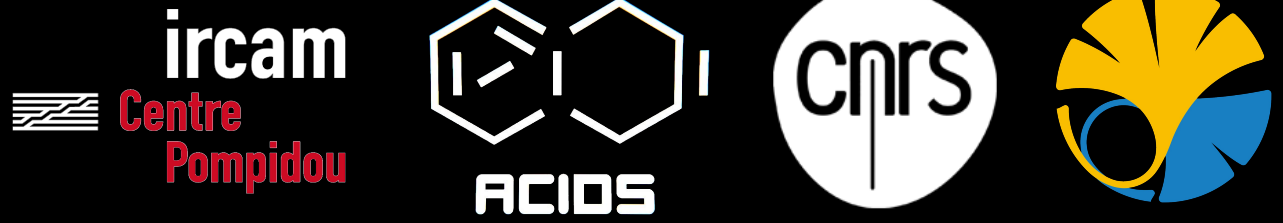


Generative Adversarial Networks

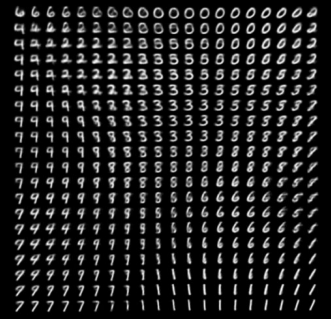
Creative Machine Learning - Course 09

Pr. Philippe Esling
esling@ircam.fr

Brief history of AI

Families of generative models that we will learn

1

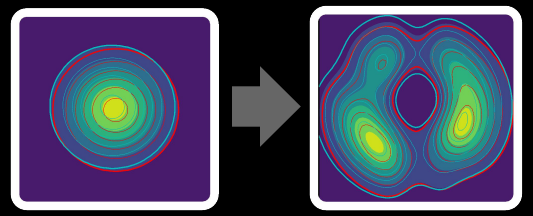


Variational Auto-Encoder (VAE)

Random variables, distributions, independence

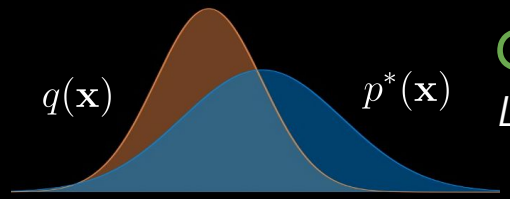
Normalizing flows

Bayes' theorem, likelihood, conjugate priors



2

3



Generative Adversarial Network (GAN)

Latent variables, probability graphs

Diffusion models

Latent variables, probability graphs

4

2015 - *Generative model*

First wave of interest in generating data
Led to current model craze (VAEs, GANs, Diffusion)

2012 onwards Deep learning era

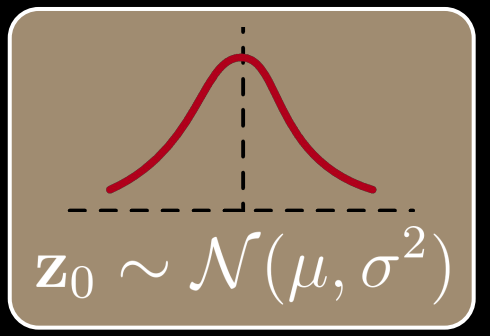
Approximate inference

Probabilistic inference deals with simple distributions

- Provide easier analytical solutions
- Implicit assumption of a simple explanation (capacity)

Issues

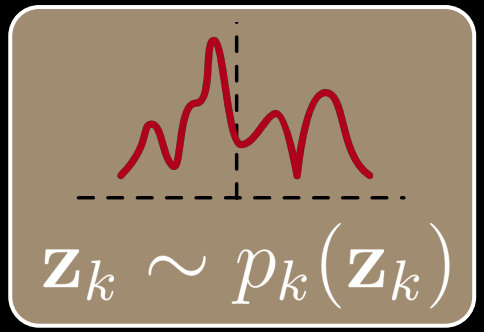
- Too simplistic assumption in most cases
- Real data follows largely complex distributions
- These distributions are usually intractable



The almighty Gaussian

How to model complex distributions but keep analytical simplicity ?

Real data ?



Entering the world of **approximate inference**

Approximating statistics

1. Sampling methods

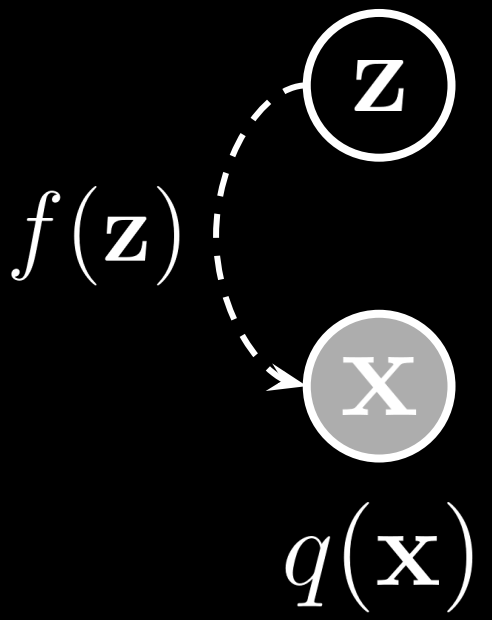
Optimizing simpler distributions

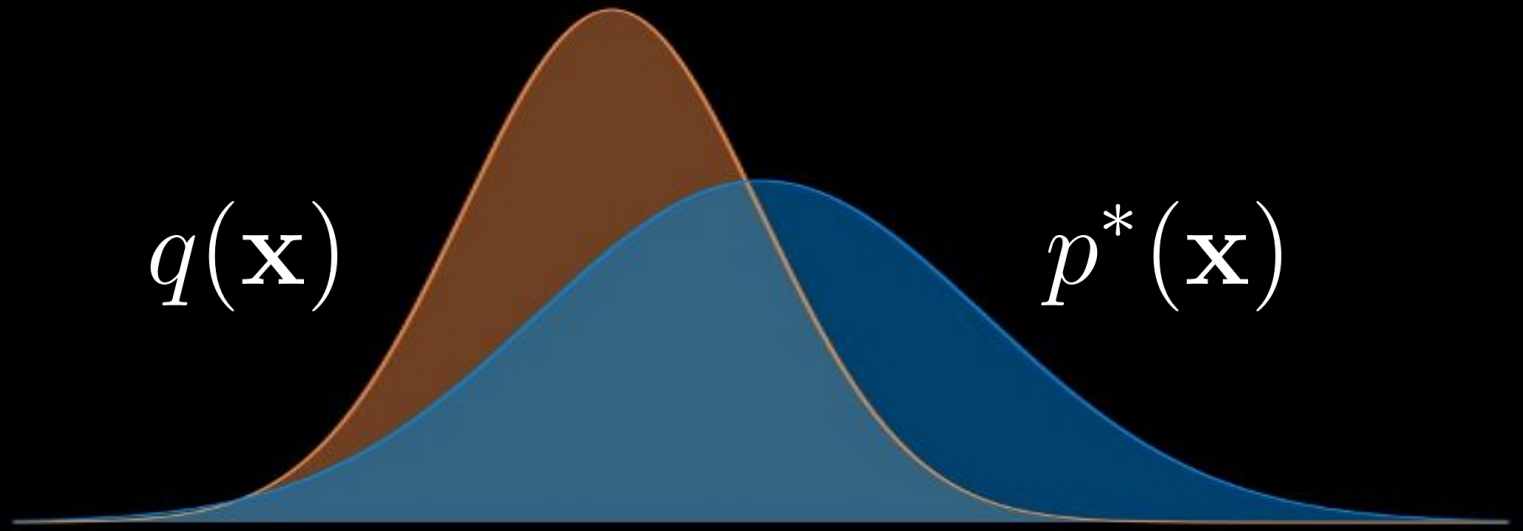
1. Variational inference
2. Normalizing flows
3. **Adversarial learning**

Learning by comparison

For our models, we only have access to $\mathbf{x}$

- **Approximate** probability observation $q(\mathbf{x})$
- Partial knowledge of **real** distribution $p^*(\mathbf{x})$
- Define a **simulator** of data $f(\mathbf{z})$

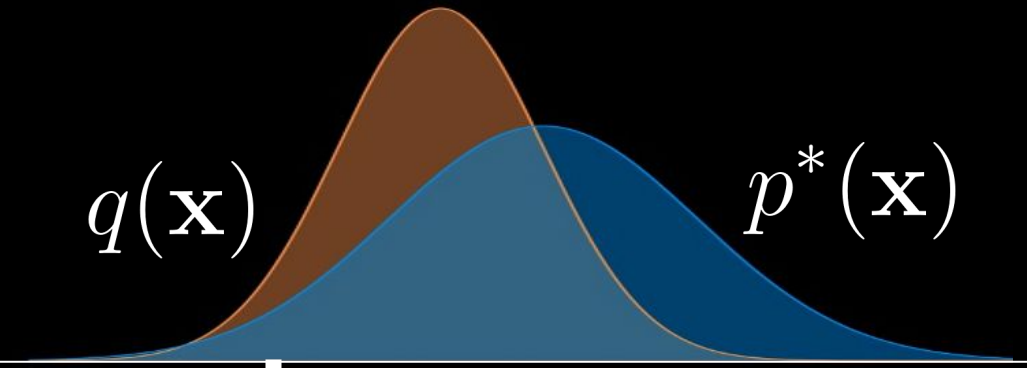




How to compare the estimated distribution $p^*(\mathbf{x})$ to the true distribution $q(\mathbf{x})$ using only ***samples***

Learning by comparison

Density estimation by comparison



$\mathcal{L}(\theta, \phi)$

Probability difference $r_\phi = p^* - q_\theta$

Maximum Mean Discrepancy

Moment matching

Probability ratio $r_\phi = \frac{p^*}{q_\theta}$

Bregman Divergence

Class probability estimation

f-Divergence

*Use a hypothesis test or comparison to **build an auxiliary model** to indicate how data simulated from the model differs from observed data*

Adjust parameters to match the data distribution by comparison

Density ratios and classification

Objective **Density ratio** $\frac{p^*(\mathbf{x})}{q_\theta(\mathbf{x})}$

Bayes' rule **(reminder)** $p(\mathbf{x} \mid y) = \frac{p(y|\mathbf{x})p(\mathbf{x})}{p(y)}$

Goal

Based on real data samples $\left\{\mathbf{x}_1^{(r)}, \cdots, \mathbf{x}_m^{(r)}\right\}$

Generate data samples $\left\{\mathbf{x}_1^{(g)}, \cdots, \mathbf{x}_n^{(g)}\right\}$ that should resemble real

1. Combine data $\{\mathbf{x}_1, \cdots, \mathbf{x}_N\} = \left\{\mathbf{x}_1^{(r)}, \cdots, \mathbf{x}_m^{(r)}, \mathbf{x}_1^{(g)}, \cdots, \mathbf{x}_n^{(g)}\right\}$

2. Assign labels $\{y_1, \cdots, y_N\} = \{+1, \cdots, +1, -1, \cdots, -1\}$

Real data *Generated data*

Equivalence $p^*(\mathbf{x}) = p(\mathbf{x}|y=+1) \quad q(\mathbf{x}) = p(\mathbf{x}|y=-1)$

Density ratios and classification

Our equivalence

$$p^*(\mathbf{x}) = p(\mathbf{x}|y=+1) \quad \textbf{real distribution}$$
$$q(\mathbf{x}) = p(\mathbf{x}|y=-1) \quad \textbf{generative distribution}$$

Developing the density ratio

$$\frac{p^*(\mathbf{x})}{q(\mathbf{x})} = \frac{p(\mathbf{x}|y=+1)}{p(\mathbf{x}|y=-1)}$$

$$p(\mathbf{x}|y=+1) = \frac{p(y=+1|\mathbf{x})p(\mathbf{x})}{p(y)}$$

$$p(\mathbf{x}|y=-1) = \frac{p(y=-1|\mathbf{x})p(\mathbf{x})}{p(y)}$$

Final simple equivalence

$$\frac{p^*(\mathbf{x})}{q(\mathbf{x})} = \frac{p(y=+1|\mathbf{x})}{p(y=-1|\mathbf{x})}$$

Our density estimation task **reduces to a *classification task***

Unsupervised-as-supervised learning

With a 2-class problem, we can introduce a simple *scoring function*

$$p(y=+1|\mathbf{x}) = D_\theta(\mathbf{x})$$
$$p(y=-1|\mathbf{x}) = 1 - D_\theta(\mathbf{x})$$

We can define this as a **Bernoulli loss**

$$\mathcal{F}(\mathbf{x},\theta,\phi) = \mathbb{E}_{p^*(\mathbf{x})}\left[\log D_\theta(\mathbf{x})\right] + \mathbb{E}_{q_\phi(\mathbf{x})}\left[\log 1 - D_\theta(\mathbf{x})\right]$$

Requires an alternate optimisation

$$\min_\phi \max_\theta \mathcal{F}(\mathbf{x},\theta,\phi)$$

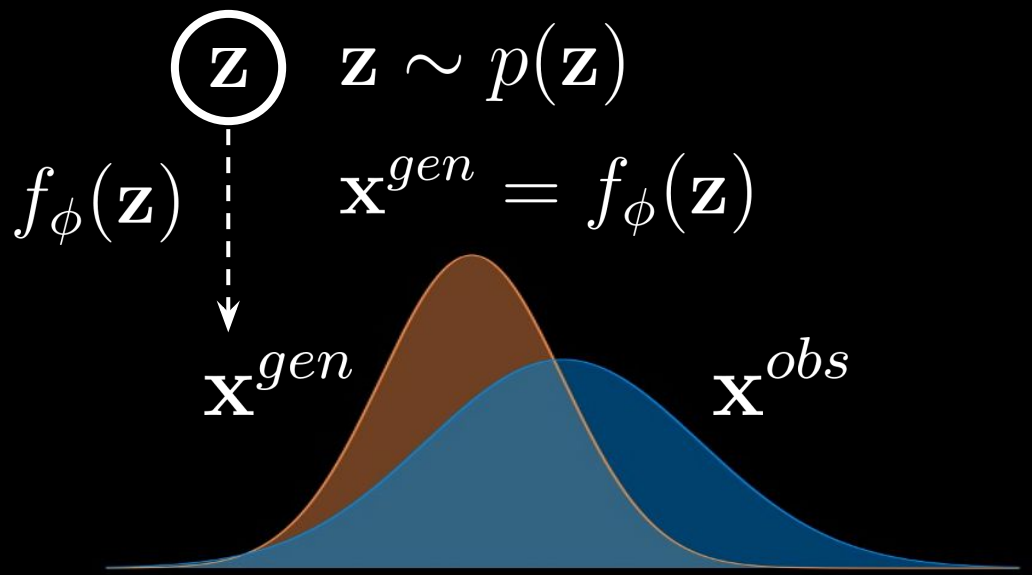
Minimize the number of examples classified as fake
(improve our generation)

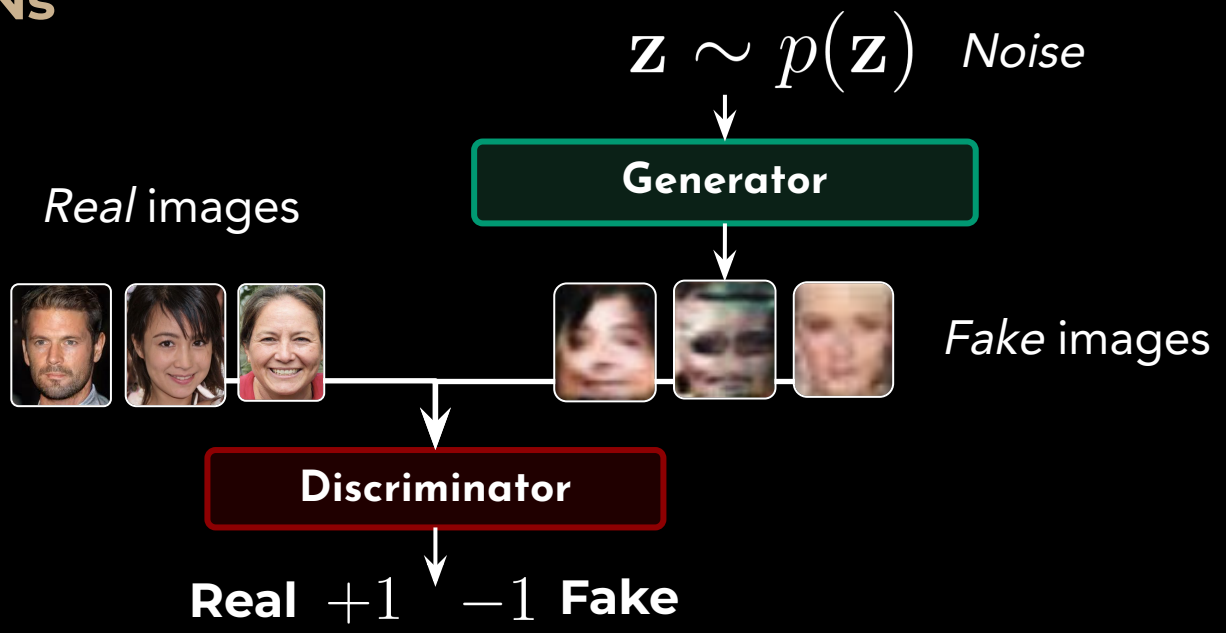
Maximize the quality of our discrimination
(improve our classification)

Paradoxical ?

Generative Adversarial Networks (GAN)

Understanding the structure of GANs





Alternating optimization objective

$$\min_{\phi} \max_{\theta} \mathcal{F}(\mathbf{x}, \theta, \phi) = \mathbb{E}_{p^*(\mathbf{x})} \left[\log D_\theta(\mathbf{x})\right] + \mathbb{E}_{q_\phi(\mathbf{x})} \left[\log 1 - D_\theta(\mathbf{x})\right]$$

Discriminator loss

$$\theta \propto \nabla_\theta \mathbb{E}_{p^*(\mathbf{x})} \left[\log D_\theta(\mathbf{x})\right] + \mathbb{E}_{q_\phi(\mathbf{x})} \left[\log 1 - D_\theta(\mathbf{x})\right]$$

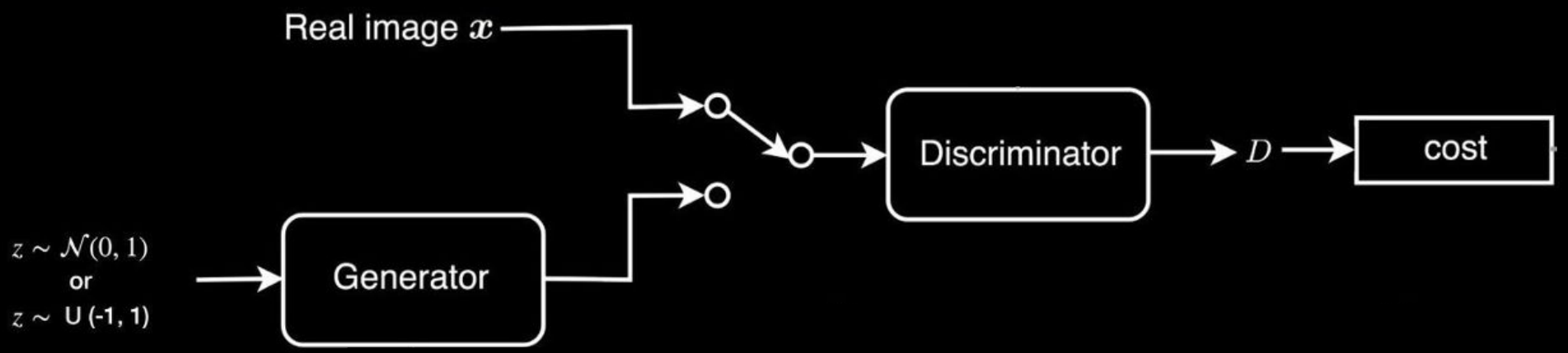
Generator loss

$$\phi \propto -\nabla_\phi \mathbb{E}_{q_\phi(\mathbf{x})} \left[\log D_\theta(f_\phi(\mathbf{z}))\right]$$

Understanding gradient flow

Can be linked to *game theory* (minmax game)

- Generator tries to create outputs that ressemble real data
- Discriminator tries to recognize them
- Defines a **zero-sum game**



$$\mathcal{F}(\mathbf{x}, \theta, \phi) = \mathbb{E}_{p^*(\mathbf{x})} \left[\log D_\theta(\mathbf{x})\right] + \mathbb{E}_{q_\phi(\mathbf{x})} \left[\log 1 - D_\theta(\mathbf{x})\right]$$

Understanding the objective

The discriminator D tries to maximize the discrimination

Changing his weights to better separate fake from real

$$\min_{\phi} \boxed{\max_{\theta}} \mathcal{F}(\mathbf{x}, \theta, \phi) = \underbrace{\mathbb{E}_{p^*(\mathbf{x})}}_{\textbf{real} \text{ data}} [\log \boxed{D_\theta(\mathbf{x})}] + \underbrace{\mathbb{E}_{q_\phi(\mathbf{x})}}_{\textbf{fake} \text{ data}} [\log 1 - \boxed{D_\theta(\mathbf{x})}]$$

⇧ 1 ⇩ 0

The generator G tries to minimize the discrimination

Generating better examples to confound the classification

$$\boxed{\min_{\phi}} \max_{\theta} \mathcal{F}(\mathbf{x}, \theta, \phi) = \underbrace{\mathbb{E}_{p^*(\mathbf{x})}}_{\textbf{real} \text{ data}} [\log \boxed{D_\theta(\mathbf{x})}] + \underbrace{\mathbb{E}_{q_\phi(\mathbf{x})}}_{\textbf{fake} \text{ data}} [\log 1 - \boxed{D_\theta(\mathbf{x})}]$$

⇩ 0 ⇧ 1

Alternating and competing objectives

Issues

- **Differences in strength** of discriminator and generator
- Leads to **unstable training dynamic**

Understanding the objective

Objective function to optimize

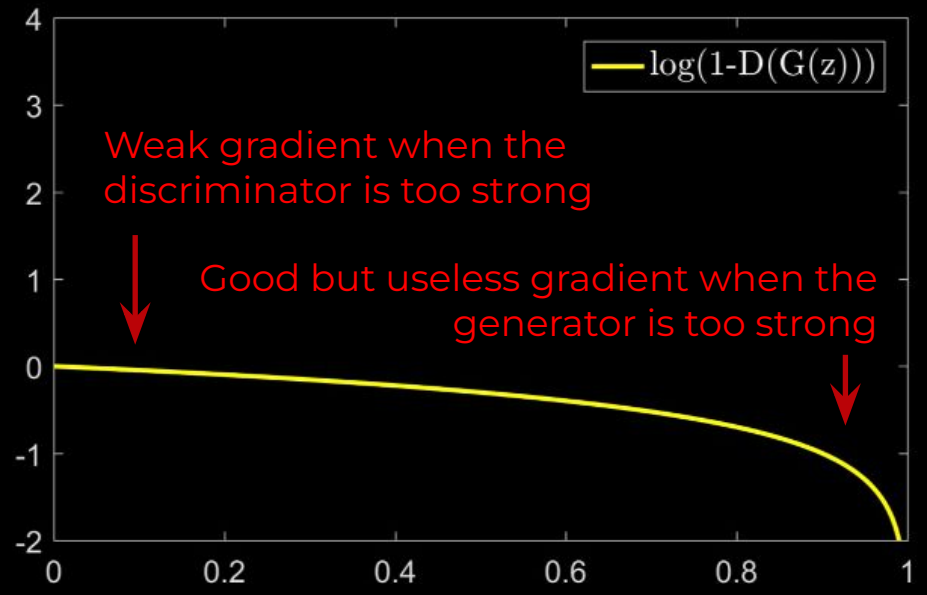
$$\min_{\phi} \max_{\theta} \mathcal{F}(\mathbf{x}, \theta, \phi) = \mathbb{E}_{p^*(\mathbf{x})} \left[\log D_\theta(\mathbf{x})\right] + \mathbb{E}_{q_\phi(\mathbf{x})} \left[\log 1 - D_\theta(G_\phi(\mathbf{z}))\right]$$

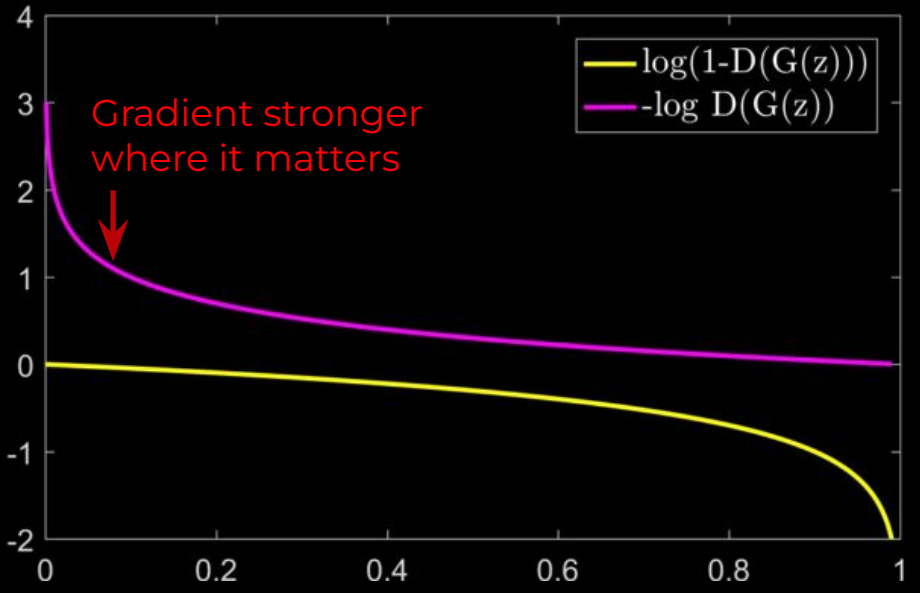
– Gradient **ascent** for the discriminator

$$\max_{\theta} \mathbb{E}_{p^*(\mathbf{x})} \left[\log D_\theta(\mathbf{x})\right] + \mathbb{E}_{q_\phi(\mathbf{x})} \left[\log 1 - D_\theta(G_\phi(\mathbf{z}))\right]$$

– Gradient **descent** for the generator

$$\min_{\phi} \mathbb{E}_{q_\phi(\mathbf{x})} \left[\log 1 - D_\theta(G_\phi(\mathbf{z}))\right] \Rightarrow \max_{\phi} \mathbb{E}_{q_\phi(\mathbf{x})} \left[D_\theta(G_\phi(\mathbf{z}))\right]$$

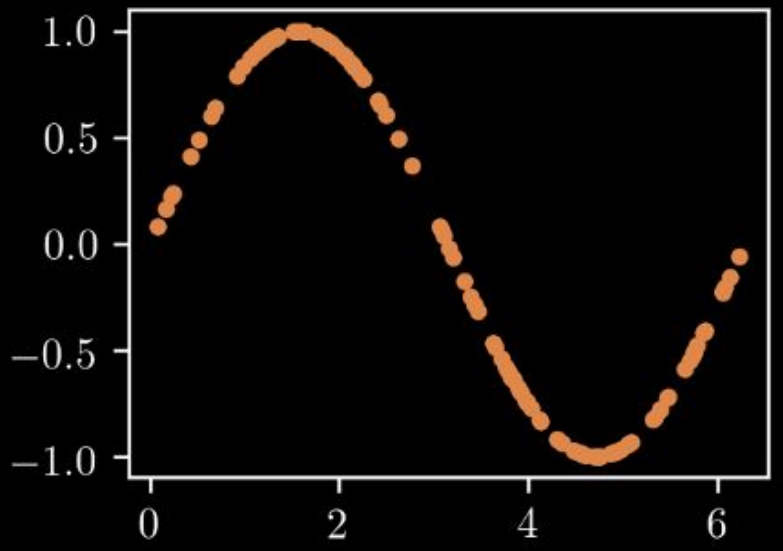


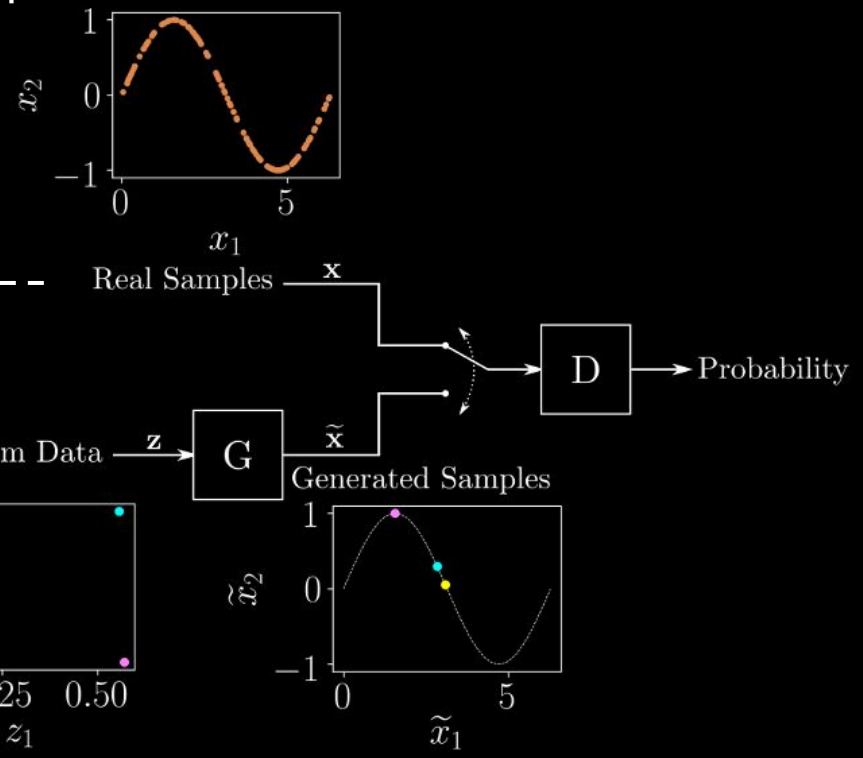


Adversarial toy exercise

Notebook toy exercise

- We will work out toy example (cf. notebook)
- Imagine we want to optimize a *sinus* distribution



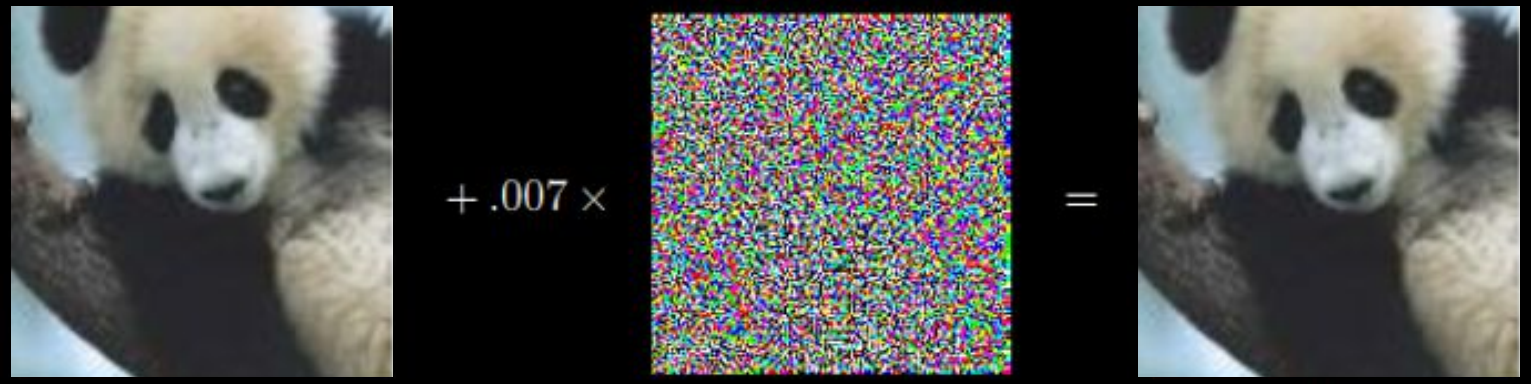


- The generator draws random samples
- *Tries to optimize the function to resemble a sinus*

Adversarial attacks

Vulnerability of machine learning models to adversarial examples

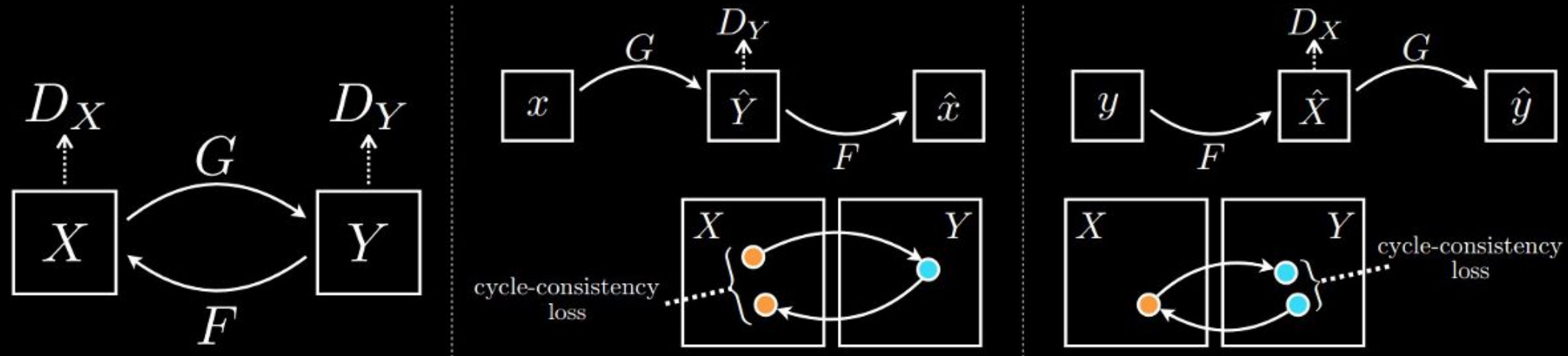
- Adding a small amount of structured (learned) noise
- Leads to very large error rates (with large confidence on errors)
- Can be applied to any type of model

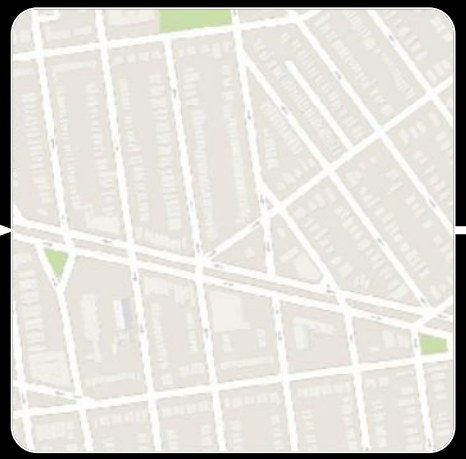


- We can directly add adversarial examples in training
  - Adversarial objective function with the *fast gradient sign method*

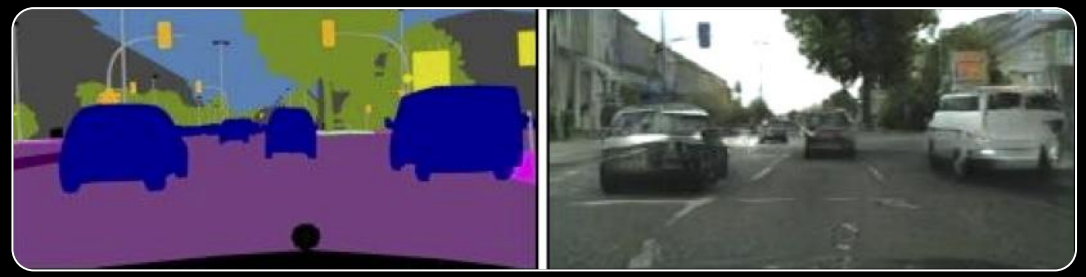
CycleGAN

Key idea of CycleGAN is the **cycle loss**

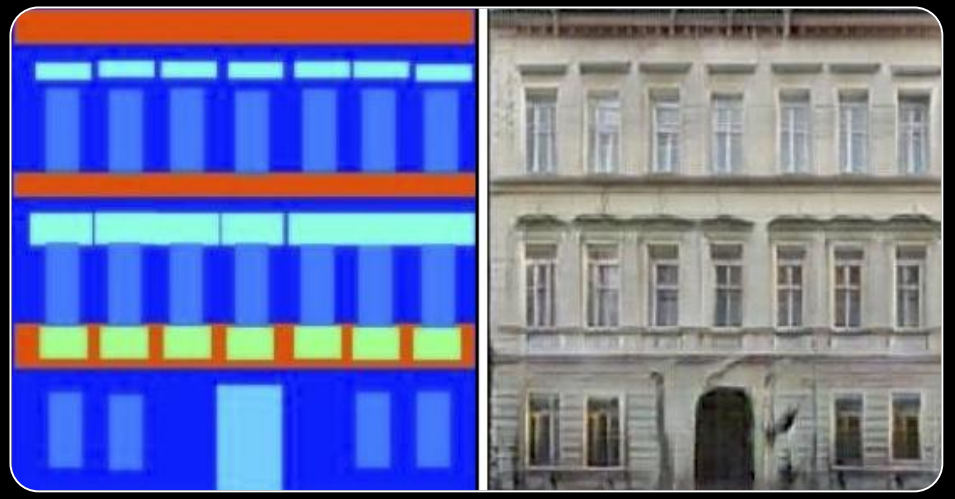


$\mathbf{x}$  $G(\mathbf{x})$  $F(G(\mathbf{x}))$ 

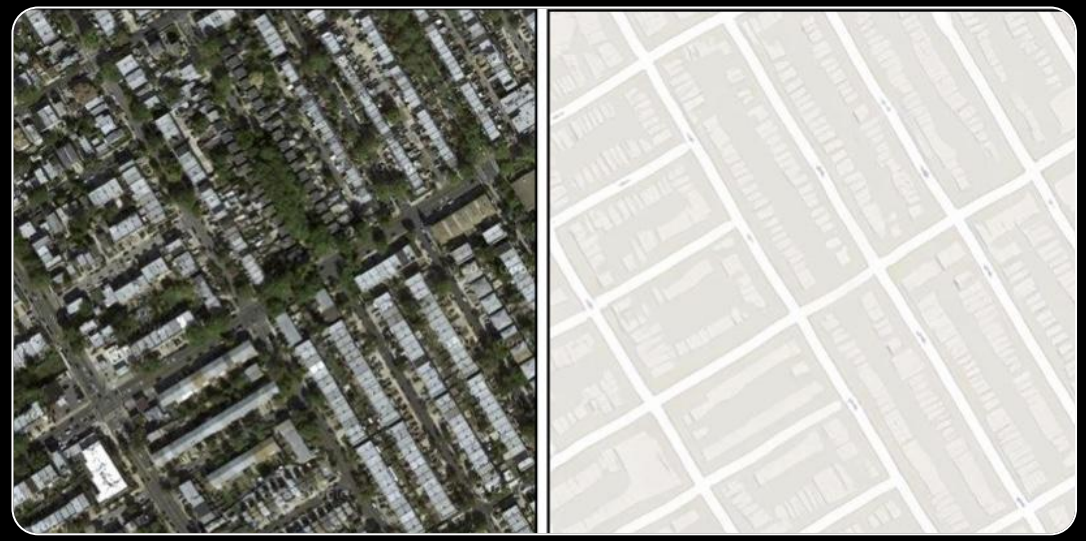
Pix2Pix - Transfer tasks



contour → photography



contour → building



photography → maps

black & white → colors

GANs in action - Style transfer

painting → photography

zebra → horses

summer → winter

photography → painting

zebra → horses

winter → summer

input picture

monet *van gogh* *cezanne* *ukiyo-e*

Progressing growing of GANs

Looking for a way to

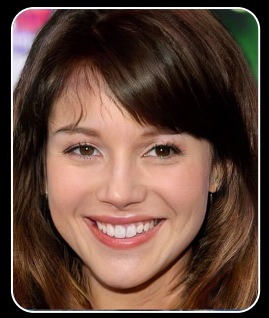
- Stabilize the training of GANs
- Increase the output quality
- Output images of 1024x1024

Key observation

GANs struggle with **finer details**

Key ideas

- Need to gradually increase image size
- Start by highly quantified images
- Use these as conditioning
- Add a heuristic for diversity

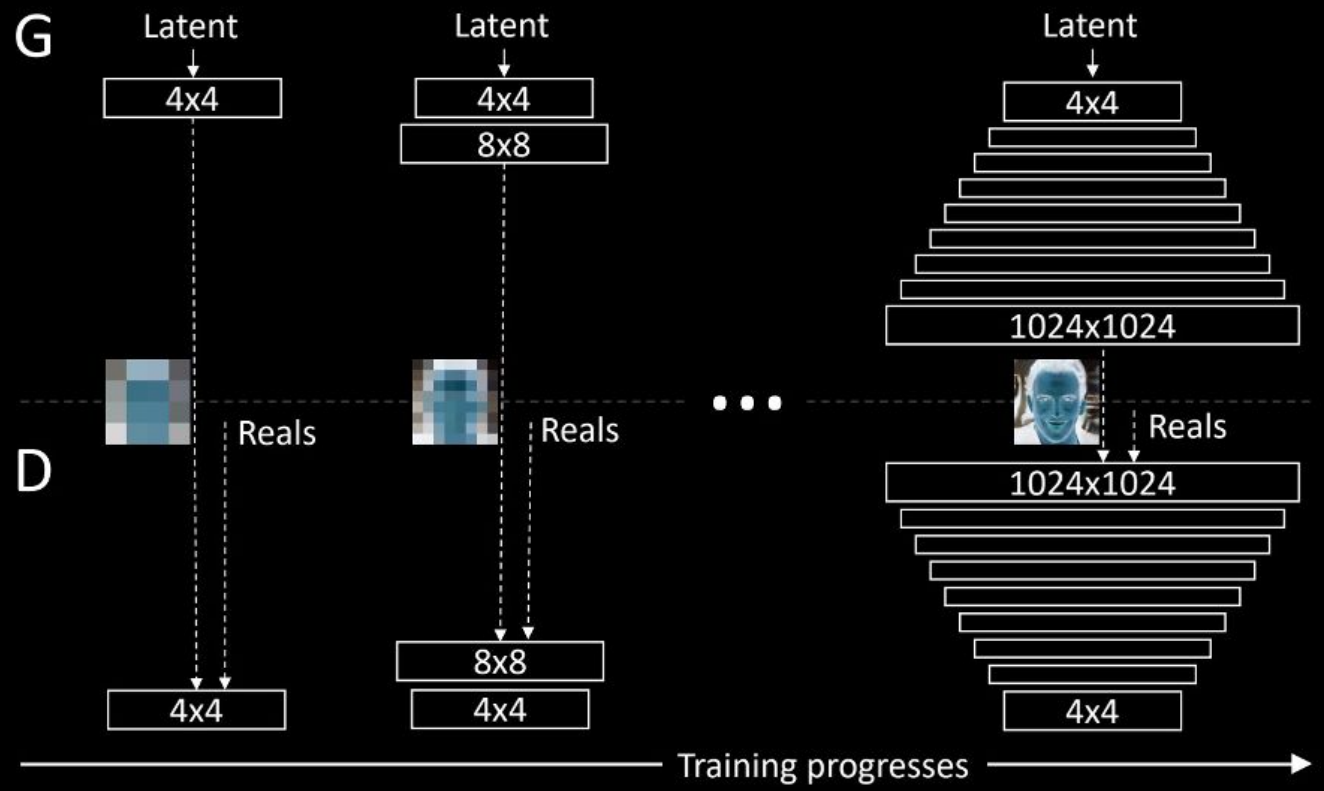



[Karras et al., Progressive Growing of GANs for Improved Quality, Stability, and Variation, ICLR 2018.]

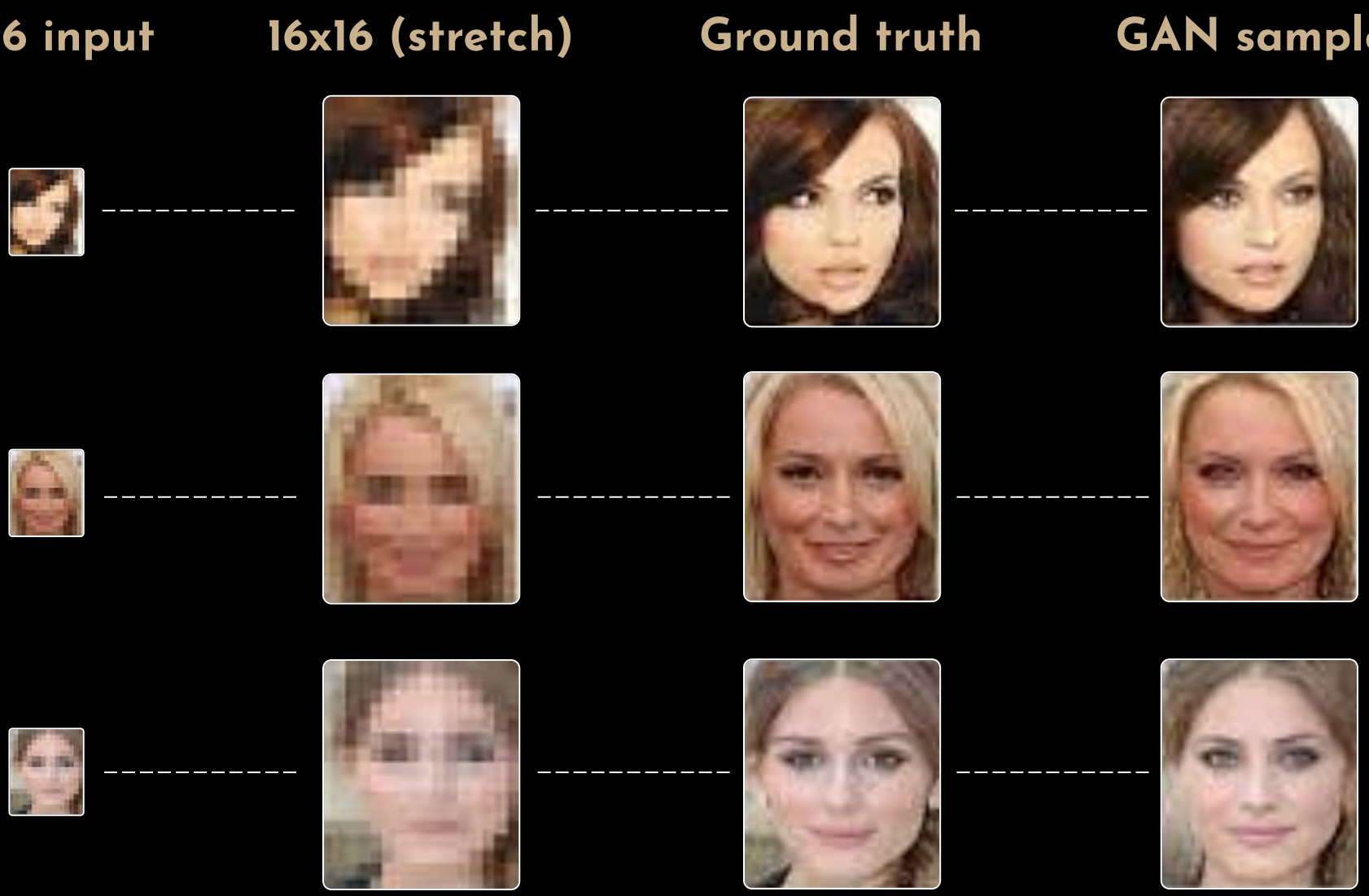
Progressive growing of GANs

Major advances

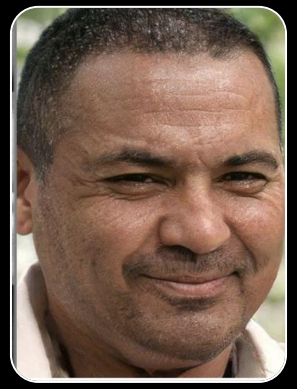
- Progressively increase the size of generation
- And consequently the size of the generator
- Allows the network to discover large-scale structures
- Then refines the image details
- Faster, as it trains on smaller images



Super-resolution with GAN



Evolution in a few years



2015 GAN

2016 CoGAN

2017 ProGAN

2018 StyleGAN

2019 BigGAN

2020 Diffusion