Generative Adversarial Networks

by Paul Hand
Northeastern University

Outline

- GANs - examples and properties
- Minimax formulation and theory
- Wasserstein GANs
- Challenges

Generative Adversarial Networks

(Goodfellow et al. 2014)

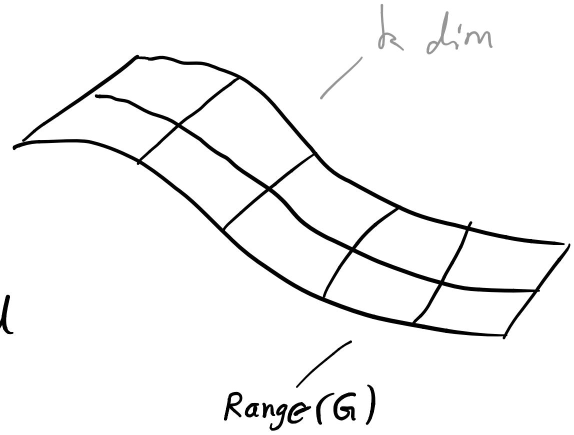
Generative model trained in a game-theoretic adversarial way

$G: \mathbb{R}^k \to \mathbb{R}^n$ st if $z \sim N(0, I_k)$ then $G(z)$ samples from a learned data distribution

latent space

image space

While G induces a distribution on $\mathbb{R}^n$, we will not attempt to maximize data likelihood



Assume that every data point is G(z) for some z ~ N(0,I)

What is a generative model?

Can sample from an approximation of a probability distribution.
Can convert a random sample as input (encoding) and can output an image

You have sample access to a unknown probability distribution. Given those samples, you want to learn the distribution in a way that allows generation of new samples.

What can they be used for?

Could generate synthetic training data (for example, could use the GAN for data augmentation)

Active learning - too expensive to label all data, use an algorithm to decide which data points are most worthwhile to be labeled - could use a GAN to synthesize a synthetic point to be optimally informative

Cheap way to sample from an otherwise complicated distribution

Image manipulation - generative model knows what the set of faces look like, and you could find the closest image in that set to some desired image

GPT-2 or 3 - Could generate art / poetry / etc. Or to build chat bot / dialogue

Why can we not easily train likelihood with a GAN as described above?

$\{x_i\}_{i=1}^n$ iid samples of D

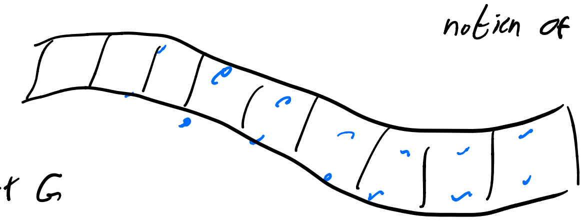
For any θ
$G_\theta(z)$ is a random var. Has notion of likelihood.

Why cant we take
$\max_\theta L(x_i;\theta)$

likelihood

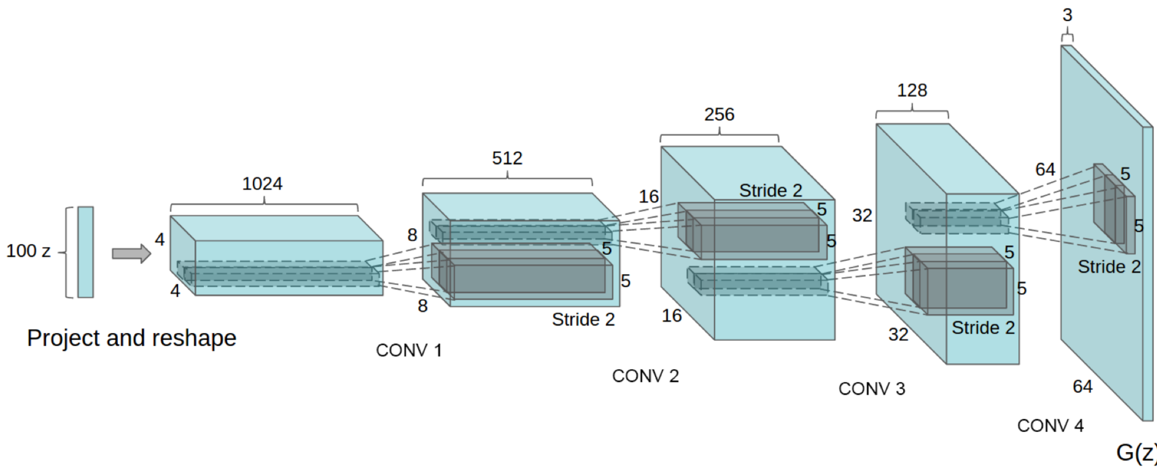
sample

params of G



Off manifold, $L = 0$. $\nabla L = 0$

Example architecture (DCGAN) (Radford et al. 2016)



Synthetic Samples when trained on LSUN Bedrooms:

Can interpolate in latent space:

original image — interpolations in z — original image

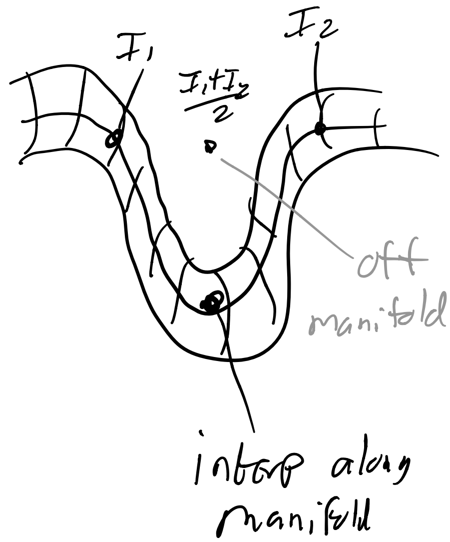
(Ulyahov et al. 2017)

Geometric Visualization

$$G: \mathbb{R}^k \to \mathbb{R}^n$$

$\mathbb{R}^k$: move smoothly

$\mathbb{R}^n$: semantically continuous deformations



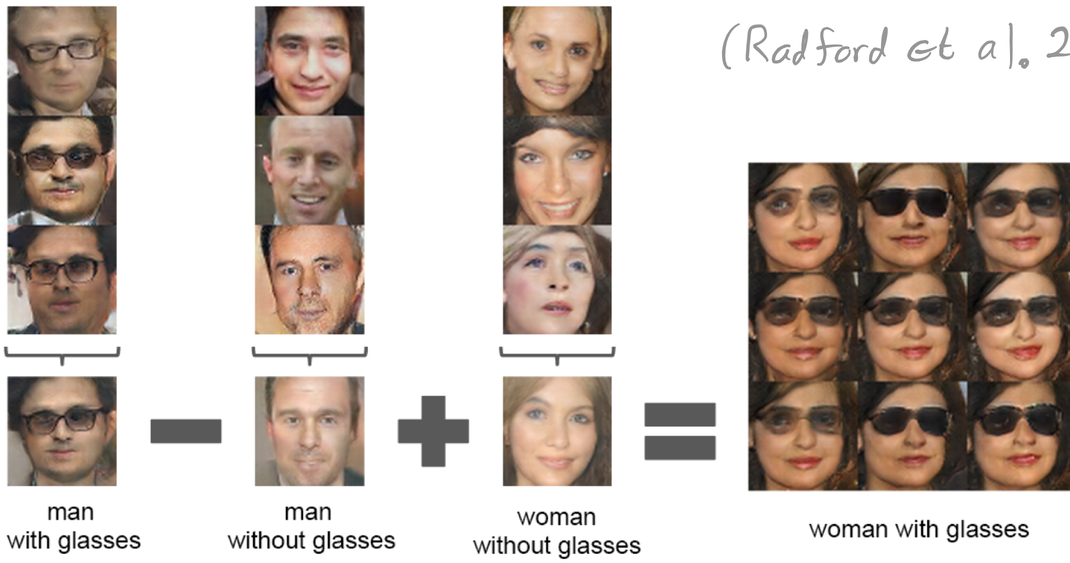
How do we know that a generative model didn't just memorize the training data (or provide trivial variations of the training data)?

We could demonstrate that interpolations between images are in the range of the model. We are sure that interpolating faces are not in the training dataset

You could generate a new sample and then find the most similar images (Take a the representation of all training images with respect to a hidden layer's activation of a classification network) in the training data to that sample. Visually inspect them.

There is semantically meaningful arithmetic in latent space:

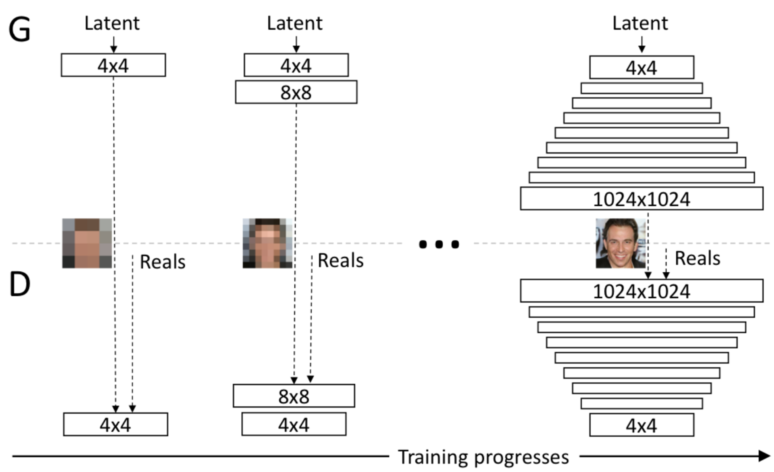
(Radford et al. 2016)



There is a direction in z corresponding to having glasses

GANs have been trained that can generate photorealistic faces

(Korras et al. 2018)



One hour of imaginary celebrities

https://youtu.be/36lE9tV9vm0

Game Theory - Example - Rock Paper Scissors

Values for P_1

$$\begin{array}{cc|ccc} & & & P_2 & \\ & & \text{Rock} & \text{Paper} & \text{Scissors} \\ \hline & \text{Rock} & 0 & -1 & 1 \\ P_1 & \text{Paper} & 1 & 0 & -1 \\ & \text{Scissors} & -1 & 1 & 0 \end{array}$$

matrix A

Suppose P_2 chooses a prob. dist. $y \in \mathbb{R}^3$

— P_1 — — — — $x \in \mathbb{R}^3$

Expected payoff to P_1 is $x^t A y$

P_1 wants max over x

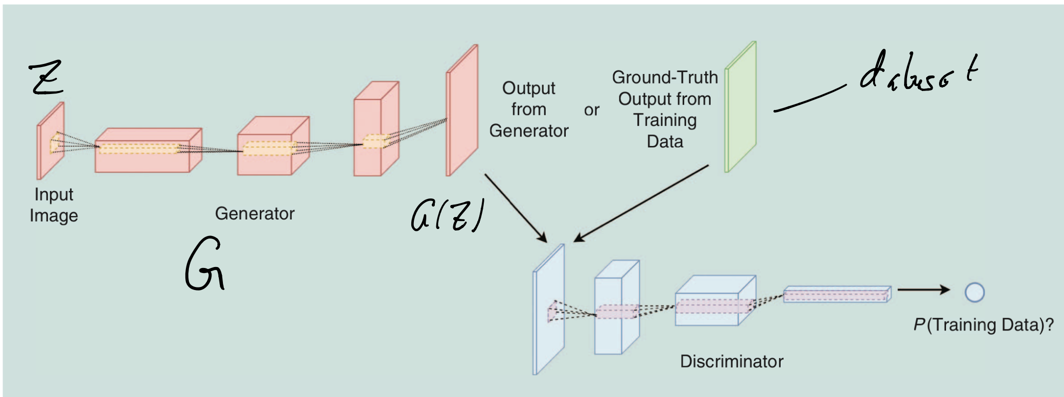
P_2 wants min over y

minimax problem

Jointly Optimal Outcome:

$$\min_y \max_x x^t A y$$

Idea: Train a model by trying to fool a concurrently trained discriminator



(Lucas et al. 2018)

Question: Is training a GAN a supervised or unsupervised learning problem?

Unsupervised. Only had samples {x_i} of a training distribution. No one labeled them.

Question: Is the output of a GAN more likely to be a superset of the training distribution or a subset of the training distribution?

Subset

If the output of the GAN were far from the training distribution, then the discriminator would learn to identify it.

Formulation of GAN training as minimax optimization

Let P_d denote data distribution
P_z be $\mathcal{N}(0, I_k)$

Let $G: \mathbb{R}^k \to \mathbb{R}^n$ be the generator
$D: \mathbb{R}^n \to [0,1]$ be $P(\text{input is real})$

Value function

$$V(D,G) = \mathbb{E}_{x \sim P_d} \log D(x) + \mathbb{E}_{z \sim P_z} \log\left(1 - D(G(z))\right)$$

Why optimize this?

it is the negative cross-entropy loss
but label = real when $x \sim p_d$
and label = not real when $z \sim p_z$

Cross entropy loss

$$\ell_{CE}(p, q) = -\sum_{s \in S} p(s) \log q(s) = -\mathbb{E}_p(\log q)$$

r.v.s over S

<u>Minimax formulation</u>

$$\min_G \max_D \mathbb{E}_{x \sim p_d} \log D(x) + \mathbb{E}_{z \sim p_z} \log(1 - D(G(z)))$$

D wants to maximize neg. cross-entropy

G wants the opposite

Question: Isn't it intractable to compute $\mathbb{E}_{x \sim p_d} \log D(x)$?

Evaluating $\mathbb{E}$ req. integral, which is intractable,
can perform sampling

Run SGD. Estimate that is exact in expectation

Minibatch Stochastic Gradient Descent Algorithm

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 $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
    • Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ 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 $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ 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.

— perhaps multiple updates to D per update to G

(Goodfellow et al. 2014)

Question: Why are there a different number of update steps for D than for G?

Need to balance the performance of discriminator and generators.

With perfect discriminator, there is no signal for the generator.

Why is the GAN value function the right thing to optimize?

Claim: For fixed G, the optimal D is

$$D^*_G(x) = \frac{P_d(x)}{P_d(x) + P_g(x)}$$

Proof:

$$V(G,D) = \mathbb{E}_{x \sim P_d} \log D(x) + \mathbb{E}_{z \sim P_z} \log(1 - D(G(z)))$$

$$= \mathbb{E}_{x \sim P_d} \log D(x) + \mathbb{E}_{x \sim P_g} \log(1 - D(x))$$

distribution induced by generator

$$= \int_{X} \Big(P_d(x) \log D(x) + P_g(x) \log(1 - D(x)) \Big) \, dx$$

To find max over D:

Use Variational Calculus and differentiate with respect to D and set equal to 0

$$\frac{P_d(x)}{D(x)} - \frac{P_g(x)}{1 - D(x)} \equiv 0$$

$$\Rightarrow D^*(x) = \frac{P_d(x)}{P_d(x) + P_g(x)} \quad \blacksquare$$

Theorem: The global minimum of

$$C(G) = \max_D V(G, D)$$

is unique and achieved iff $P_g = P_d$.

Proof: By previous claim,

$$C(G) = \mathbb{E}_{x \sim P_d} \log D_G^*(x) + \mathbb{E}_{x \sim P_g} \log(1 - D_G^*(x))$$

$$= \mathbb{E}_{x \sim P_d} \log P_d \frac{2}{P_d + P_g} + \mathbb{E}_{x \sim P_g} \log P_g \frac{2}{P_d + P_g} - \log 4$$

$$= -\log 4 + \underbrace{D_{KL}\left(P_d \,\Big\|\, \frac{P_d + P_g}{2}\right) + D_{KL}\left(P_g \,\Big\|\, \frac{P_d + P_g}{2}\right)}_{\text{Jensen Shannon Divergence}}$$

nonnegative and 0 iff $P_d = P_g$ ∎

Limits on this theory:

Nonparametric, infinite capacity models (all probability distributions)

Does not assure the minimax problem can be solved to global optimality

Are GANs Created Equal? A Large-Scale Study

Mario Lucic[⋆] **Karol Kurach**[⋆] **Marcin Michalski** **Olivier Bousquet** **Sylvain Gelly**
Google Brain

Table 1: Generator and discriminator loss functions. The main difference whether the discriminator outputs a probability (MM GAN, NS GAN, DRAGAN) or its output is unbounded (WGAN, WGAN GP, LS GAN, BEGAN), whether the gradient penalty is present (WGAN GP, DRAGAN) and where is it evaluated.

GAN	Discriminator Loss	Generator Loss
MM GAN	$\mathcal{L}_{\mathrm{D}}^{\mathrm{GAN}} = -\mathbb{E}_{x\sim p_d}[\log(D(x))] - \mathbb{E}_{\hat{x}\sim p_g}[\log(1-D(\hat{x}))]$	$\mathcal{L}_{\mathrm{G}}^{\mathrm{GAN}} = \mathbb{E}_{\hat{x}\sim p_g}[\log(1-D(\hat{x}))]$
NS GAN	$\mathcal{L}_{\mathrm{D}}^{\mathrm{NSGAN}} = -\mathbb{E}_{x\sim p_d}[\log(D(x))] - \mathbb{E}_{\hat{x}\sim p_g}[\log(1-D(\hat{x}))]$	$\mathcal{L}_{\mathrm{G}}^{\mathrm{NSGAN}} = -\mathbb{E}_{\hat{x}\sim p_g}[\log(D(\hat{x}))]$
WGAN	$\mathcal{L}_{\mathrm{D}}^{\mathrm{WGAN}} = -\mathbb{E}_{x\sim p_d}[D(x)] + \mathbb{E}_{\hat{x}\sim p_g}[D(\hat{x})]$	$\mathcal{L}_{\mathrm{G}}^{\mathrm{WGAN}} = -\mathbb{E}_{\hat{x}\sim p_g}[D(\hat{x})]$
WGAN GP	$\mathcal{L}_{\mathrm{D}}^{\mathrm{WGANGP}} = \mathcal{L}_{\mathrm{D}}^{\mathrm{WGAN}} + \lambda\mathbb{E}_{\hat{x}\sim p_g}[(\vert\vert\nabla D(\alpha x + (1-\alpha\hat{x})\vert\vert_2 - 1)^2]$	$\mathcal{L}_{\mathrm{G}}^{\mathrm{WGANGP}} = -\mathbb{E}_{\hat{x}\sim p_g}[D(\hat{x})]$
LS GAN	$\mathcal{L}_{\mathrm{D}}^{\mathrm{LSGAN}} = -\mathbb{E}_{x\sim p_d}[(D(x)-1)^2] + \mathbb{E}_{\hat{x}\sim p_g}[D(\hat{x})^2]$	$\mathcal{L}_{\mathrm{G}}^{\mathrm{LSGAN}} = -\mathbb{E}_{\hat{x}\sim p_g}[(D(\hat{x}-1))^2]$
DRAGAN	$\mathcal{L}_{\mathrm{D}}^{\mathrm{DRAGAN}} = \mathcal{L}_{\mathrm{D}}^{\mathrm{GAN}} + \lambda\mathbb{E}_{\hat{x}\sim p_d+\mathcal{N}(0,c)}[(\vert\vert\nabla D(\hat{x})\vert\vert_2 - 1)^2]$	$\mathcal{L}_{\mathrm{G}}^{\mathrm{DRAGAN}} = \mathbb{E}_{\hat{x}\sim p_g}[\log(1-D(\hat{x}))]$
BEGAN	$\mathcal{L}_{\mathrm{D}}^{\mathrm{BEGAN}} = \mathbb{E}_{x\sim p_d}[\vert\vert x - \mathrm{AE}(x)\vert\vert_1] - k_t\mathbb{E}_{\hat{x}\sim p_g}[\vert\vert\hat{x} - \mathrm{AE}(\hat{x})\vert\vert_1]$	$\mathcal{L}_{\mathrm{G}}^{\mathrm{BEGAN}} = \mathbb{E}_{\hat{x}\sim p_g}[\vert\vert\hat{x} - \mathrm{AE}(\hat{x})\vert\vert_1]$

Many formulations of GANs.
Why use a NS GAN instead of a MM GAN?

non saturating — minimax "Vanilla"

Vanishing gradients early in training. Mathematical sketch

$\log(1 - D(\hat{x}))$

$\nabla_\theta \log(1 - D(\hat{x})) = \frac{\nabla D(\hat{x})}{1 - D(\hat{x})}$ $\frac{\text{small}}{4}$

grad will be small

$\nabla \log(D(\hat{x}))$

$= \frac{\nabla D(\hat{x}) \text{ - small}}{D(\hat{x}) \text{ - small}}$

much larger

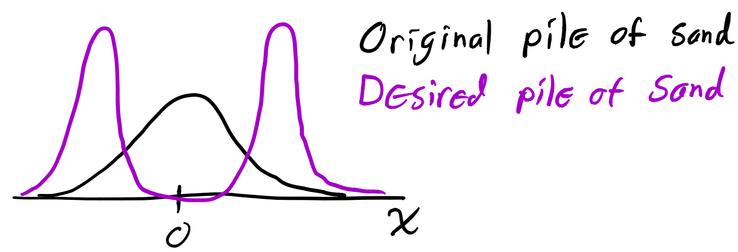
Wasserstein GAN

(Arjovsky et al. 2017)

Goal: minimize "distance" between P_d and P_g

Use Earth mover distance
(Wasserstein-1 distance)

Illustration:



Move each grain such that average distance moved is minimized

Formally,

$$W(P_d, P_g) = \inf_{\gamma \in \Pi(P_d, P_g)} \mathbb{E}_{(x,y)\sim\gamma} \|x-y\|$$

w/ $\Pi(P_d, P_g) = \left\{ \begin{array}{l} \text{joint distributions on } (x,y) \\ \text{s.t. marginals are } P_d \text{ and } P_g \end{array} \right\}$

Visualization of transport plan Pi

Why minimize EMD?

Plain GAN (earlier) roughly minimizes

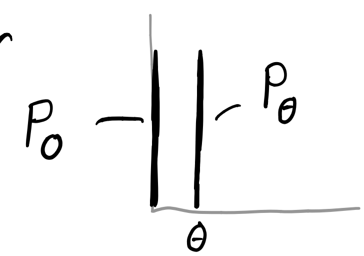
$$D_{KL}\left(P_d \,\|\, \frac{P_d+P_g}{2}\right) + D_{KL}\left(P_g \,\|\, \frac{P_d+P_g}{2}\right) = JS(P_d, P_g)$$

Jensen-Shannon divergence

This is not continuous in P_d and P_g, but EMD is.

Example:

Consider uniform distribution over the 2d line segment
$P_\theta = \{(\theta, y) \mid 0 \le y \le 1\} \subset \mathbb{R}^2$



$$KL(P_0, P_\theta) = \begin{cases} \infty & \theta \neq 0 \\ 0 & \theta = 0 \end{cases}$$

$$JS(P_0, P_\theta) = \begin{cases} \log 2 & \theta \neq 0 \\ 0 & \theta = 0 \end{cases}$$

$$W(P_0, P_\theta) = |\theta|$$

As $\theta \to 0$, only $W(P_0, P_\theta) \to 0$.

Approximating EMD w/ nets

By Kantorovich-Rubinstein duality

$$W(P_d, P_g) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x \sim P_d} f(x) - \mathbb{E}_{x \sim P_g} f(x)$$

Lipschitz constant: $\|f\|_L = \sup_{x \neq y} \frac{\|f(x) - f(y)\|}{\|x - y\|}$

At the expense of a factor of K, can take sup over $\|f\|_L \leq K$

To estimate $W(P_d, P_g)$:

$$\max_{w \in W} \mathbb{E}_{x \sim P_d} f_w(x) - \mathbb{E}_{z \sim P_z} f_w(G_\theta(z))$$

Where f_w are neural nets w/ parameters w in a compact set W.

Eg each weight is in $[-0.01, 0.01]$

WGAN formulation

$$\min_{w} \max_{\theta} \mathbb{E}_{x \sim p_d} f_w(x) - \mathbb{E}_{z \sim p_z} f_w(G_\theta(z))$$

Call f_w the "critic"

Algorithm:

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.

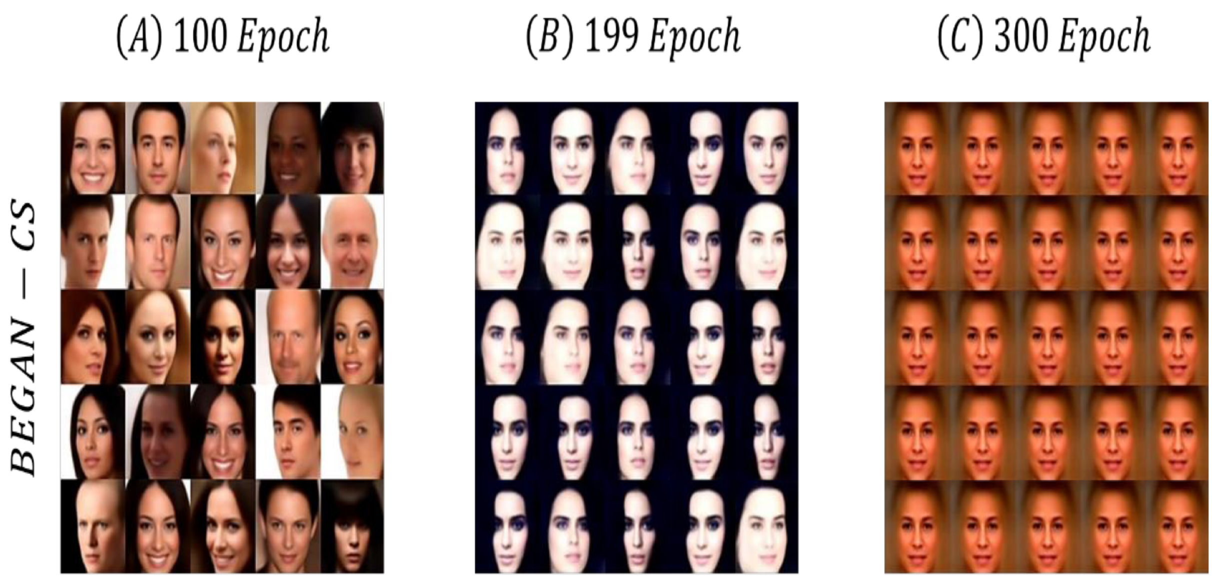
Require: : w_0, initial critic parameters. θ_0, initial generator's parameters.

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

Challenges with GANs:

- Difficulty in training (Eg # D updates per G update)

- Mode collapse



(Park et al. 2020)

- No evaluation metric
- No likelihood estimates
- Difficult to invert

$$\min_{z} \| G(z) - y \|^2$$

How would you evaluate the quality of a GAN?

Qualitative assessments - you look at it and see if they look right
You look for systematic issues (drop in StyleGAN)

You could evaluate the average reconstruction error from a test set and the range of the GAN

How would you invert a GAN?