. Test statictic is likelihood free.
If $T(\mathcal{S}_1, \mathcal{S}_2) < \alpha$, then accept H_0 , else reject it.

Desired behaviour

- ▶ The generative model $p(\mathbf{x}|\boldsymbol{\theta})$ minimizes the value of test statistic $T(\mathcal{S}_1, \mathcal{S}_2)$.
- ▶ Find appropriate test statistic in high dimensions is hard. We could try to learn the appropriate $T(\mathcal{S}_1, \mathcal{S}_2)$.

Vanila GAN

- ▶ **Generator:** latent variable model $\mathbf{x} = G(\mathbf{z})$, which minimizes two-sample test objective.
- ▶ **Discriminator:** function $D(\mathbf{x})$, which distinguish real samples from model samples and maximizes two-sample test statistic.

Objective

$$V(G, D) = \min_G \max_D \mathbb{E}_{\pi(\mathbf{x})} \log D(\mathbf{x}) + \mathbb{E}_{p(\mathbf{z})} \log(1 - D(G(\mathbf{z})))$$

For fixed generator $G(\mathbf{z})$ discriminator is performing binary classification with cross entropy loss.

This minimax game has global optimum $\pi(\mathbf{x}) = p(\mathbf{x}|\theta)$.

<https://arxiv.org/abs/1406.2661>

Vanila GAN optimality

Objective (fixed G)

$$\max_D V(D) = \mathbb{E}_{\pi(\mathbf{x})} \log D(\mathbf{x}) + \mathbb{E}_{p(\mathbf{z})} \log(1 - D(G(\mathbf{z})))$$

Optimal discriminator

$$\begin{aligned} V(G, D) &= \mathbb{E}_{\pi(\mathbf{x})} \log D(\mathbf{x}) + \mathbb{E}_{p(\mathbf{x}|\theta)} \log(1 - D(\mathbf{x})) \\ &= \int \underbrace{[\pi(\mathbf{x}) \log D(\mathbf{x}) + p(\mathbf{x}|\theta) \log(1 - D(\mathbf{x})]}_{y(D)} d\mathbf{x} \end{aligned}$$

$$\frac{dy(D)}{dD} = \frac{\pi(\mathbf{x})}{D(\mathbf{x})} - \frac{p(\mathbf{x}|\theta)}{1 - D(\mathbf{x})} = 0 \quad \Rightarrow \quad D^*(\mathbf{x}) = \frac{\pi(\mathbf{x})}{\pi(\mathbf{x}) + p(\mathbf{x}|\theta)}$$

Vanila GAN optimality

Objective (fixed D)

$$\max_G V(G, D^*) = \mathbb{E}_{\pi(\mathbf{x})} \log D^*(\mathbf{x}) + \mathbb{E}_{p(\mathbf{z})} \log(1 - D^*(G(\mathbf{z})))$$

Optimal generator

$$\begin{aligned} V(G, D^*) &= \mathbb{E}_{\pi(\mathbf{x})} \log \frac{\pi(\mathbf{x})}{\pi(\mathbf{x}) + p(\mathbf{x}|\theta)} + \mathbb{E}_{p(\mathbf{x}|\theta)} \log \frac{p(\mathbf{x}|\theta)}{\pi(\mathbf{x}) + p(\mathbf{x}|\theta)} \\ &= KL \left(\pi(\mathbf{x}) \parallel \frac{\pi(\mathbf{x}) + p(\mathbf{x}|\theta)}{2} \right) + KL \left(p(\mathbf{x}|\theta) \parallel \frac{\pi(\mathbf{x}) + p(\mathbf{x}|\theta)}{2} \right) - 2 \log 2 \\ &= 2JSD(\pi(\mathbf{x}) \parallel p(\mathbf{x}|\theta)) - 2 \log 2. \end{aligned}$$

Vanila GAN optimality

Optimal generator

$$V(G, D^*) = 2JSD(\pi(\mathbf{x}) || p(\mathbf{x}|\theta)) - 2 \log 2.$$

Jensen-Shannon divergence

$$JSD(\pi(\mathbf{x}) || p(\mathbf{x}|\theta)) = \frac{1}{2} \left[KL\left(\pi(\mathbf{x}) || \frac{\pi(\mathbf{x}) + p(\mathbf{x}|\theta)}{2}\right) + KL\left(p(\mathbf{x}|\theta) || \frac{\pi(\mathbf{x}) + p(\mathbf{x}|\theta)}{2}\right) \right]$$

Also called symmetric KL divergence. Could be used as a distance measure!

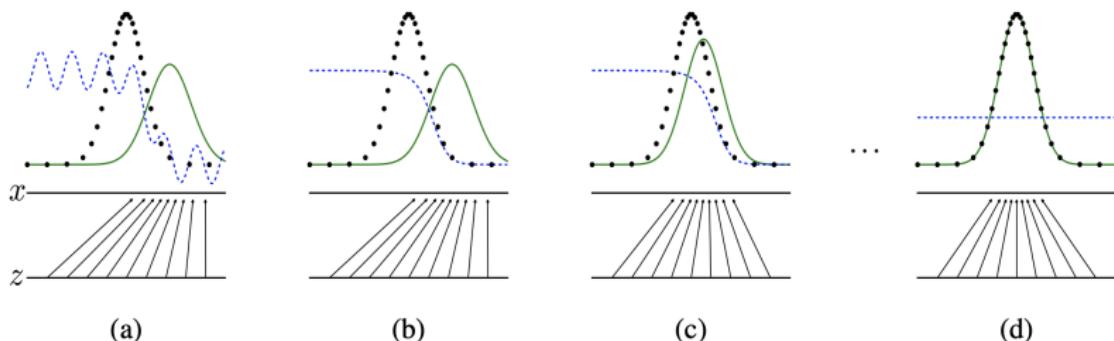
$$V(G^*, D^*) = -2 \log 2, \quad \pi(\mathbf{x}) = p(\mathbf{x}|\theta).$$

If the generator updates are made in function space and discriminator is optimal at every step, then the generator is guaranteed to converge to the data distribution.

Vanila GAN

Objective

$$V(G, D) = \min_G \max_D \mathbb{E}_{\pi(\mathbf{x})} \log D(\mathbf{x}) + \mathbb{E}_{p(\mathbf{z})} \log(1 - D(G(\mathbf{z})))$$



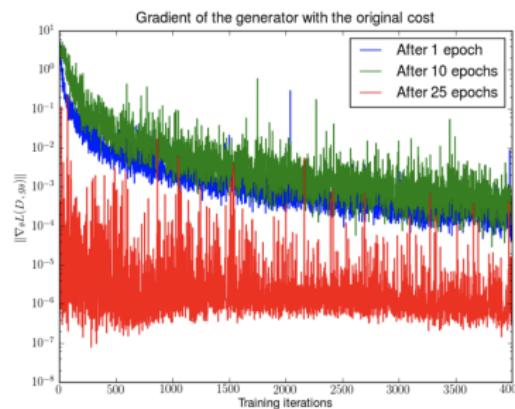
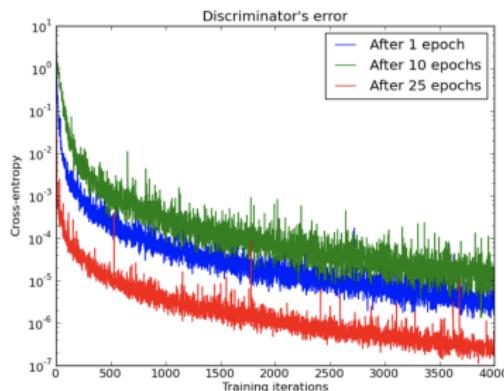
- ▶ Generator updates are made in parameter space.
 - ▶ Discriminator is not optimal at every step.
 - ▶ Generator and discriminator loss keeps oscillating during GAN training.

Vanishing gradients

Objective

$$V(G, D) = \min_G \max_D \mathbb{E}_{\pi(x)} \log D(x) + \mathbb{E}_{p(z)} \log(1 - D(G(z)))$$

Early in learning, G is poor, D can reject samples with high confidence. In this case, $\log(1 - D(G(z)))$ saturates.



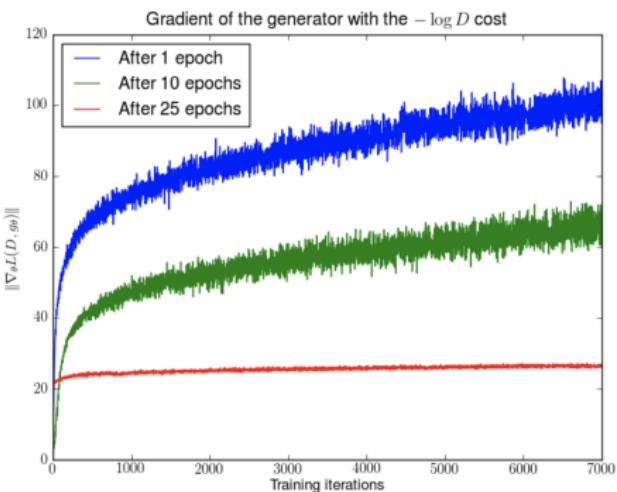
Vanishing gradients

Objective

$$V(G, D) = \min_G \max_D \mathbb{E}_{\pi(x)} \log D(x) + \mathbb{E}_{p(z)} \log(1 - D(G(z)))$$

Maximize $\log D(G(z))$ instead of $\log(1 - D(G(z)))$.

Gradients are getting much stronger, but the training is unstable (with increasing mean and variance).



Likelihood-based vs GAN

GAN objective (for optimal discriminator)

$$V(G, D^*) = 2JSD(\pi(\mathbf{x}) || p(\mathbf{x}|\theta)) - \log 4.$$

Likelihood model objective

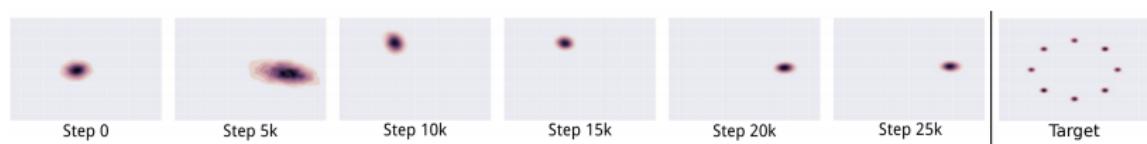
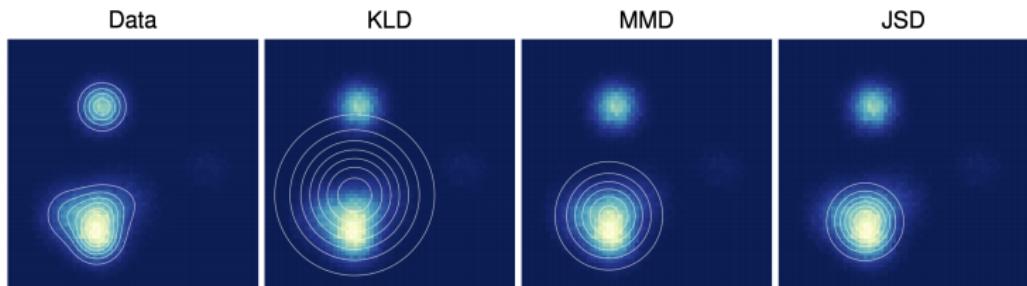
$$\begin{aligned}\max_{\theta} \log p(\mathbf{X}|\theta) &\approx \max_{\theta} \mathbb{E}_{\pi(\mathbf{x})} \log p(\mathbf{x}|\theta) = \\&= \max_{\theta} \mathbb{E}_{\pi(\mathbf{x})} \log p(\mathbf{x}|\theta) + \mathbb{E}_{\pi(\mathbf{x})} \log \pi(\mathbf{x}) = \\&= \max_{\theta} \mathbb{E}_{\pi(\mathbf{x})} \log \frac{p(\mathbf{x}|\theta)}{\pi(\mathbf{x})} = \min_{\theta} KL(\pi(\mathbf{x}) || p(\mathbf{x}|\theta))\end{aligned}$$

What is the difference between JS and KL divergences?

<https://arxiv.org/abs/1511.01844>

Mode collapse

The phenomena where the generator of a GAN collapses to one or few distribution modes.



Alternate architectures, adding regularization terms, injecting small noise perturbations and other millions bags and tricks are used to avoid the mode collapse.

<https://arxiv.org/abs/1406.2661>

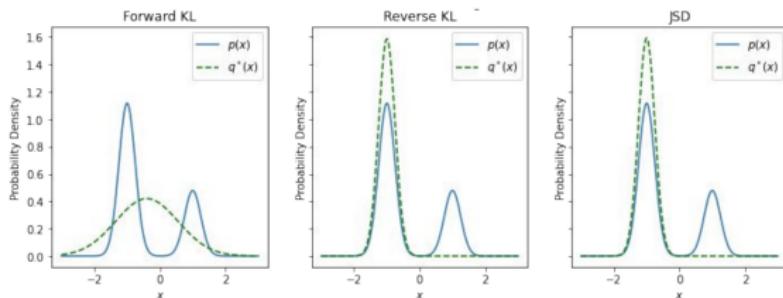
<https://arxiv.org/abs/1611.02163>

Mode collapse

Mode covering vs mode seeking

$$KL(\pi||p) = \int \pi(\mathbf{x}) \log \frac{\pi(\mathbf{x})}{p(\mathbf{x})} d\mathbf{x}, \quad KL(p||\pi) = \int p(\mathbf{x}) \log \frac{p(\mathbf{x})}{\pi(\mathbf{x})} d\mathbf{x}$$

$$JSD(\pi||p) = \frac{1}{2} \left[KL \left(\pi(x) || \frac{\pi(x) + p(x)}{2} \right) + KL \left(p(x) || \frac{\pi(x) + p(x)}{2} \right) \right]$$



<https://sites.google.com/view/berkeley-cs294-158-sp20/home>

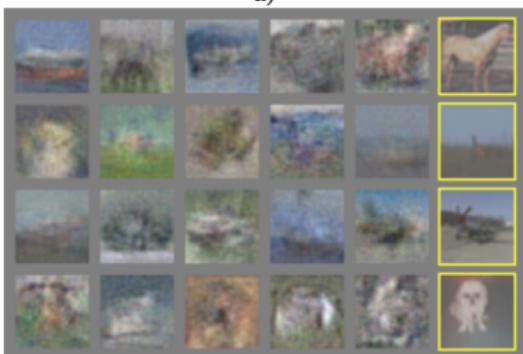
Vanila GAN results



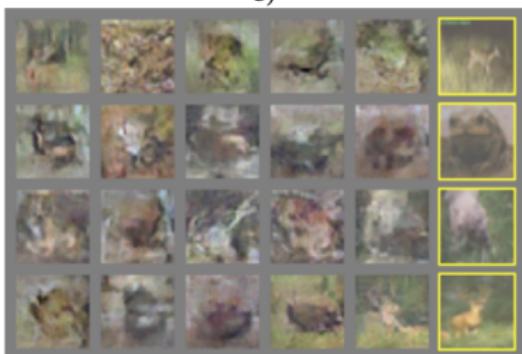
a)



b)

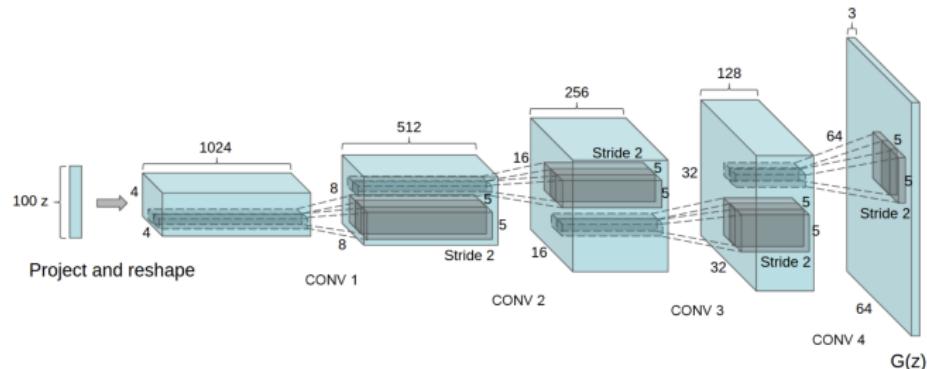


c)



d)

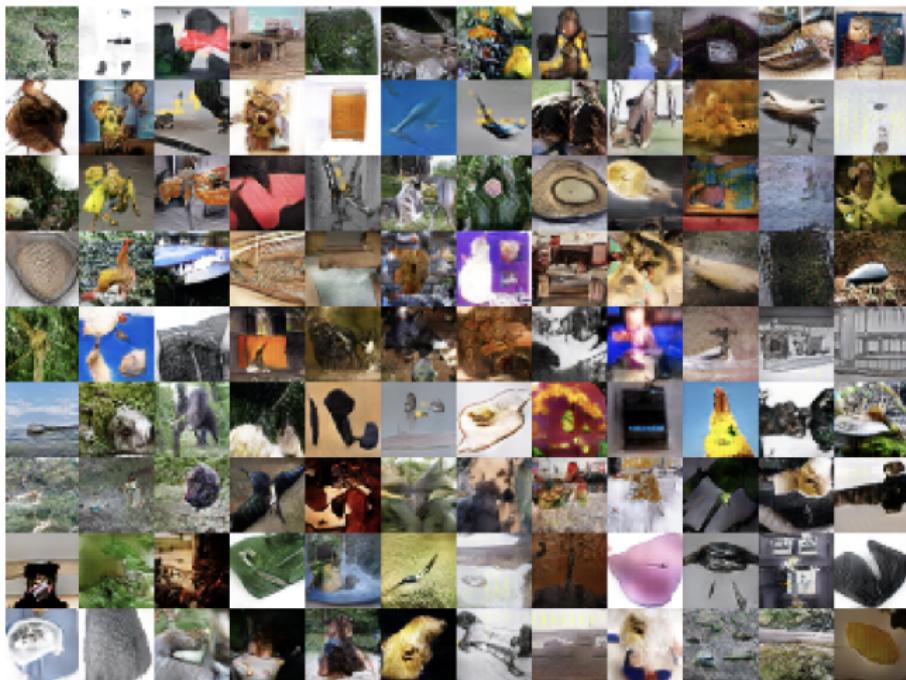
Deep Convolutional GAN



- ▶ Mean-pooling instead of max-pooling.
- ▶ Transposed convolutions in the generator for upsampling.
- ▶ Downsample with strided convolutions and average pooling.
- ▶ ReLU for generator, Leaky-ReLU (0.2) for discriminator.
- ▶ Output nonlinearity: tanh for Generator, sigmoid for discriminator.
- ▶ Batch Normalization used to prevent mode collapse (not applied at the output of G and input of D).
- ▶ Adam: small LR = 2e-4; small momentum: 0.5, batch-size: 128.

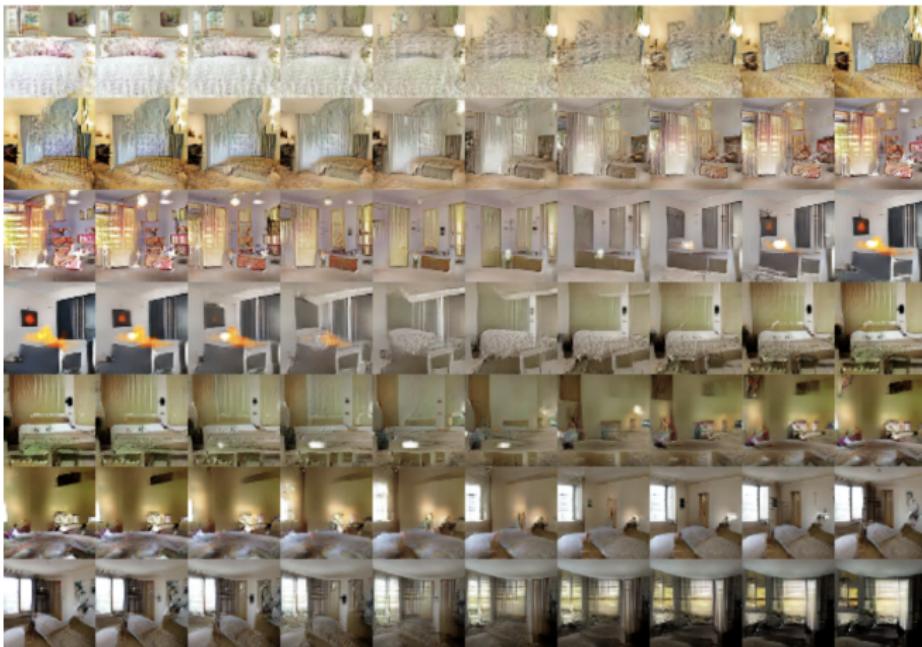
Deep Convolutional GAN

ImageNet samples



Deep Convolutional GAN

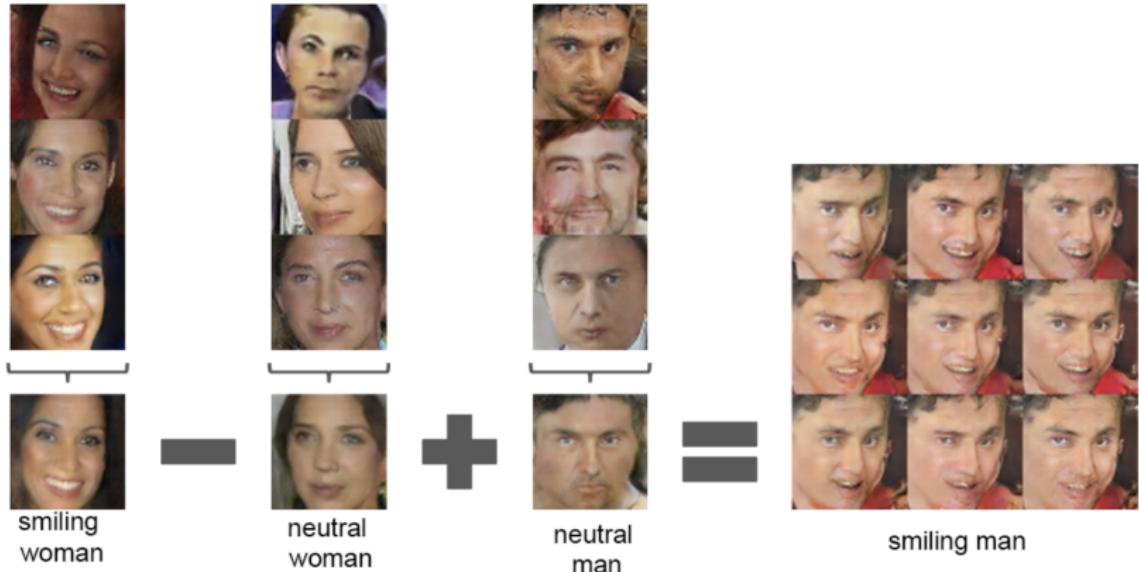
Smooth interpolations



<https://arxiv.org/abs/1511.06434>

Deep Convolutional GAN

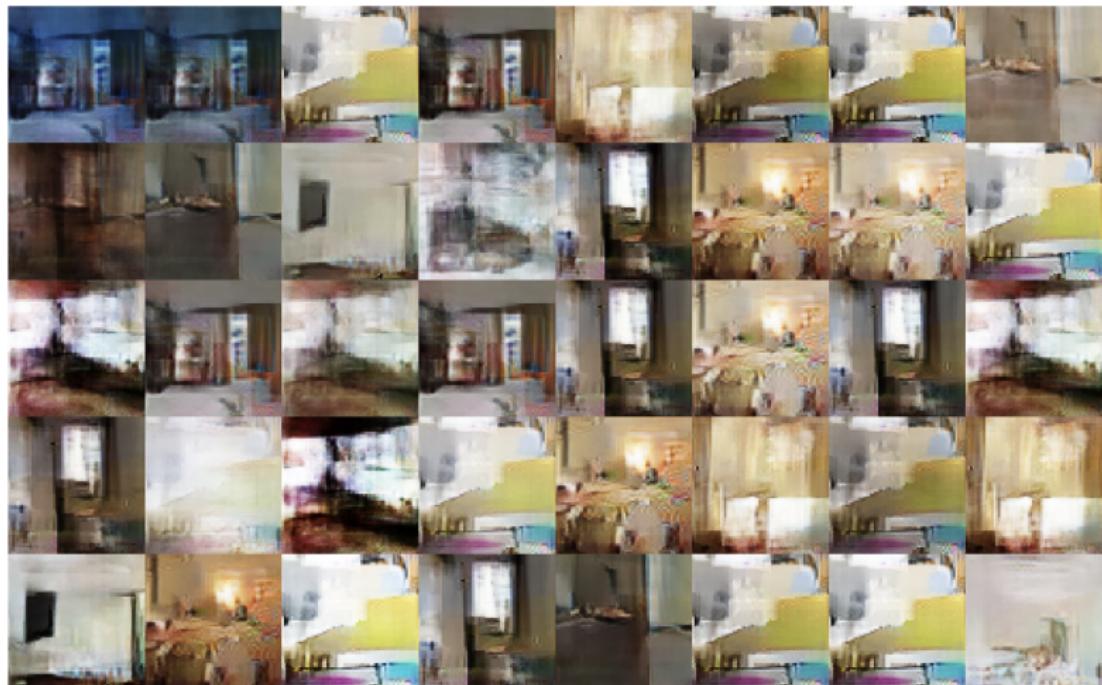
Vector arithmetic



<https://arxiv.org/abs/1511.06434>

Deep Convolutional GAN

Mode collapse



Improved techniques for training GANs

- ▶ Feature matching

$$\mathcal{L}_G = \|\mathbb{E}_{\pi(\mathbf{x})} \mathbf{d}(\mathbf{x}) - \mathbb{E}_{p(\mathbf{z})} \mathbf{d}(G(\mathbf{z}))\|_2^2$$

Here $\mathbf{d}(\mathbf{x})$ – intermediate layer of discriminator. Matching the learned discriminator statistics instead of the output of the discriminator. Helps to avoid the vanishing gradients for sufficiently good discriminator.

- ▶ Historical averaging adds extra loss term for generator and discriminator losses

$$\|\boldsymbol{\theta} - \sum_{t=1}^T \boldsymbol{\theta}_t\|_2^2.$$

Here $\boldsymbol{\theta}_t$ – value of parameters at the previous step t . It allows to stabilize training procedure.

Improved techniques for training GANs

- ▶ One-sided label smoothing. Instead of using one-hot labels in classification, use $(1 - \alpha)$ for real data (the generated samples are not smoothed).

$$D^*(\mathbf{x}) = \frac{(1 - \alpha)\pi(\mathbf{x})}{\pi(\mathbf{x}) + p(\mathbf{x}|\theta)}$$

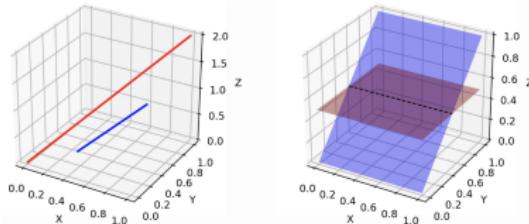
- ▶ Virtual batch normalization. BatchNorm makes samples within minibatch are highly correlated.



Use reference fixed batch to compute the normalization statistics. To avoid overfitting construct batch with the reference batch and the current sample.

Informal theoretical results

- ▶ Since z usually has lower dimensionality compared to x , manifold $G(z)$ has a measure 0 in x space. Hence, support of $p(x|\theta)$ lies on low-dimensional manifold.
- ▶ Distribution of real images $\pi(x)$ is also concentrated on a low dimensional manifold.



- ▶ If $\pi(x)$ and $p(x|\theta)$ have disjoint supports, then there is a smooth optimal discriminator. In this case we won't really be able to learn anything by backproping through it.
- ▶ For such low-dimensional disjoint manifolds

$$KL(\pi||p) = KL(p||\pi) = \infty, \quad JSD(\pi||p) = \log 2$$

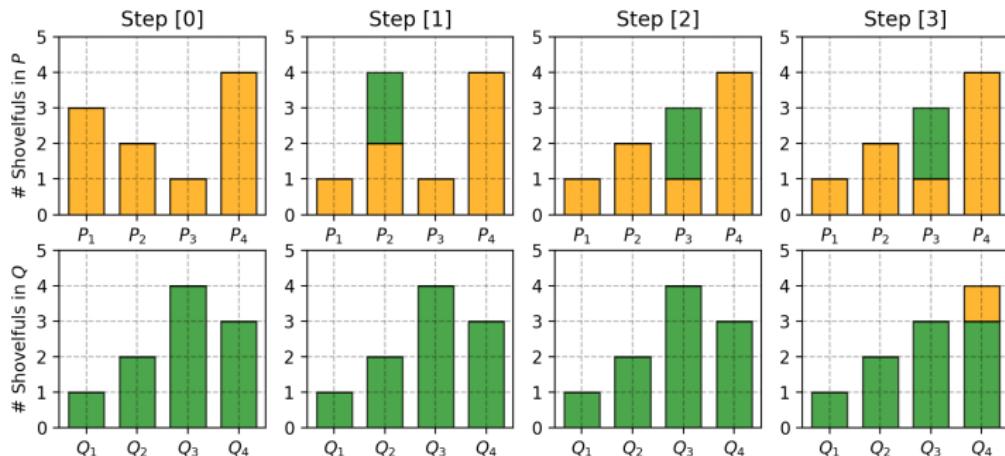
- ▶ Adding continuous noise to the inputs of the discriminator smoothes the distributions of the probability mass.

<https://lilianweng.github.io/lil-log/2017/08/20/from-GAN-to-WGAN.html>

<https://arxiv.org/abs/1701.04862>

Wasserstein distance (discrete)

Also called Earth Mover's distance. The minimum cost of moving and transforming a pile of dirt in the shape of one probability distribution to the shape of the other distribution.



$$W(P, Q) = 2(\text{step 1}) + 2(\text{step 2}) + 1(\text{step 3}) = 5$$

Wasserstein distance

$$W(\pi, p) = \inf_{\gamma \in \Pi(\pi, p)} \mathbb{E}_{(\mathbf{x}, \mathbf{y}) \sim \gamma} \|\mathbf{x} - \mathbf{y}\| = \inf_{\gamma \in \Pi(\pi, p)} \int \|\mathbf{x} - \mathbf{y}\| \gamma(\mathbf{x}, \mathbf{y}) d\mathbf{x} d\mathbf{y}$$

- ▶ $\Pi(\pi, p)$ – the set of all joint distributions $\gamma(\mathbf{x}, \mathbf{y})$ with marginals π and p ($\int \gamma(\mathbf{x}, \mathbf{y}) d\mathbf{x} = p(\mathbf{y})$, $\int \gamma(\mathbf{x}, \mathbf{y}) d\mathbf{y} = \pi(\mathbf{x})$)
- ▶ $\gamma(\mathbf{x}, \mathbf{y})$ – transportation plan (the amount of "dirt" that should be transported from point \mathbf{x} to point \mathbf{y}).
- ▶ $\gamma(\mathbf{x}, \mathbf{y})$ – the amount, $\|\mathbf{x} - \mathbf{y}\|$ – the distance.

For better understanding of transportation plan function γ , try to write down the plan for previous discrete case.

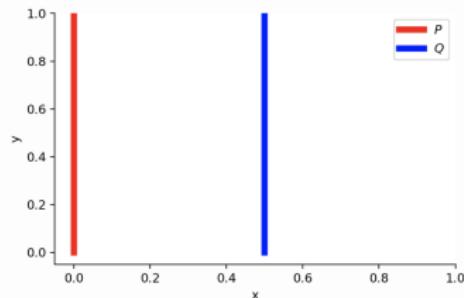
<https://arxiv.org/abs/1701.07875>

Wasserstein distance vs KL vs JSD

Consider 2d distributions

$$\pi(x, y) = (0, U[0, 1])$$

$$p(x, y|\theta) = (\theta, U[0, 1])$$



- ▶ $\theta = 0$. Distributions are the same

$$KL(\pi||p) = KL(p||\pi) = JSD(p||pi) = W(\pi, p) = 0$$

- ▶ $\theta \neq 0$

$$KL(\pi||p) = \int_{U[0,1]} 1 \log \frac{1}{0} dy = \infty = KL(\pi||p)$$

$$JSD(\pi||p) = \frac{1}{2} \left(\int_{U[0,1]} 1 \log \frac{1}{1/2} dy + \int_{U[0,1]} 1 \log \frac{1}{1/2} dy \right) = \log 2$$

$$W(\pi, p) = |\theta|$$

Wasserstein distance vs KL vs JSD

Theorem 1

Let $G(\mathbf{z}, \theta)$ be any feedforward neural network, and $p(\mathbf{z})$ a prior over \mathbf{z} such that $\mathbb{E}_{\mathbf{z} \sim p(\mathbf{z})} \|\mathbf{z}\| < \infty$. Then therefore $W(\pi, p)$ is continuous everywhere and differentiable almost everywhere.

Theorem 2

Let π be a distribution on a compact space \mathcal{X} and $\{p_t\}_{t=1}^{\infty}$ be a sequence of distributions on \mathcal{X} .

$$KL(\pi || p_t) \rightarrow 0 \text{ (or } KL(p_t || \pi) \rightarrow 0) \quad (1)$$

$$JSD(\pi || p_t) \rightarrow 0 \quad (2)$$

$$W(\pi || p_t) \rightarrow 0 \quad (3)$$

Then, considering limits as $t \rightarrow \infty$, (1) implies (2), (2) implies (3).

Wasserstein GAN

Wasserstein distance

$$W(\pi, p) = \inf_{\gamma \in \prod(\pi, p)} \mathbb{E}_{(\mathbf{x}, \mathbf{y}) \sim \gamma} \|\mathbf{x} - \mathbf{y}\| = \inf_{\gamma \in \prod(\pi, p)} \int \|\mathbf{x} - \mathbf{y}\| \gamma(\mathbf{x}, \mathbf{y}) d\mathbf{x} d\mathbf{y}$$

The infimum across all possible joint distributions in $\prod(\pi, p)$ is intractable.

Kantorovich-Rubinstein duality

$$W(\pi, p) = \frac{1}{K} \max_{\|f\|_L \leq K} [\mathbb{E}_{\mathbf{x} \sim \pi} f(\mathbf{x}) - \mathbb{E}_{\mathbf{x} \sim p} f(\mathbf{x})],$$

where $\|f\|_L \leq K$ are K -Lipschitz continuous functions
 $(f : \mathcal{X} \rightarrow \mathbb{R})$

$$|f(\mathbf{x}_1) - f(\mathbf{x}_2)| \leq K \|\mathbf{x}_1 - \mathbf{x}_2\|, \quad \text{for all } \mathbf{x}_1, \mathbf{x}_2 \in \mathcal{X}.$$

Wasserstein GAN

Kantorovich-Rubinstein duality

$$W(\pi, p) = \frac{1}{K} \max_{\|f\|_L \leq K} [\mathbb{E}_{\mathbf{x} \sim \pi} f(\mathbf{x}) - \mathbb{E}_{\mathbf{x} \sim p} f(\mathbf{x})],$$

- ▶ Now we have to ensure that f is K -Lipschitz continuous.
- ▶ Let make $f(\mathbf{x}, \phi)$ parametrized by parameters ϕ .
- ▶ If parameters ϕ lies in a compact set Φ then $f(\mathbf{x}, \phi)$ will be K -Lipschitz continuous function.
- ▶ Let clamp the parameters to a fixed box $\Phi \in [-0.01, 0.01]^d$ after each gradient update.

$$\max_{\phi \in \Phi} [\mathbb{E}_{\mathbf{x} \sim \pi} f(\mathbf{x}, \phi) - \mathbb{E}_{\mathbf{x} \sim p} f(\mathbf{x}, \phi)] \leq \max_{\|f\|_L \leq K} [\mathbb{E}_{\mathbf{x} \sim \pi} f(\mathbf{x}) - \mathbb{E}_{\mathbf{x} \sim p} f(\mathbf{x})] = KW(\pi || p)$$

Wasserstein GAN

Vanilla GAN objective

$$\min_G \max_D \mathbb{E}_{\pi(x)} \log D(x) + \mathbb{E}_{p(z)} \log(1 - D(G(z)))$$

WGAN objective

$$\min_G W(\pi, p) = \min_G \frac{1}{K} \max_{\phi \in \Phi} [\mathbb{E}_{x \sim \pi} f(x, \phi) - \mathbb{E}_{z \sim p} f(G(z), \phi)].$$

- ▶ Discriminator D is similar to the function f , but not the same (it is not a classifier anymore). In the WGAN model, function f is usually called *critic*.
- ▶ "Weight clipping is a clearly terrible way to enforce a Lipschitz constraint". If the clipping parameter is large, it is hard to train the critic till optimality. If the clipping parameter is too small, it could lead to vanishing gradients.

Wasserstein GAN

Algorithm 1 WGAN, our proposed algorithm. All experiments in the paper used the default values $\alpha = 0.00005$, $c = 0.01$, $m = 64$, $n_{\text{critic}} = 5$.

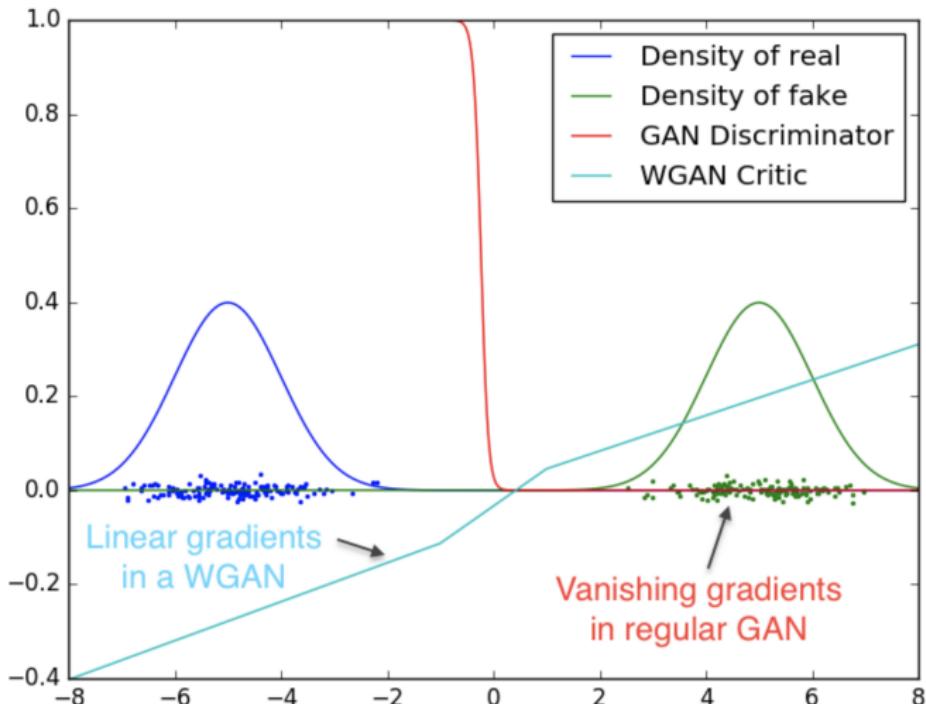
Require: : α , the learning rate. c , the clipping parameter. m , the batch size.

n_{critic} , the number of iterations of the critic per generator iteration.

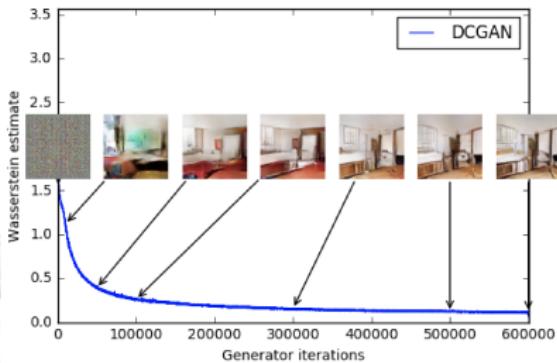
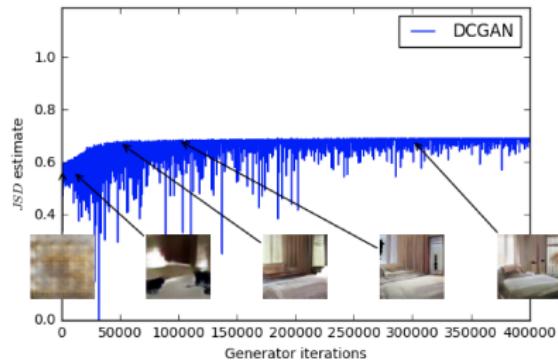
Require: : w_0 , initial critic parameters. θ_0 , initial generator's parameters.

```
1: while  $\theta$  has not converged do
2:   for  $t = 0, \dots, n_{\text{critic}}$  do
3:     Sample  $\{x^{(i)}\}_{i=1}^m \sim \mathbb{P}_r$  a batch from the real data.
4:     Sample  $\{z^{(i)}\}_{i=1}^m \sim p(z)$  a batch of prior samples.
5:      $g_w \leftarrow \nabla_w \left[ \frac{1}{m} \sum_{i=1}^m f_w(x^{(i)}) - \frac{1}{m} \sum_{i=1}^m f_w(g_\theta(z^{(i)})) \right]$ 
6:      $w \leftarrow w + \alpha \cdot \text{RMSProp}(w, g_w)$ 
7:      $w \leftarrow \text{clip}(w, -c, c)$ 
8:   end for
9:   Sample  $\{z^{(i)}\}_{i=1}^m \sim p(z)$  a batch of prior samples.
10:   $g_\theta \leftarrow -\nabla_\theta \frac{1}{m} \sum_{i=1}^m f_w(g_\theta(z^{(i)}))$ 
11:   $\theta \leftarrow \theta - \alpha \cdot \text{RMSProp}(\theta, g_\theta)$ 
12: end while
```

Wasserstein GAN



Wasserstein GAN



- ▶ JSD correlates poorly with the sample quality. Stays constant nearly maximum value $\log 2 \approx 0.69$.
- ▶ W is highly correlated with the sample quality.

<https://arxiv.org/abs/1701.07875>

Wasserstein GAN

WGAN converged without batch norm and constant number of filters



"In no experiment did we see evidence of mode collapse for the WGAN algorithm."



<https://arxiv.org/abs/1701.07875>

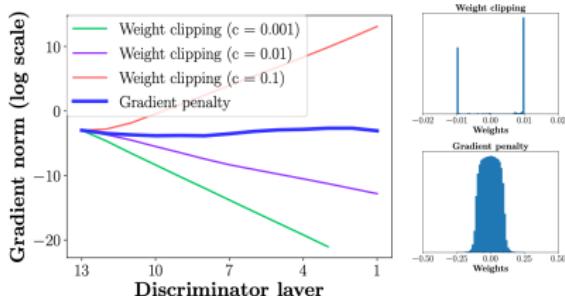
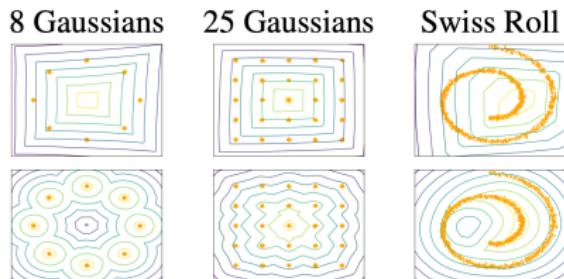
Wasserstein GAN with Gradient Penalty

The generator distribution is fixed and equals to the real distribution + Gaussian noise.

Problems with weight clipping:

- ▶ The critic ignores higher moments of the data distribution.
- ▶ The gradient either grow or decay exponentially.

Gradient penalty makes the gradients more stable.



<https://arxiv.org/abs/1704.00028>

Wasserstein GAN with Gradient Penalty

<https://arxiv.org/abs/1704.00028>

References

- ▶ A note on the evaluation of generative models
<https://arxiv.org/abs/1511.01844>
Summary:
- ▶ Generative Adversarial Nets
<https://arxiv.org/abs/1406.2661>
Summary:
- ▶ Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks
<https://arxiv.org/abs/1511.06434>
Summary:
- ▶ Improved techniques for training GANs
<https://arxiv.org/abs/1606.03498>
Summary:
- ▶ Towards principled methods for training generative adversarial networks
<https://arxiv.org/abs/1701.04862>
Summary:
- ▶ Wasserstein GAN
<https://arxiv.org/abs/1701.07875>
Summary:
- ▶ Improved Training of Wasserstein GANs
<https://arxiv.org/abs/1704.00028>
Summary: