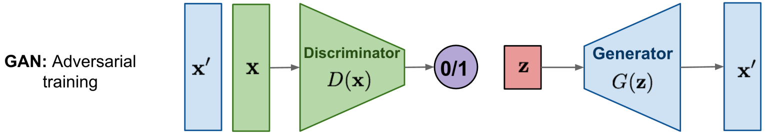
MAC5921 – Deep Learning

Aula 19 – 31/10/2023

GAN models

Nina S. T. Hirata

Vanilla GAN



https://www.rootstrap.com/blog/how-to-generate-images-with-ai/

$\mathbf{x}$: entrada real
$\mathbf{x}'$: entrada fake / imagem gerada por G
$\mathbf{z}$: vetor latente

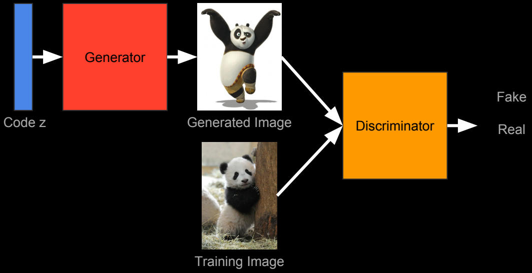
$$\min_G \max_D \mathcal{L}(D, G) = E_{\mathbf{x}}\left[\log D(\mathbf{x})\right] + E_{\mathbf{z}}\left[\log(1 - D(\mathbf{z}))\right]$$

Generative Adversarial Network (GAN)

Intuitions:

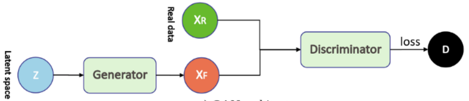
- Generator tries the best to cheat the discriminator by generating more realistic images

- Discriminator tries the best to distinguish whether the image is generated by computers or not

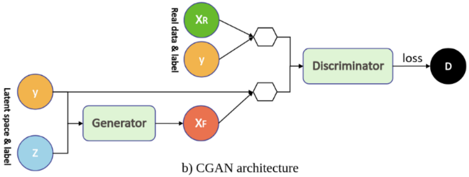


c-GAN

$$\min_{G} \max_{D} \mathcal{L}(D, G) = E_{\mathbf{x}}\left[\log D(\mathbf{x}|\mathrm{y})\right] + E_{\mathbf{z}}\left[\log(1 - D(\mathbf{z}|\mathrm{y}))\right]$$



a) GAN architecture

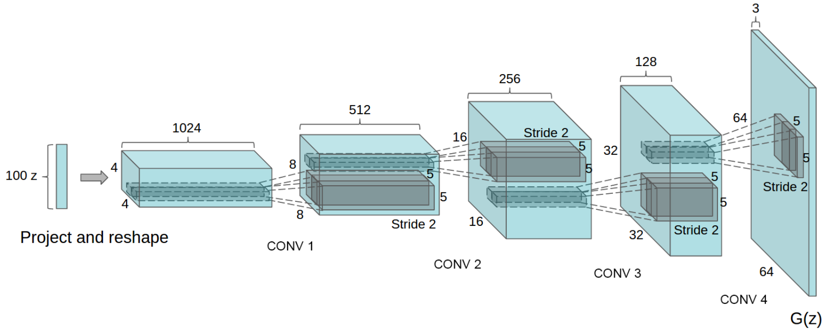


b) CGAN architecture

https://mc.ai/a-tutorial-on-conditional-generative-adversarial-nets-keras-implementation/

DCGAN – Deep Convolutional GAN

https://arxiv.org/abs/1511.06434



Architecture guidelines for stable Deep Convolutional GANs

- Replace any pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator).
- Use batchnorm in both the generator and the discriminator.
- Remove fully connected hidden layers for deeper architectures.
- Use ReLU activation in generator for all layers except for the output, which uses Tanh.
- Use LeakyReLU activation in the discriminator for all layers.

pix2pix – usa duas imagens pareadas

https://arxiv.org/abs/1611.07004

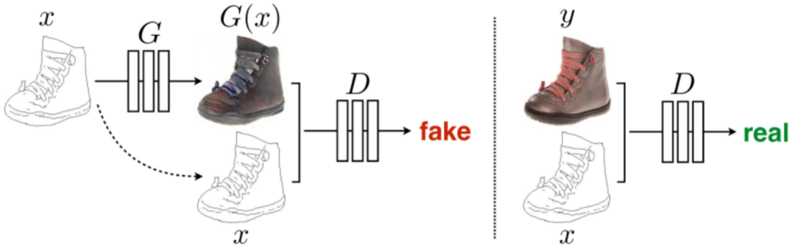
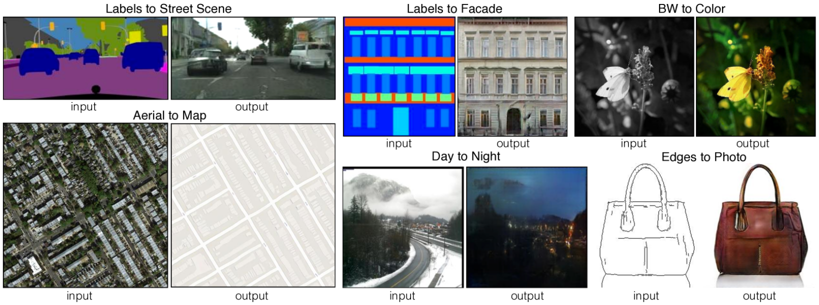


Figure 2: Training a conditional GAN to map edges→photo. The discriminator, D, learns to classify between fake (synthesized by the generator) and real {edge, photo} tuples. The generator, G, learns to fool the discriminator. Unlike an unconditional GAN, both the generator and discriminator observe the input edge map.

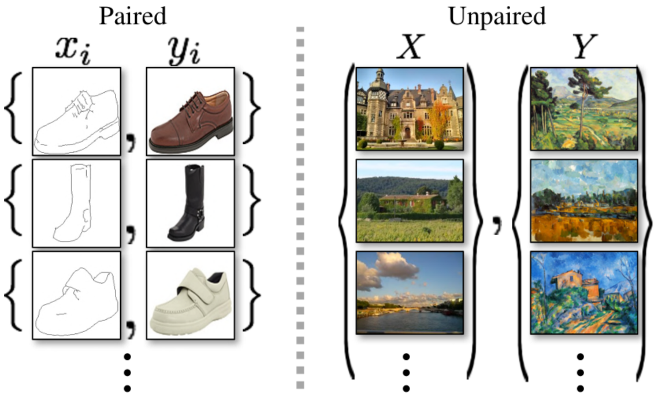
pix2pix

https://arxiv.org/abs/1611.07004



CycleGAN – usa duas imagens não pareadas

https://arxiv.org/pdf/1703.10593.pdf



CycleGAN

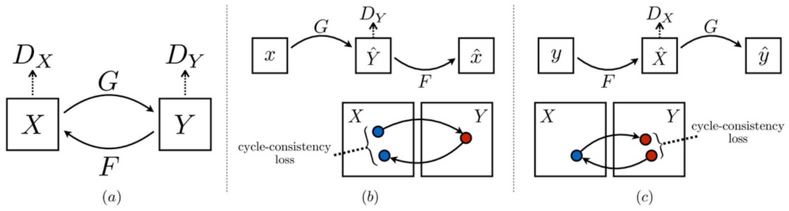
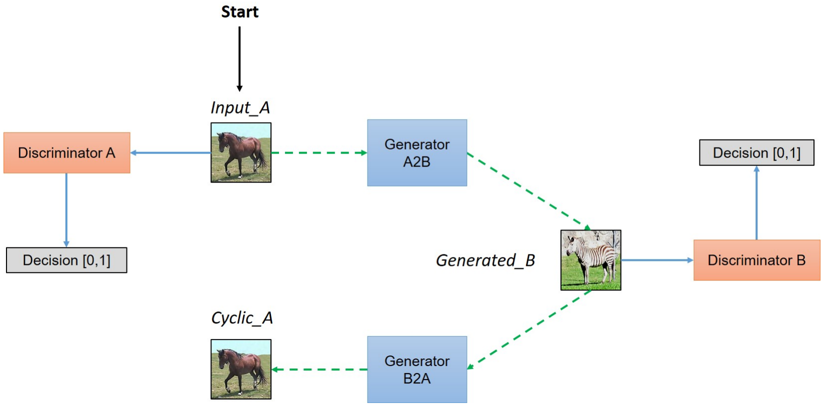
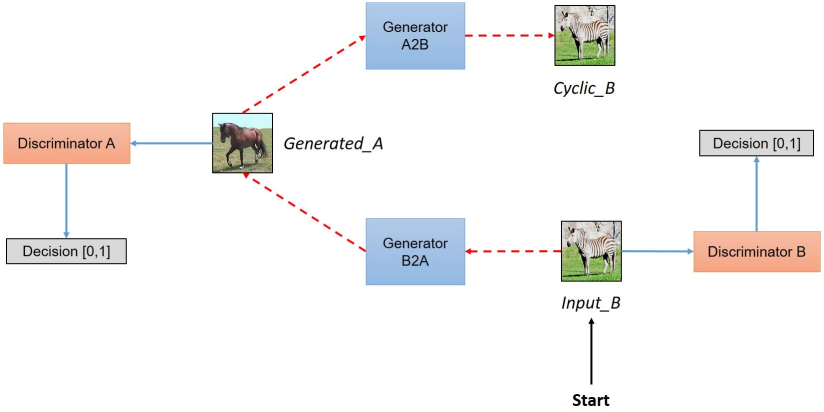


Figure 3: (a) Our model contains two mapping functions $G : X \to Y$ and $F : Y \to X$, and associated adversarial discriminators D_Y and D_X. D_Y encourages G to translate X into outputs indistinguishable from domain Y, and vice versa for D_X and F. To further regularize the mappings, we introduce two *cycle consistency losses* that capture the intuition that if we translate from one domain to the other and back again we should arrive at where we started: (b) forward cycle-consistency loss: $x \to G(x) \to F(G(x)) \approx x$, and (c) backward cycle-consistency loss: $y \to F(y) \to G(F(y)) \approx y$

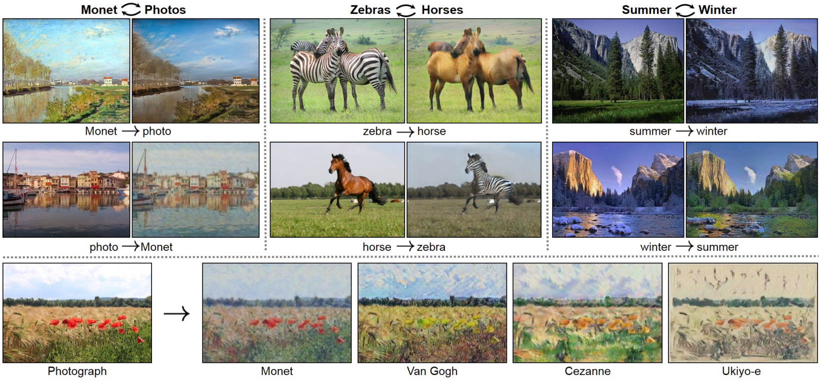
CycleGAN


Start
Input_A
Discriminator A
Decision [0,1]
Generator A2B
Decision [0,1]
Generated_B
Discriminator B
Cyclic_A
Generator B2A

CycleGAN

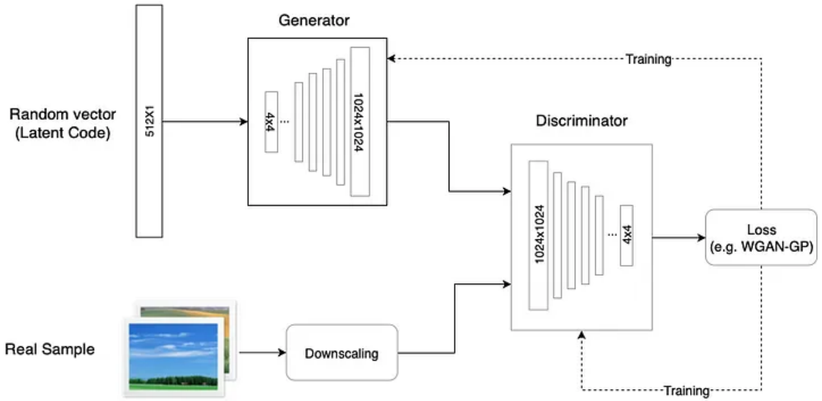

Generator A2B
Cyclic_B
Discriminator A
Generated_A
Decision [0,1]
Decision [0,1]
Generator B2A
Discriminator B
Input_B
Start

CycleGAN



Monet ⇄ Photos
Zebras ⇄ Horses
Summer ⇄ Winter
Monet → photo
zebra → horse
summer → winter
photo → Monet
horse → zebra
winter → summer
Photograph
Monet
Van Gogh
Cezanne
Ukiyo-e

proGAN

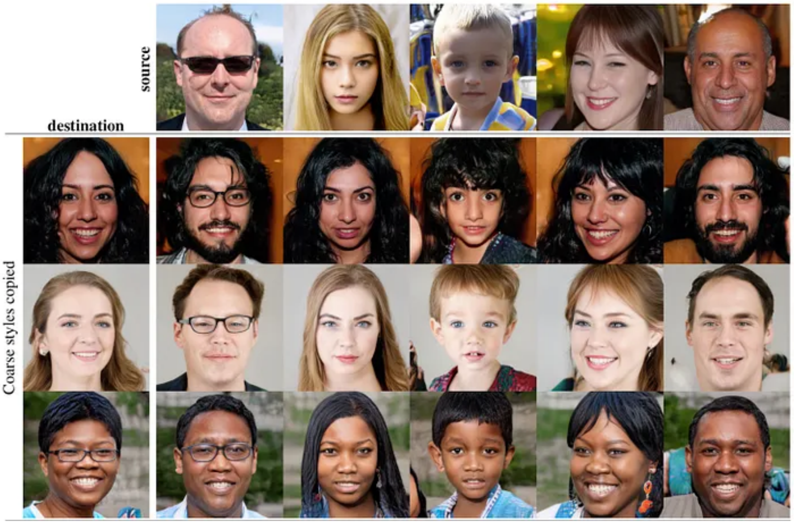

Generator
Training
Random vector
(Latent Code)
512X1
4x4
...
1024x1024
Discriminator
1024x1024
...
4x4
Loss
(e.g. WGAN-GP)
Real Sample
Downscaling
Training

Style GAN

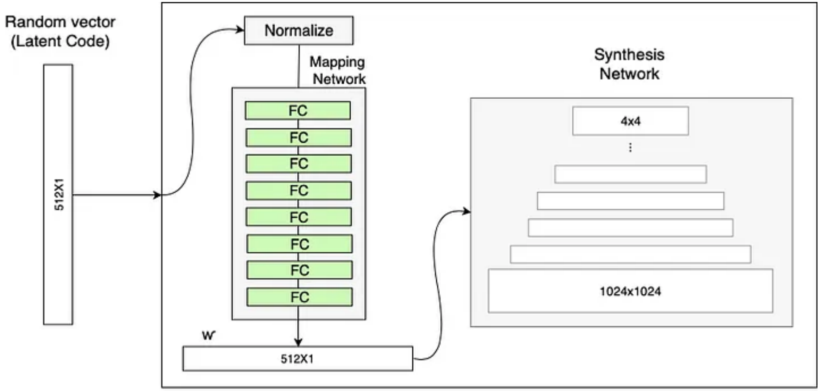
https://arxiv.org/abs/1812.04948

StyleGAN generates the artificial image gradually, starting from a very low resolution and continuing to a high resolution (1024×1024). By modifying the input of each level separately, it controls the visual features that are expressed in that level, from coarse features (pose, face shape) to fine details (hair color), without affecting other levels.

Style GAN

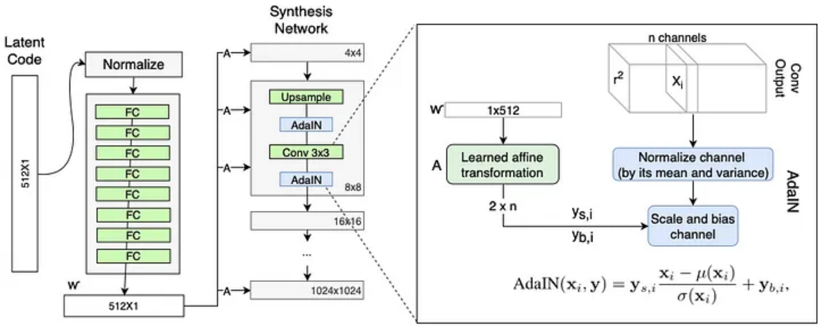


Style GAN – The Mapping Network



Gerar um novo vetor latente em que cada componente representa uma feature, menos correlacionados entre si

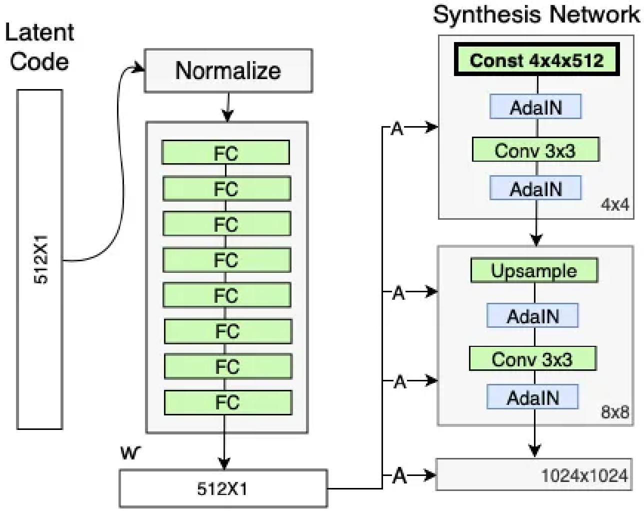
Style GAN – Style Modules (AdaIN)



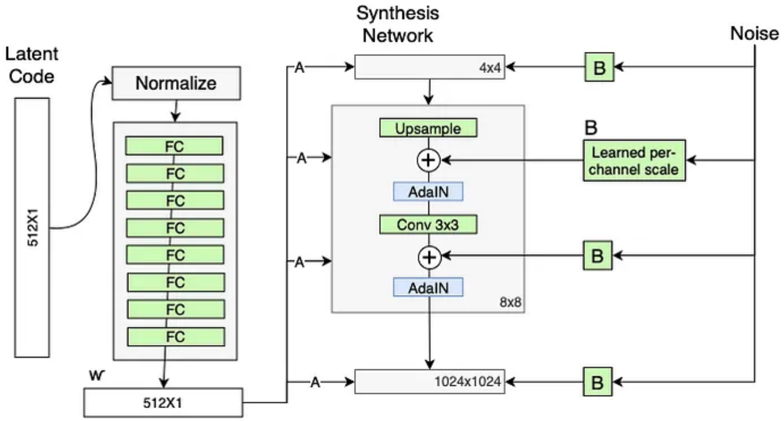
$$\mathrm{AdaIN}(\mathbf{x}_i, \mathbf{y}) = \mathbf{y}_{s,i} \frac{\mathbf{x}_i - \mu(\mathbf{x}_i)}{\sigma(\mathbf{x}_i)} + \mathbf{y}_{b,i},$$

Transfere o novo vetor latente para o módulo de síntese

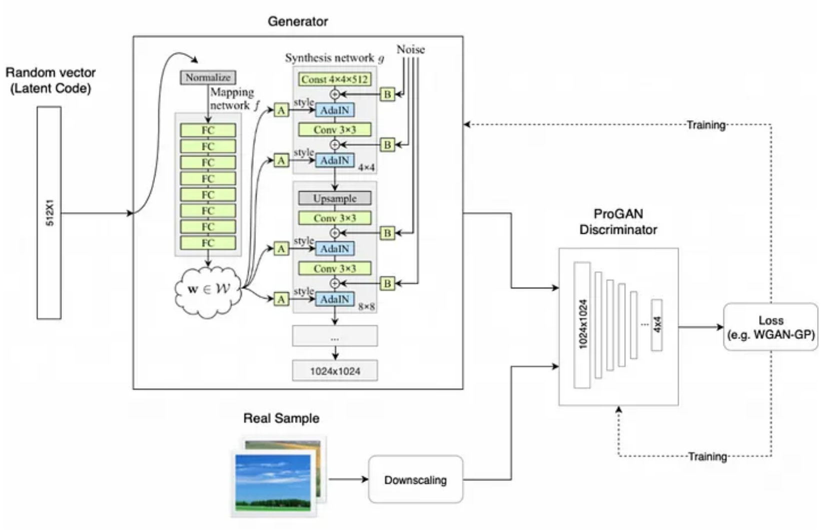
Style GAN – remover a entrada tradicional do módulo de síntese



Style GAN – adicionar variação estocástica a cada camada do módulo de síntese



Style GAN - arquitetura completa



Super Resolution GAN (SRGAN)

https://arxiv.org/pdf/1609.04802.pdf