Implicit models – Adversarial Learning

INTELLIGENT SYSTEMS FOR PATTERN RECOGNITION (ISPR)

DAVIDE BACCIU – DIPARTIMENTO DI INFORMATICA - UNIVERSITA’ DI PISA

DAVIDE.BACCIU@UNIPI.IT

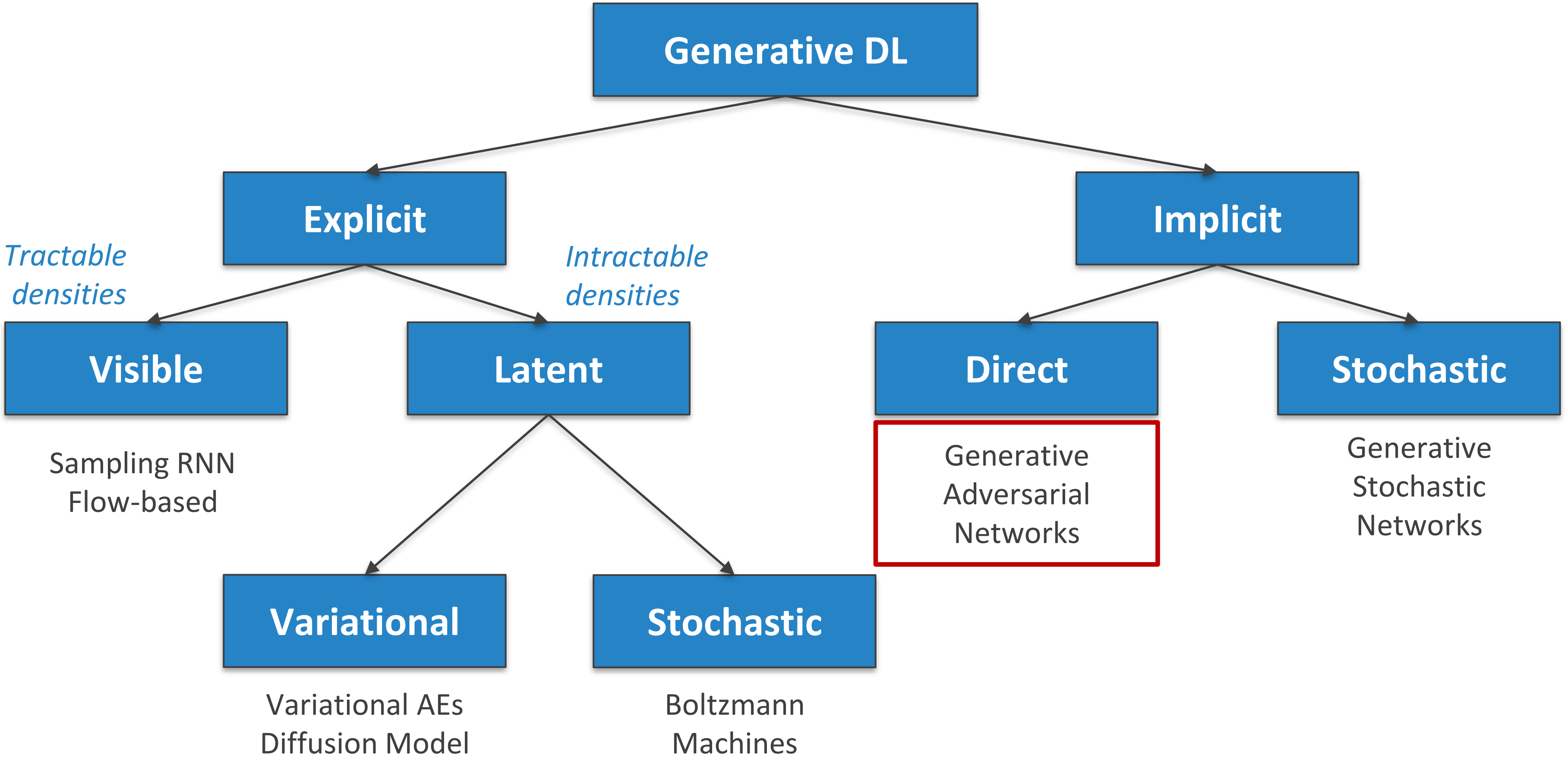
Generative Learning from a DL Perspective

Given training data, learn a (deep) neural network that can generate new samples from (an approximation of) the data distribution

Two approaches

- Explicit $\Longrightarrow$ Learn a model density $P_\theta(x)$
- Implicit $\Longrightarrow$ Learn a process that samples data from $P_\theta(x) \approx P(x)$

Our Taxonomy



Adapted from I. Goodfellow, Tutorial on Generative Adversarial Networks, 2017

Università di Pisa

Distribution Learning Vs Learning to Sample

- Variational AEs learn to approximate an intractable distribution

$$P_\theta(\boldsymbol{x}) = \int P_\theta(\boldsymbol{x}|\boldsymbol{z})P(\boldsymbol{z})d\boldsymbol{z}$$

then sample it to generate the output

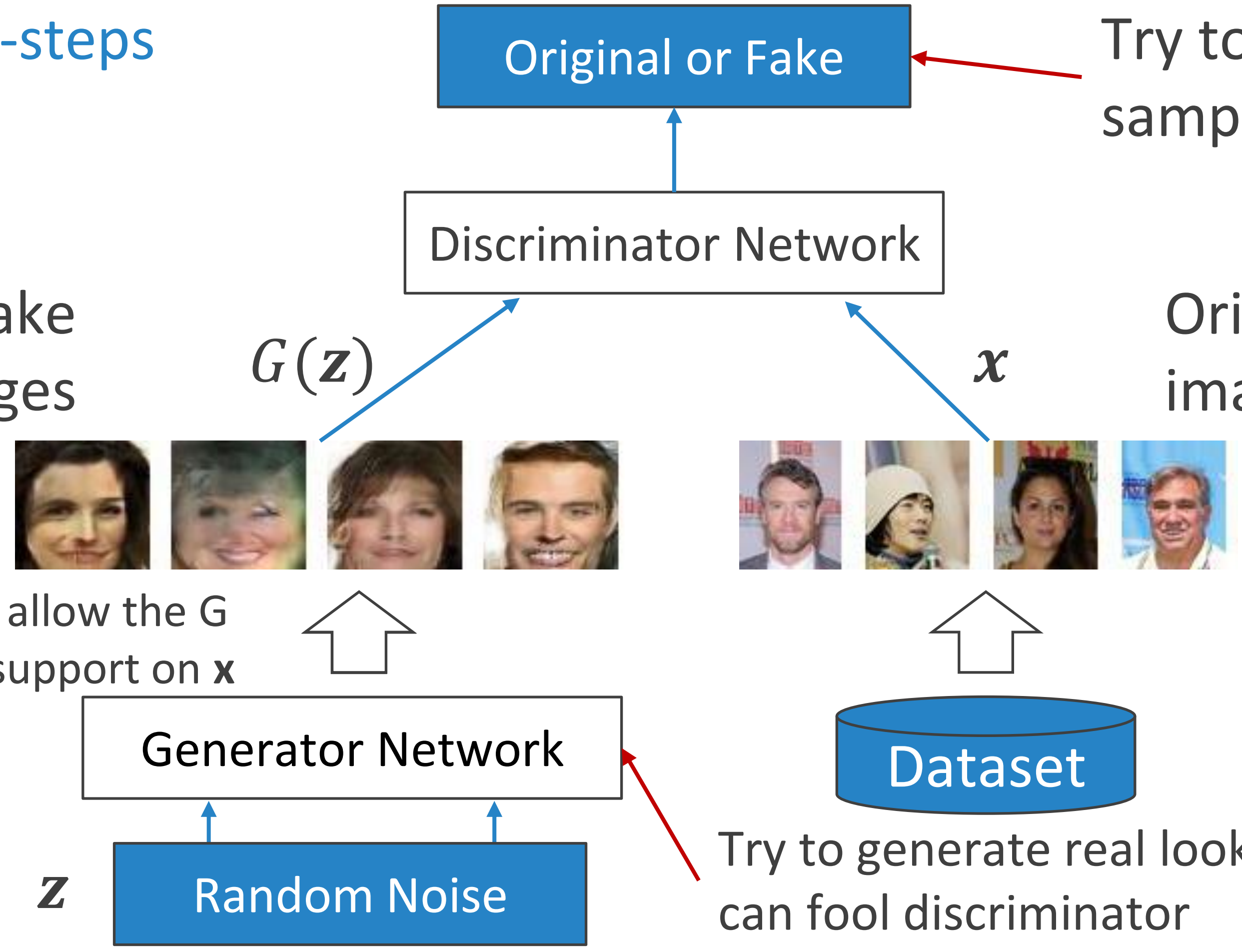
- What if we learn to generate samples rather than learning the distribution?
  - Generative Adversarial Networks (GAN)
  - Game theoretic approach

The GAN Catch

- We need to learn to sample from a complex, high-dimensional training distribution
  - No straightforward way to do this
- The catch
  - Sample from a simple distribution: random noise
  - Train a differentiable deterministic function (neural network) to transform random noise to the training distribution

Generative Adversarial Networks

Again, a two-steps architecture



z size $\geq$ **x** size to allow the G distribution full support on **x**

Alternate Optimization

$$C = \min_{\theta_G} \max_{\theta_D} \left[\mathbb{E}_x \left[\log \underbrace{D_{\theta_D}(x)}_{\text{Discriminator output for real data } x} \right] + \mathbb{E}_z \left[\log(1 - \underbrace{D_{\theta_D}\left(G_{\theta_G}(z)\right)}_{\text{Discriminator output for fake data G(z)}}) \right] \right]$$

- Discriminator output is likelihood of input being real
- Discriminator tries to maximize C s.t.
  - $D_{\theta_D}(x) \rightarrow 1$ and $D_{\theta_D}\left(G_{\theta_G}(z)\right) \rightarrow 0$
- Generator tries to minimize C s.t.
  - $D_{\theta_D}\left(G_{\theta_G}(z)\right) \rightarrow 1$

Alternate Optimization

$$C = \min_{\theta_G} \max_{\theta_D} \left[\mathbb{E}_x [\log D_{\theta_D}(x)] + \mathbb{E}_z [\log(1 - D_{\theta_D}(G_{\theta_G}(z)))] \right]$$

1. Discriminator gradient ascent

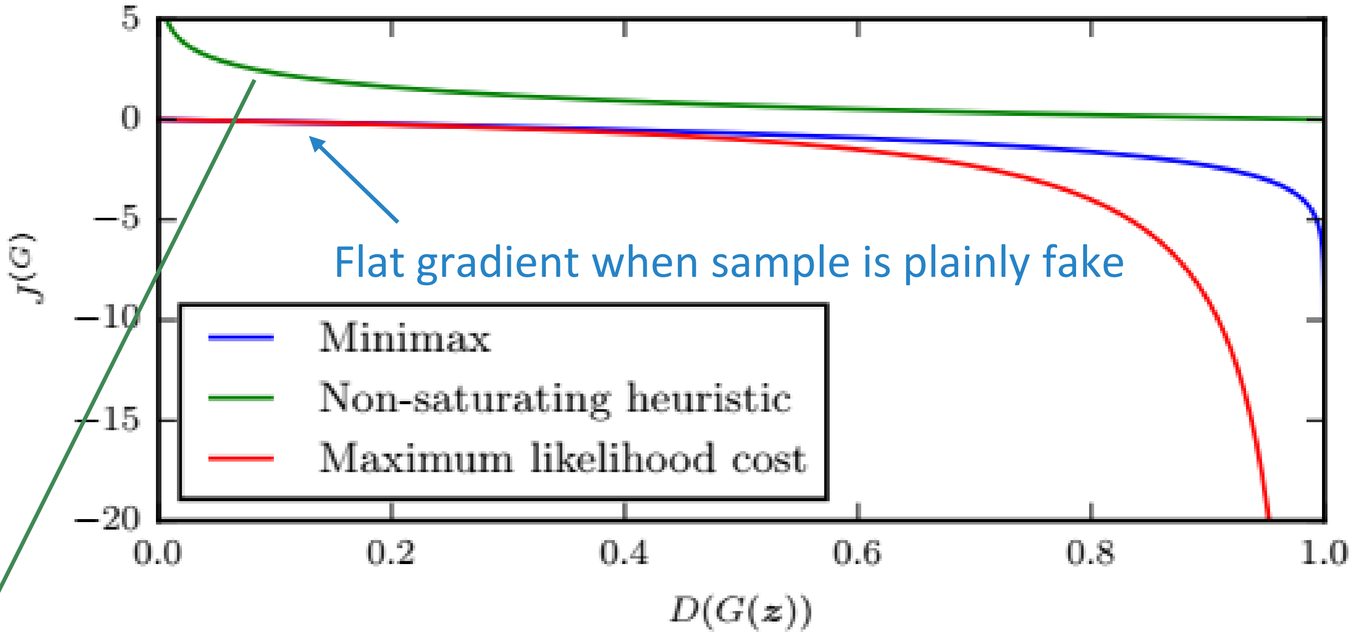
$$C_D = \max_{\theta_D} \left[\mathbb{E}_x [\log D_{\theta_D}(x)] + \mathbb{E}_z [\log(1 - D_{\theta_D}(G_{\theta_G}(z)))] \right]$$

2. Generator gradient descent

$$C_G = \min_{\theta_G} \underbrace{\left[\mathbb{E}_z [\log(1 - D_{\theta_D}(G_{\theta_G}(z)))] \right]}_{\text{Optimizing this doesn't really work}}$$

The Issue and a Solution

The cost that the Generator receives in response to generate $G(\boldsymbol{z})$ depends only on the Discriminator response



$$C_G = \max_{\theta_G} \left[\mathbb{E}_z \left[\log \left(D_{\theta_D} \left(G_{\theta_G}(z) \right) \right) \right] \right]$$ maximize likelihood of discriminator being wrong

GAN Training Pseudo-Algorithm

for number of training iterations **do**

for k steps **do** Stability trick

- 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:

$$\nabla_{\theta_d} \boxed{\frac{1}{m} \sum_{i=1}^{m}} \left[\log D_{\theta_d}(x^{(i)}) + \log(1 - D_{\theta_d}(G_{\theta_g}(z^{(i)}))) \right]$$

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 ascending its stochastic gradient (improved objective):

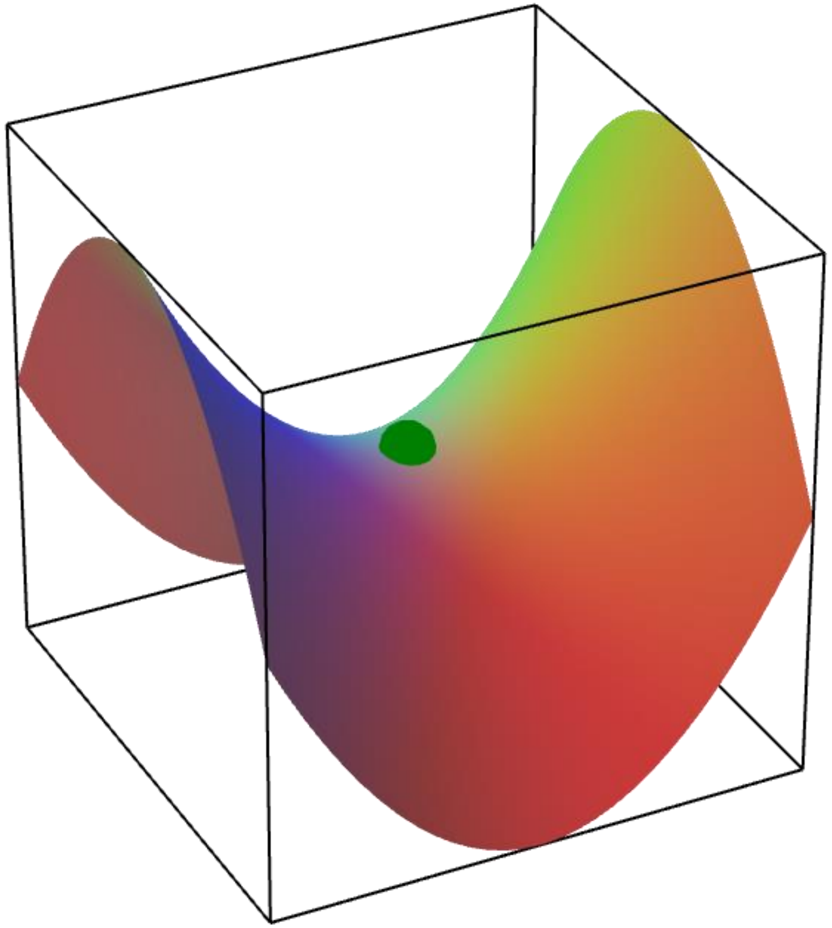
$$\nabla_{\theta_g} \boxed{\frac{1}{m} \sum_{i=1}^{m}} \log(D_{\theta_d}(G_{\theta_g}(z^{(i)})))$$

Expectation

end for

A Hard Two-Player Game

- The optimal solution of the min-max problem is a saddle point
- Little stability
  - Initially lot of heuristic work
  - Now converged to more principled solutions



Wasserstein Distance Models

Attempts to solve the hardness of training adversarial generators by optimizing the Wasserstein distance (EMD) between the generator and empirical distribution filtered trough the discriminator function D

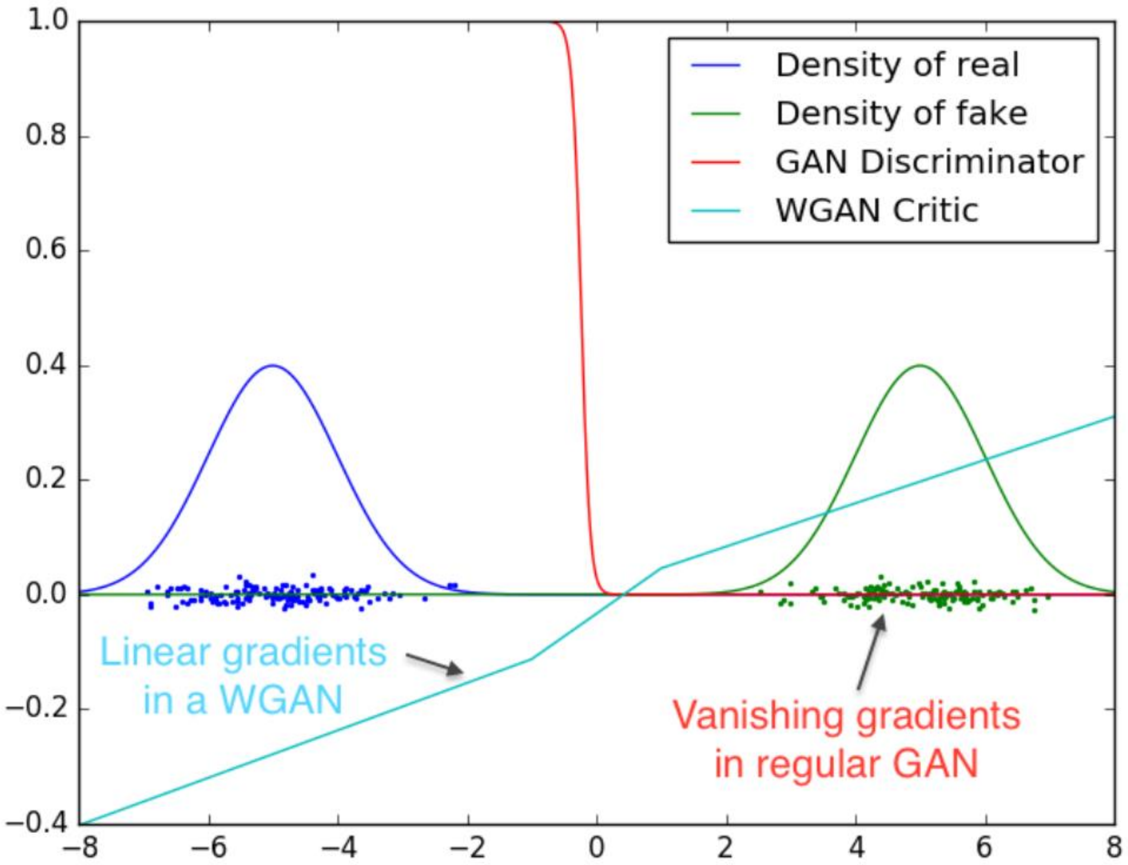
$$G^* = \arg\min_G \boldsymbol{W}(\mu, \mu_G)$$
$$= \arg\min_G \sup_{\|D\|_L \leq 1} \left[\mathbb{E}_{x\sim\mu}[D(x)] - \mathbb{E}_{x\sim\mu_G}[D(x)] \right]$$

$\|D\|_L \leq 1$

Requires optimizing D under a constraint on Lipschitz seminorm

- Clipping D weights (slow to converge)

Effect of the Wasserstein Loss

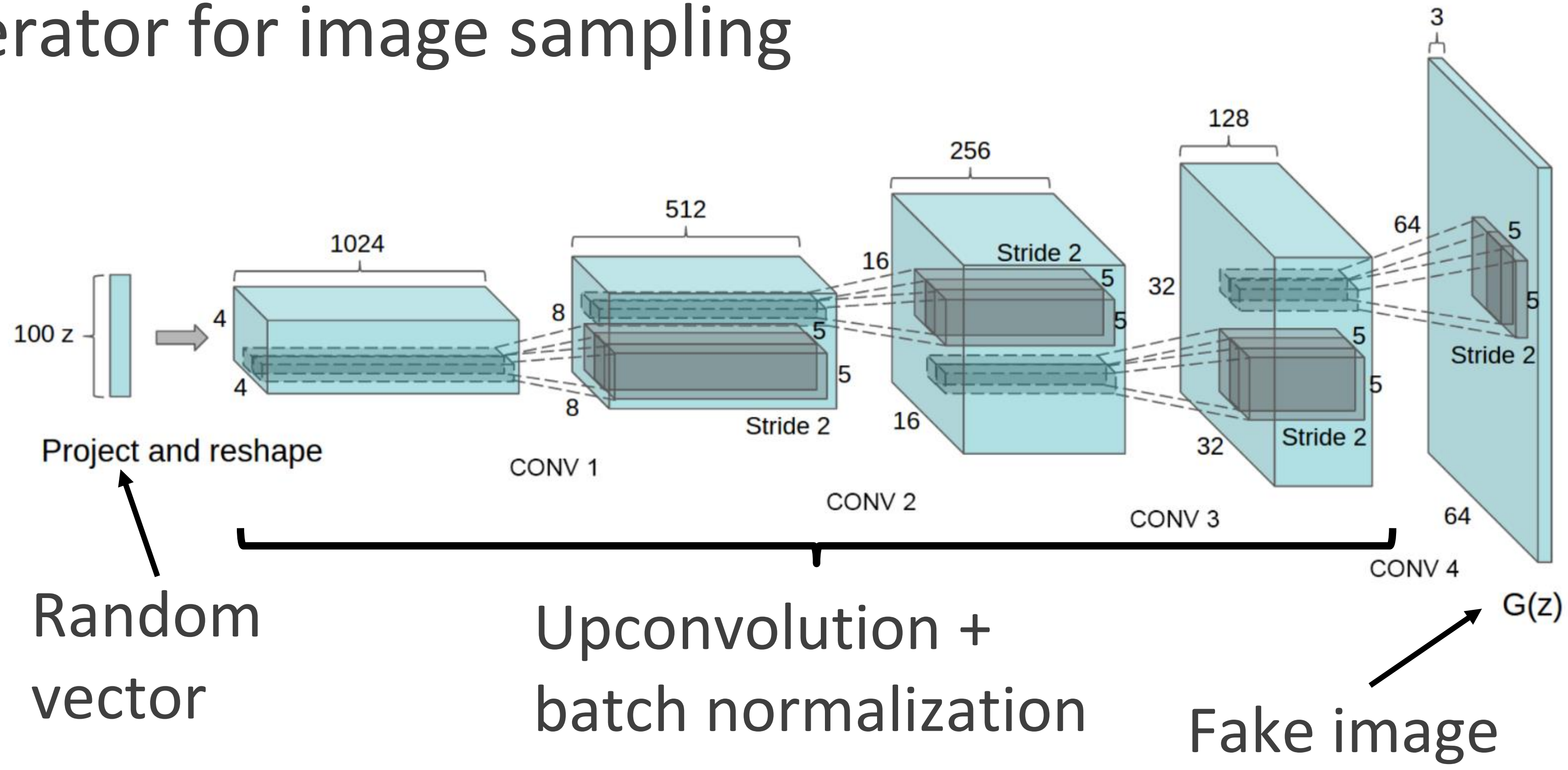


- Classical GAN loss results in saturation (discriminator with zero loss)
- WGAN provide gradients across all the range of training conditions

https://arxiv.org/pdf/1701.07875.pdf

The DCGAN Architecture

Generator for image sampling



Radford et al, “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”, ICLR 2016

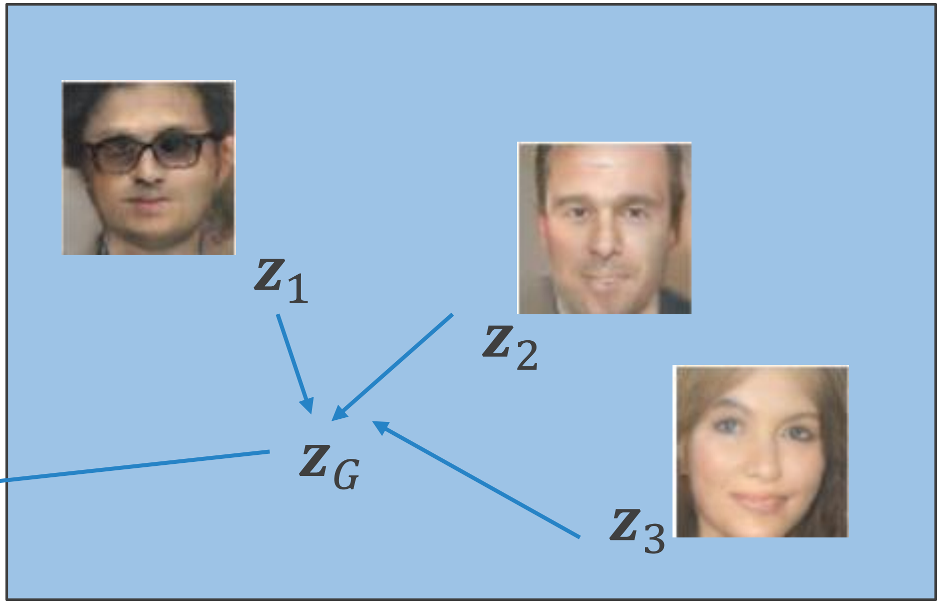
GAN and Images

Radford et al, “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”, ICLR 2016

Latent Space Arithmetic

Can do sensible linear operations on noise vectors (arithmetic, interpolation)

$$\boldsymbol{z}_G = \boldsymbol{z}_1 - \boldsymbol{z}_2 + \boldsymbol{z}_3$$

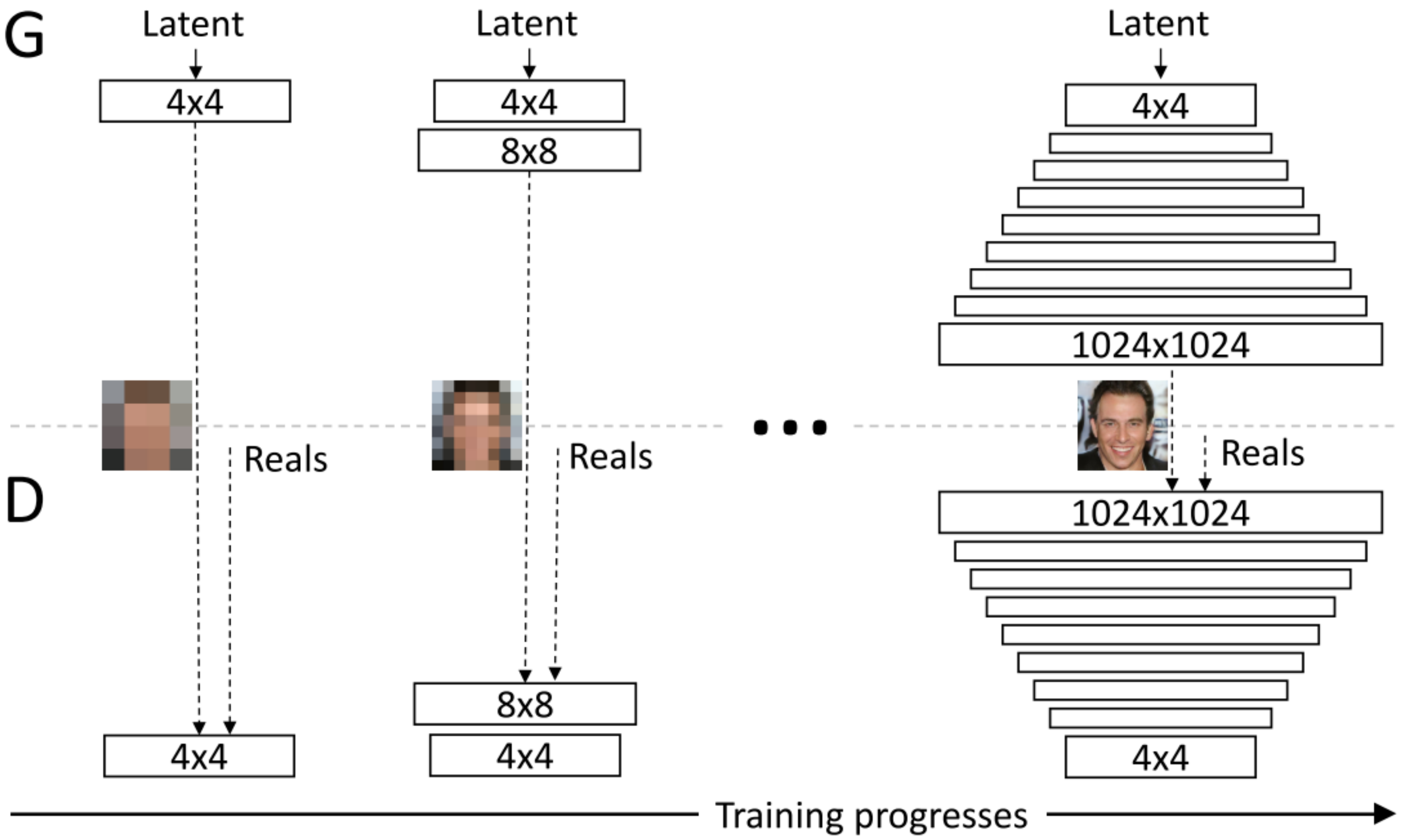


Radford et al, “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”, ICLR 2016

How to Get High Resolution Samples?

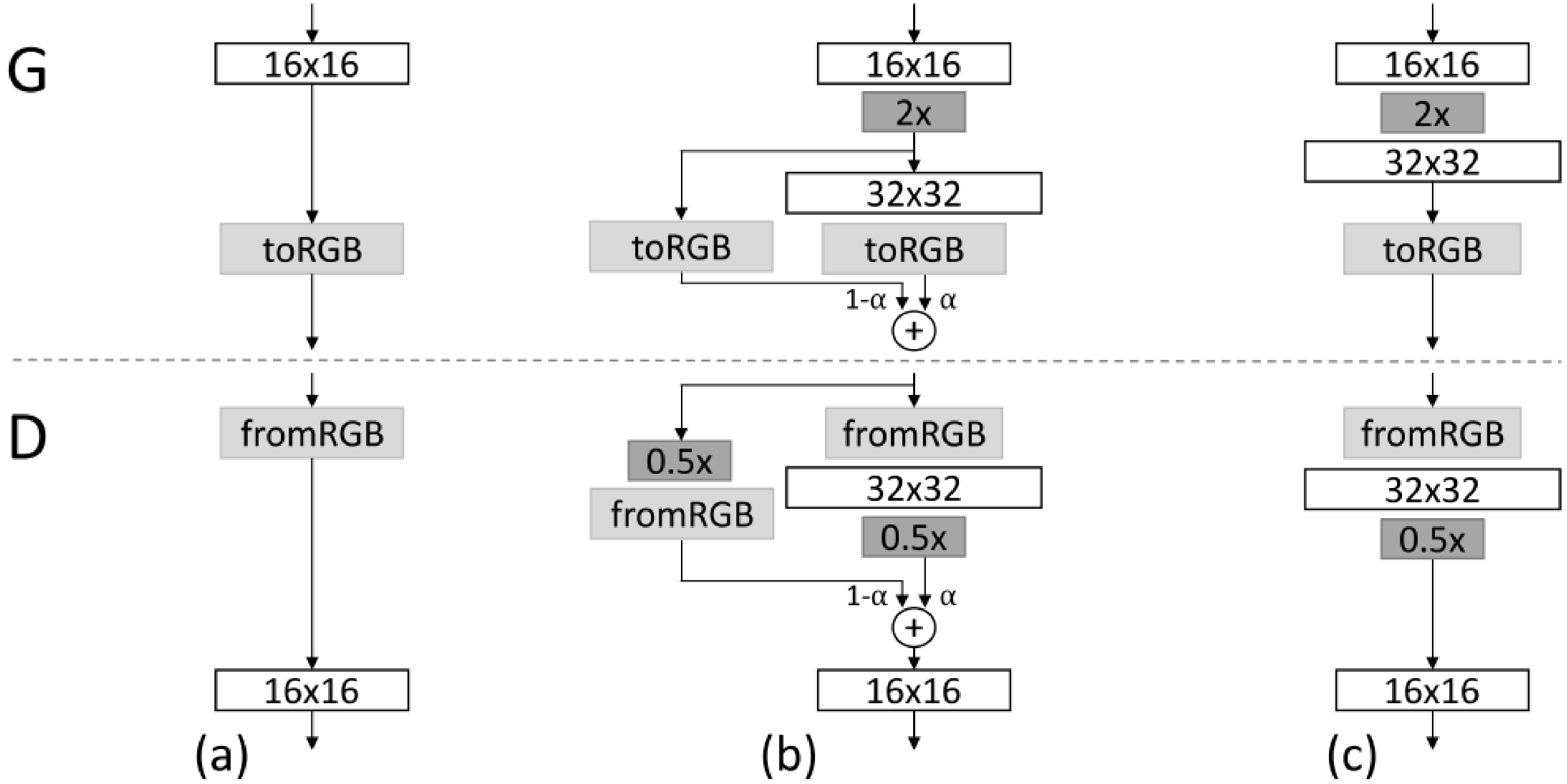
https://arxiv.org/abs/1710.10196

Progressive GAN



https://arxiv.org/abs/1710.10196

Progressive GAN – Smooth Transition

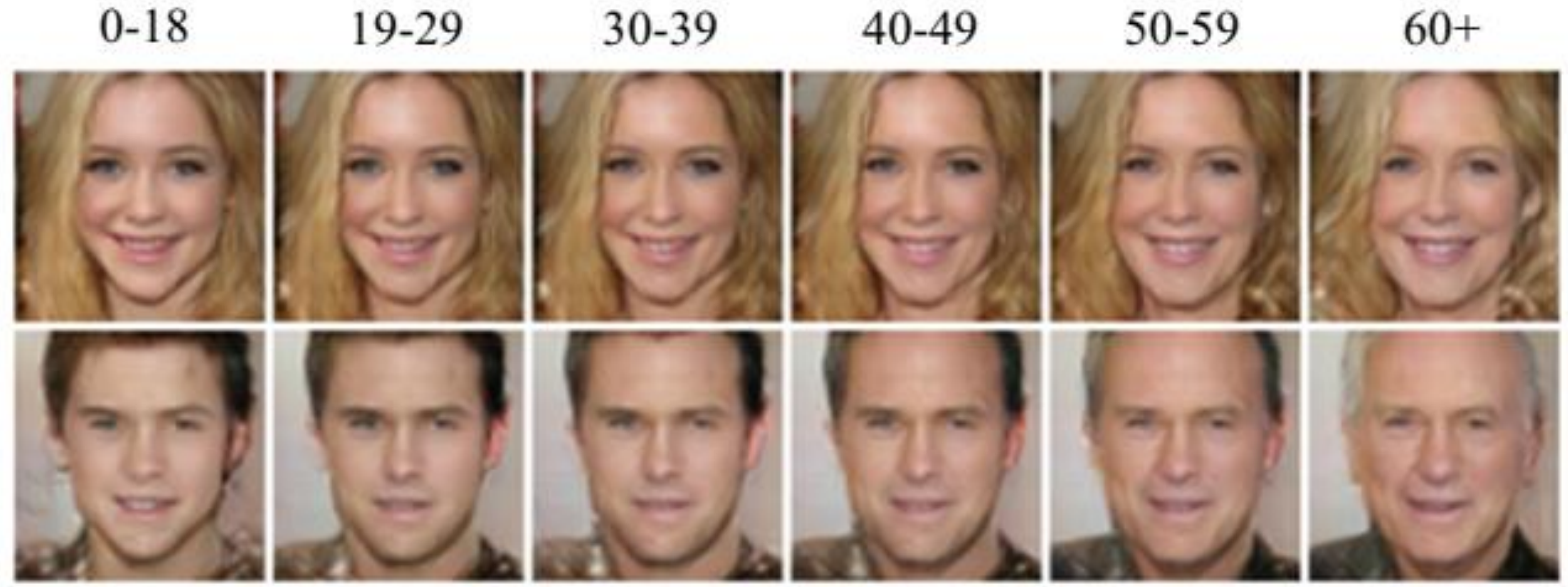


https://arxiv.org/abs/1710.10196

Conditional Generation

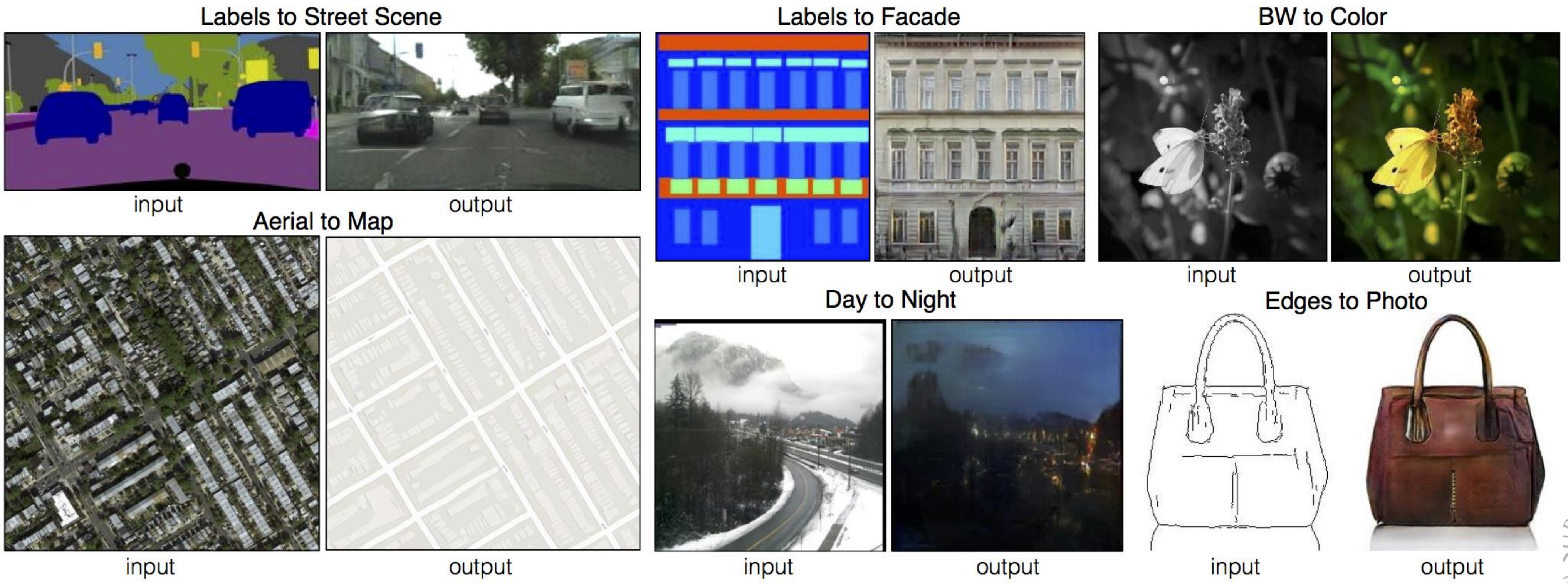
Learn a mapping from an observed side information **x** and a random noise vector **z** to the fooling samples **y**

$$G\colon \{x, z\} \rightarrow y$$



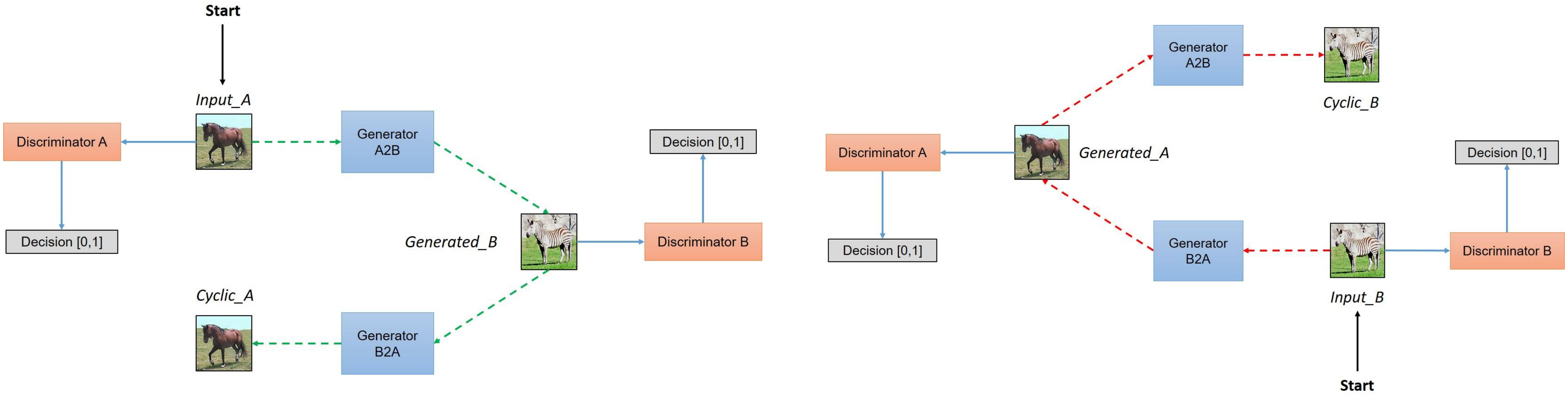
Antipov et al, "Face Aging With Conditional Generative Adversarial Networks", ICIP 2017

Conditional Generation – Image2Image



Isola et al, “Image-to-Image Translation with Conditional Adversarial Networks”, 2016

CycleGAN – Style transfer without pairing

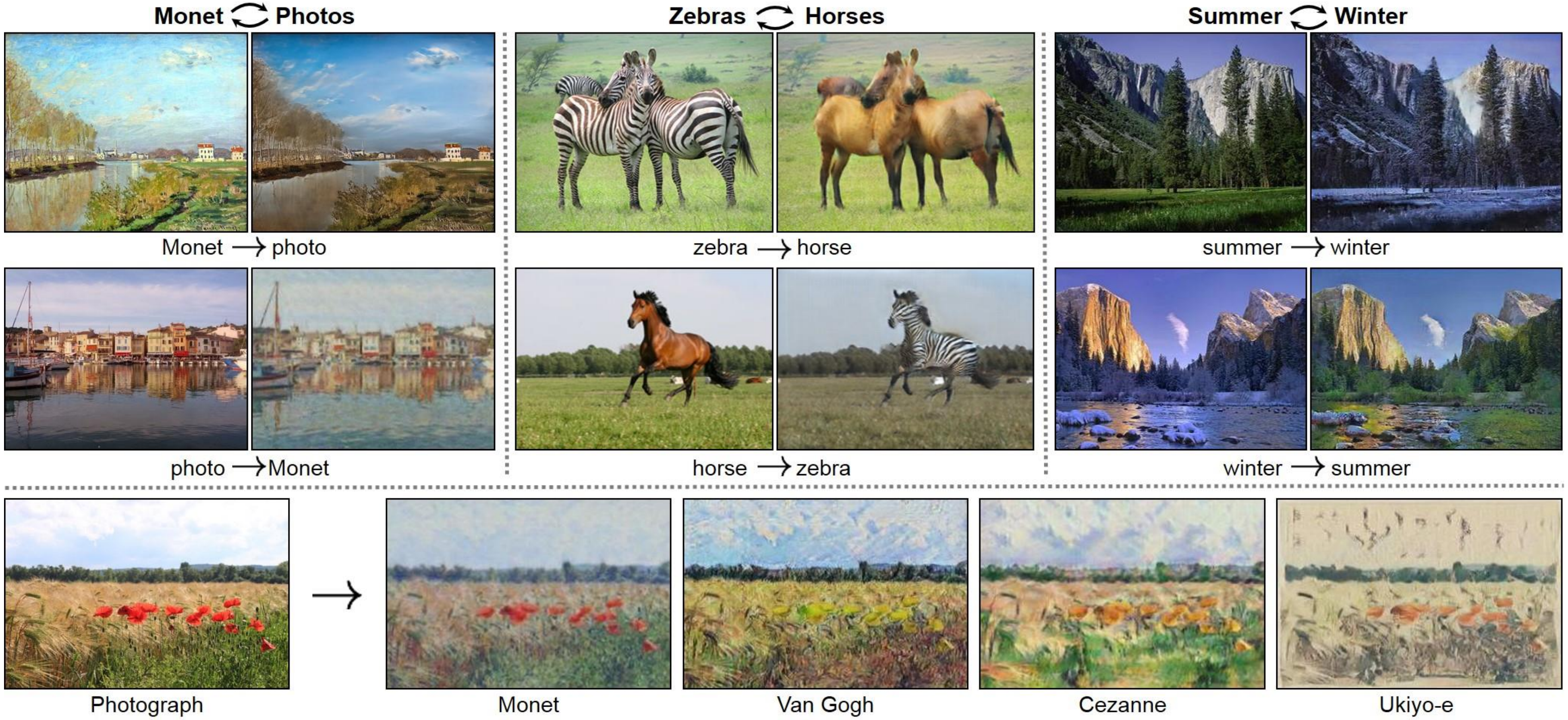


Enforce alignment by ensuring that generated images in domain B can lead to good fakes in domain A and vice versa

https://junyanz.github.io/CycleGAN/

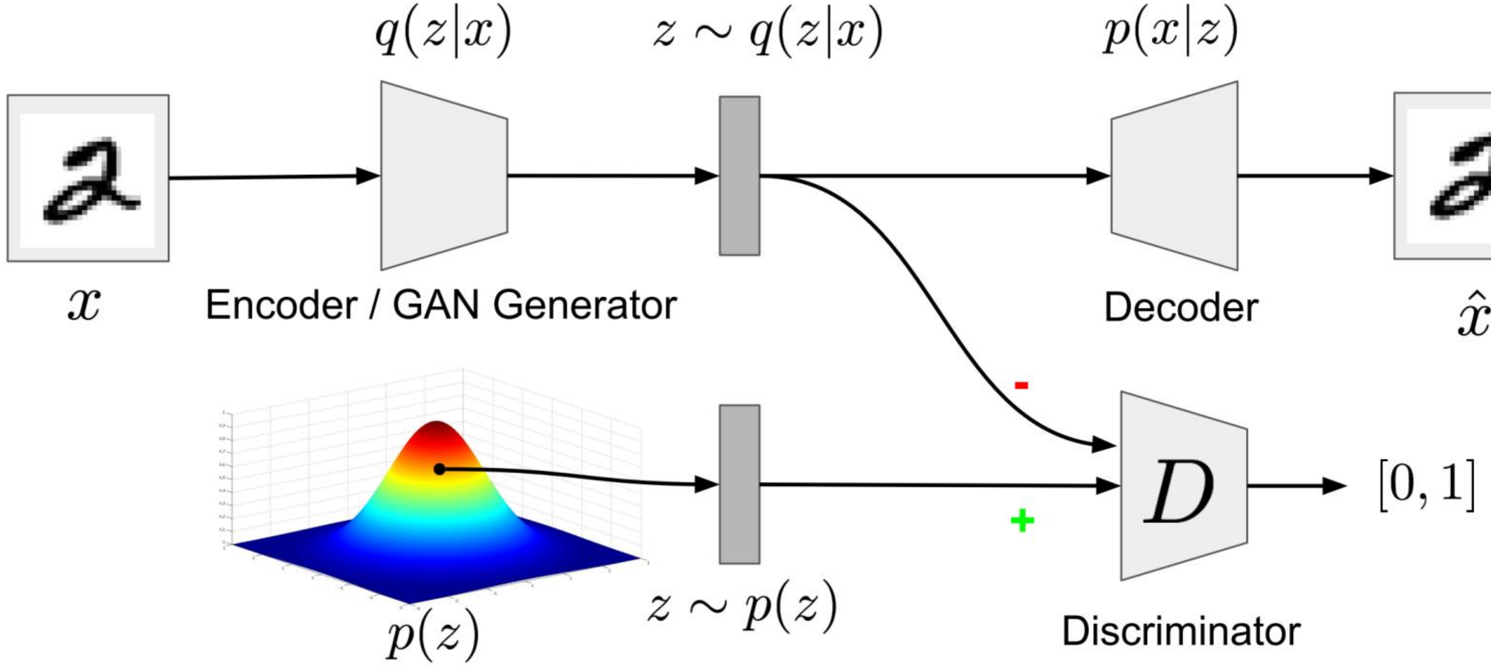
CycleGAN Examples

https://junyanz.github.io/CycleGAN/



Best of 2 worlds?

Adversarial autoencoders (AAE)



Force the latent codes to be indistinguishable from samples of a priori distribution

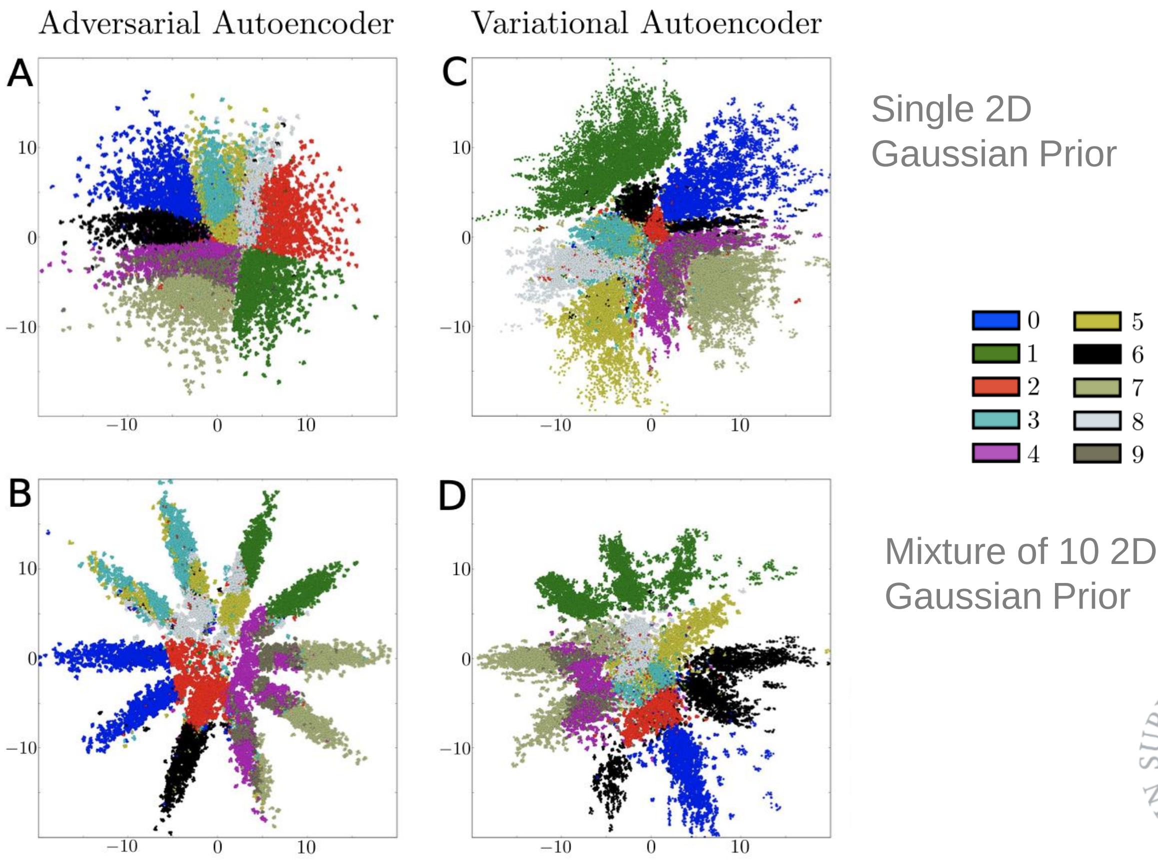
Training AAE

$$\mathcal{L}(x) = \mathbb{E}_Q[\log P(x|z)] - \underbrace{KL(Q(z|x)||P(z))}_{\text{Replaced by an adversarial loss}}$$

- Reconstruction phase - Update the encoder and decoder to minimize reconstruction error
- Regularization phase - Update discriminator to distinguish true prior samples from generated samples; update generator to fool the discriminator
- Adversarial regularization allows to impose priors for which we cannot compute the KL divergence

AAE Vs VAE

AAE yields a smoother coverage of the latent space



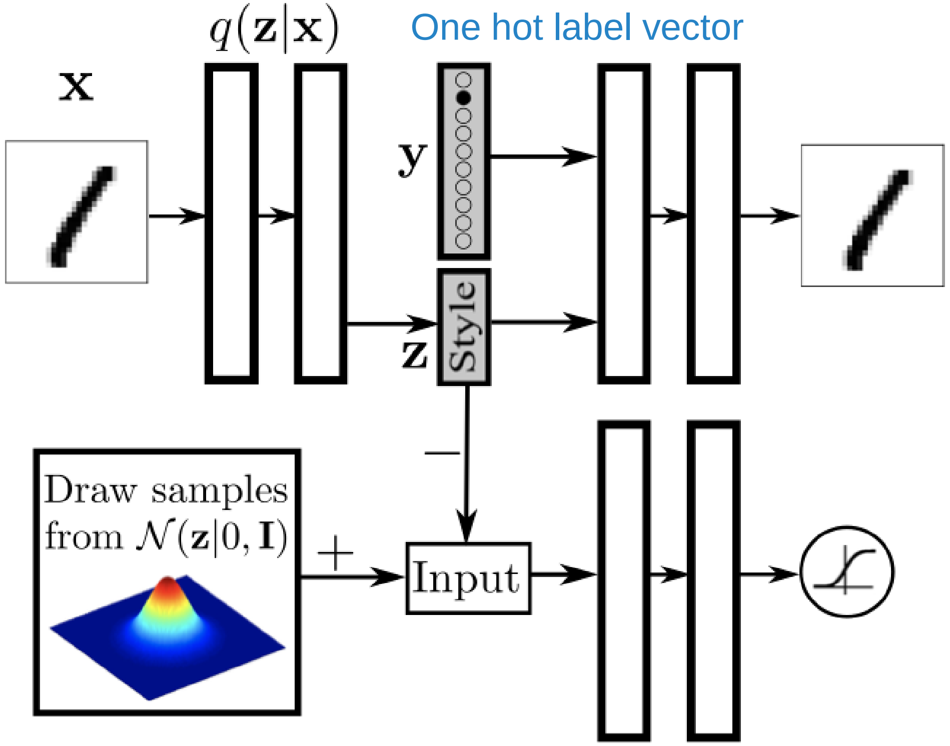
Single 2D Gaussian Prior

Mixture of 10 2D Gaussian Prior

https://arxiv.org/abs/1511.05644

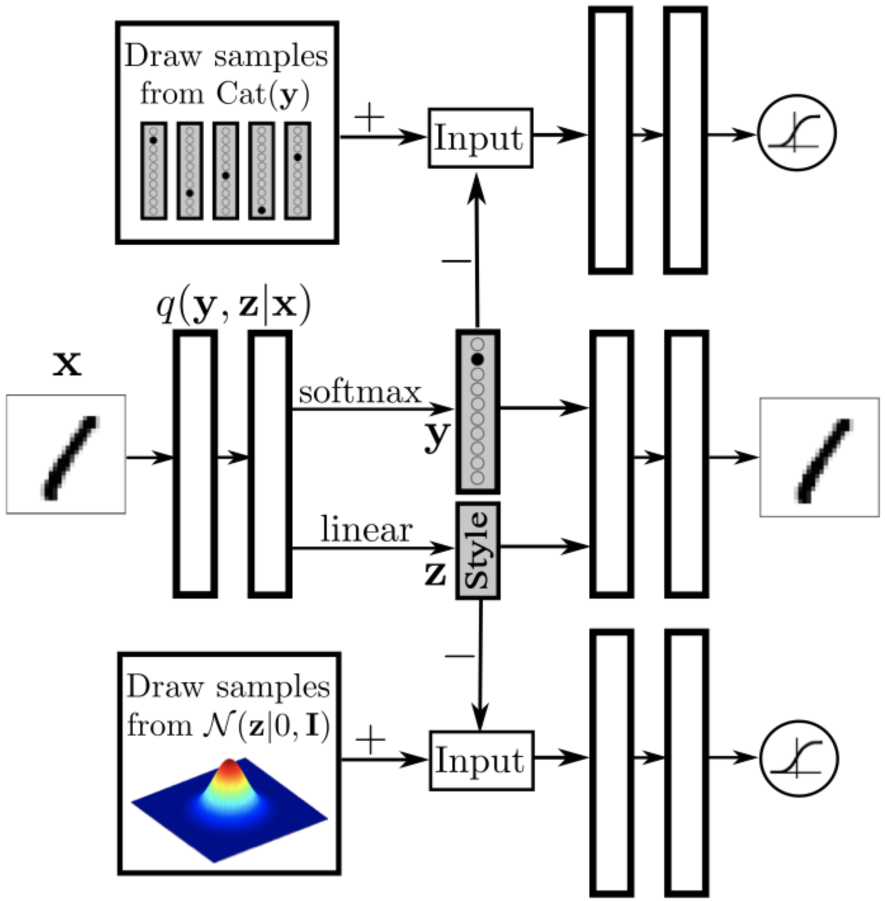
AAE – Style transfer (supervised)

Incorporate label information explicitly to force **z** to capture class-independent information (e.g. style)



https://arxiv.org/abs/1511.05644

AAE – Semi-supervised learning



Factorize latent code in

- One hot encoding vector **y**
- Continuous code **z**

Distribution of **y** made little distinguishable from a multinomial (induced from data)

https://arxiv.org/abs/1511.05644

Software

- A TF implementation of PixelCNN
- A list of acknowledged VAE implementations is kept by Kingma here
- Plenty of DCGAN implementations
  - Torch
  - Tensorflow
- Conditional GAN for image-to-image
  - Pytorch code
  - Demo
- So many GANs: check out the GAN-Zoo

Take Home Messages

- GAN – Learn to sample rather than learn the distribution
  - Sample quality only recently surpassed by diffusion models (tomorrow)
  - Unstable/difficult to train
  - Cannot perform inference (no distribution learning)
  - Needs differentiable generator
- Adversarial Autoencoders
  - Leveraging adversarial penalties in place of KL regularization
  - Useful to impose "complex" or empirical priors

Next Lecture

Back to explicit approaches

- Diffusion models
- Generative learning as a noising-denoising process
- Guided diffusion
- Conditional diffusion
- Latent space diffusion