Universidad Carlos III de Madrid

Generative Adversarial Networks (GANs)

Daniel Foronda-Pascual

Applications

- Image generation
- Style transfer
- Data augmentation
- Super-resolution
- Image-to-image translation
- Inpainting & restoration
- Text-to-image synthesis
- Synthetic medical data
- Deepfakes & facial reenactment

Image generation

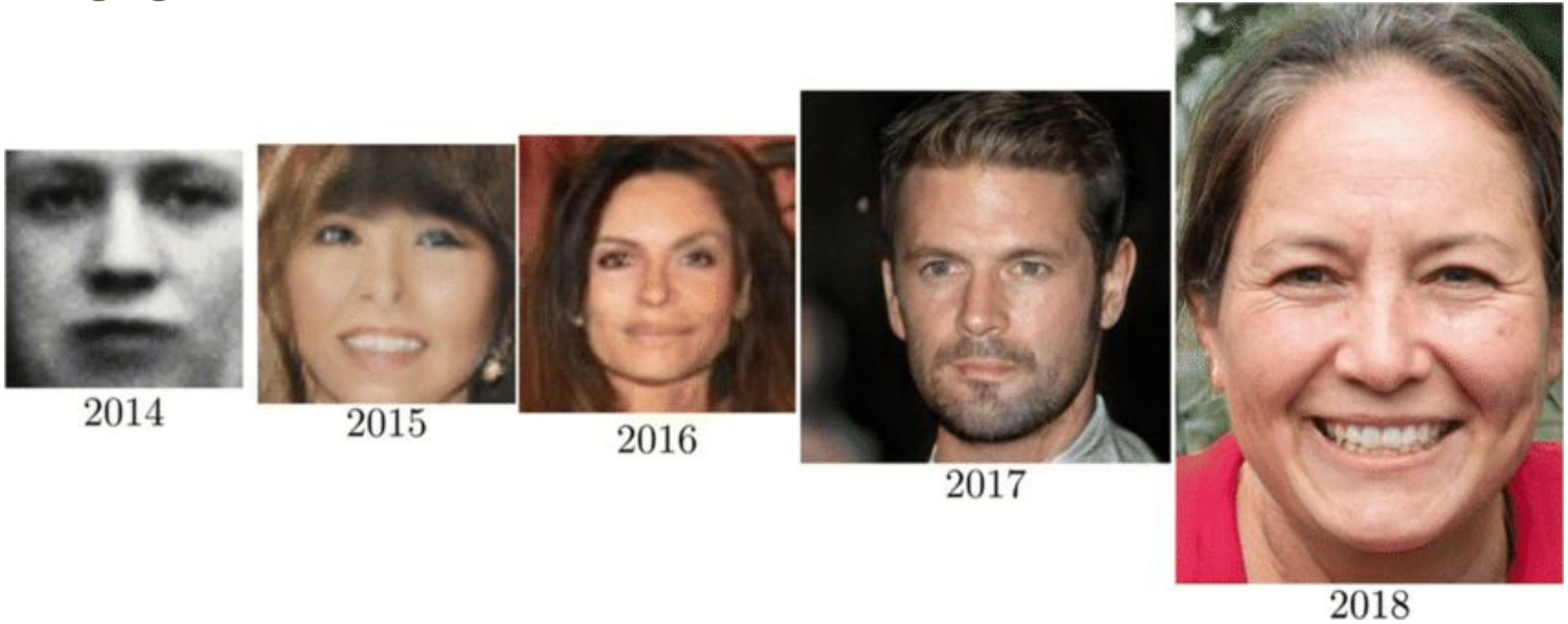
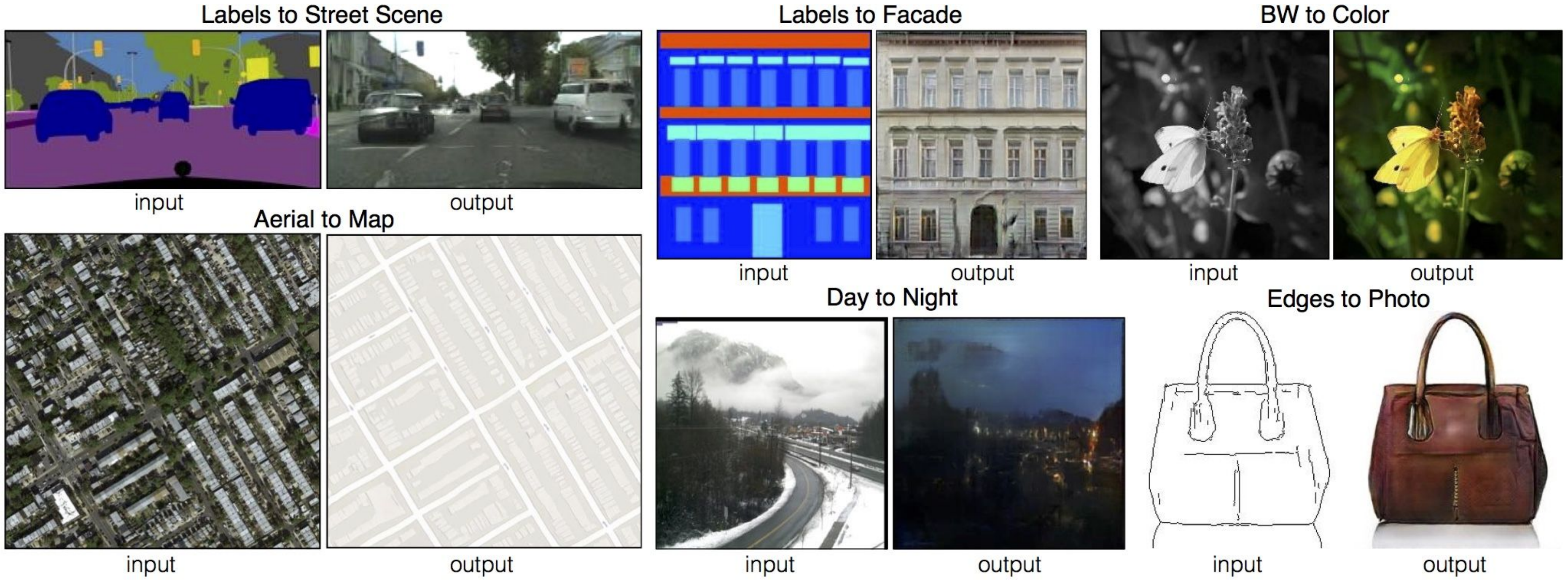
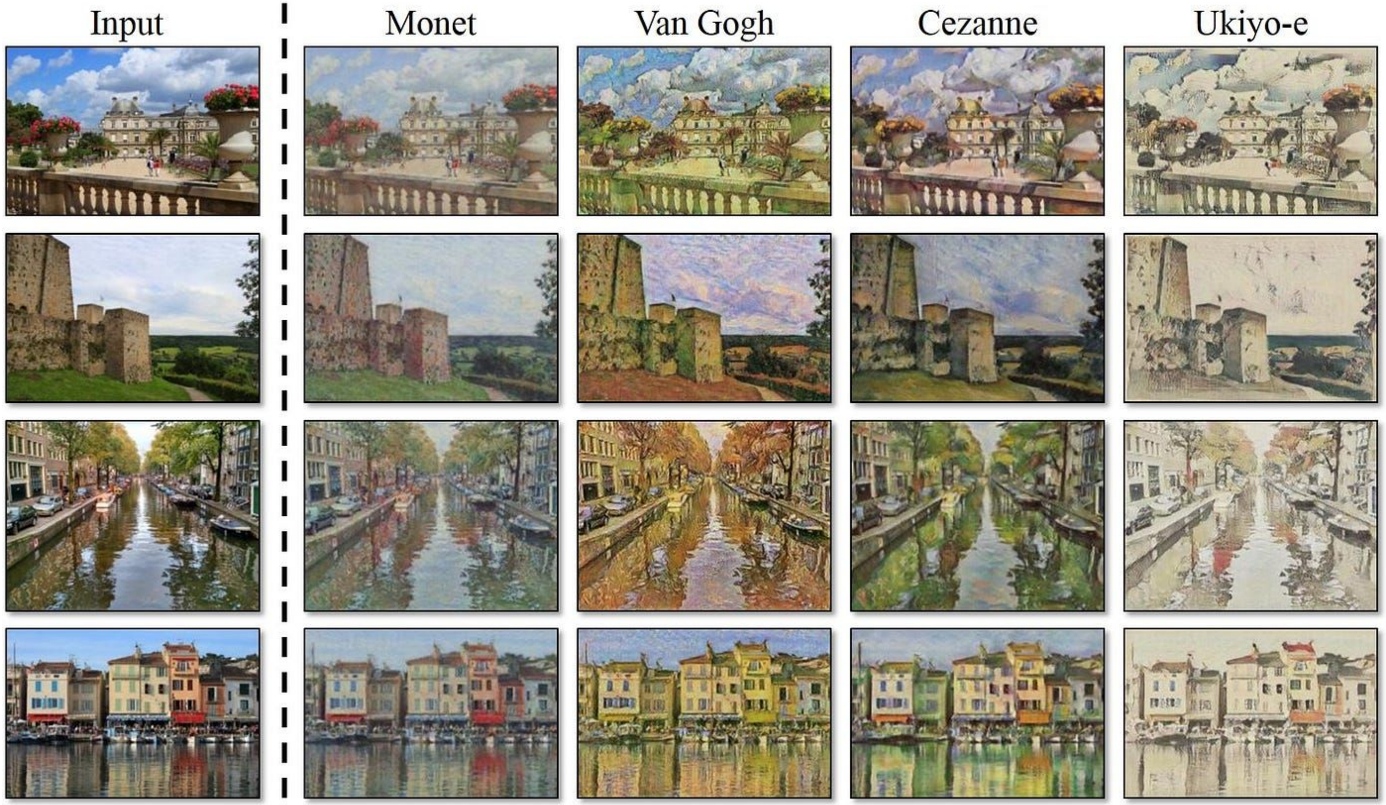


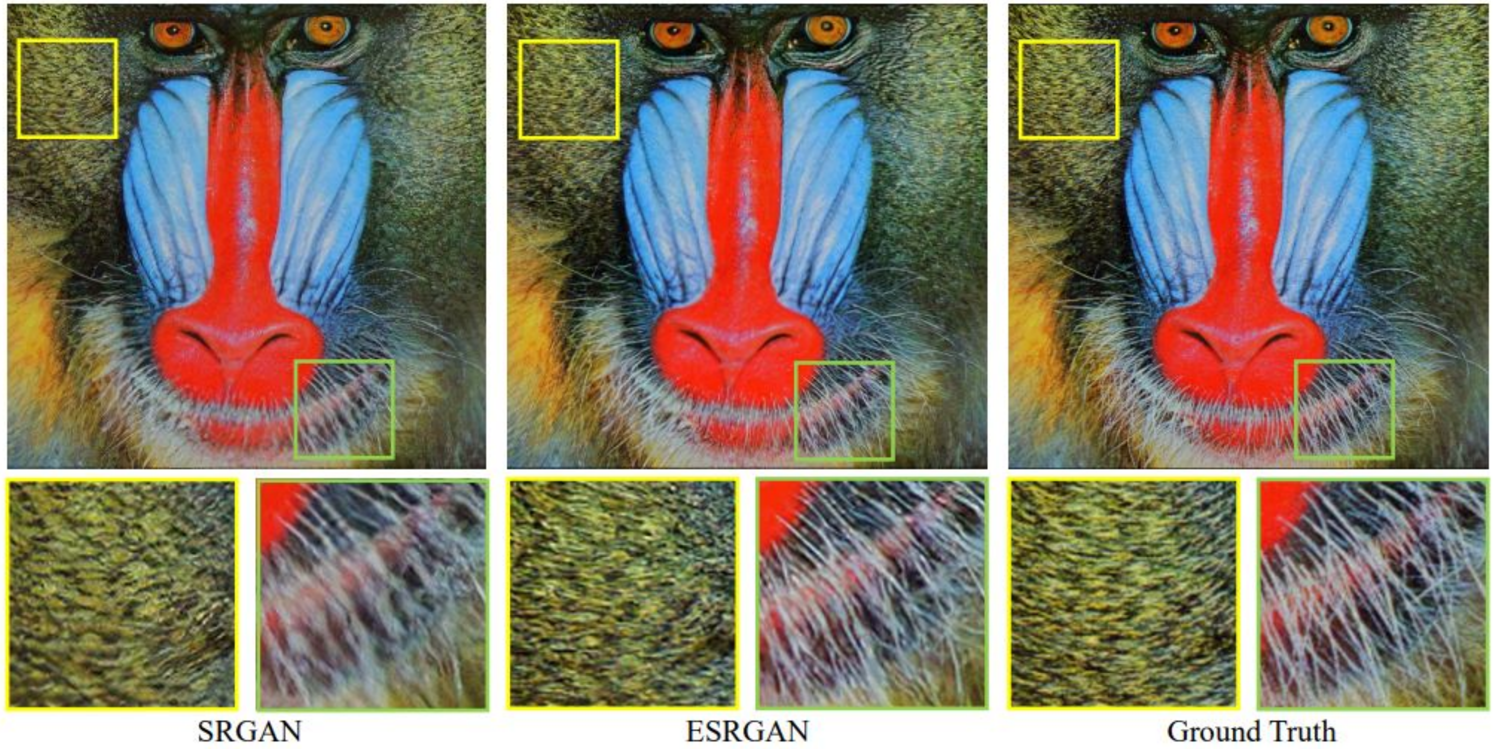
Image to image translation



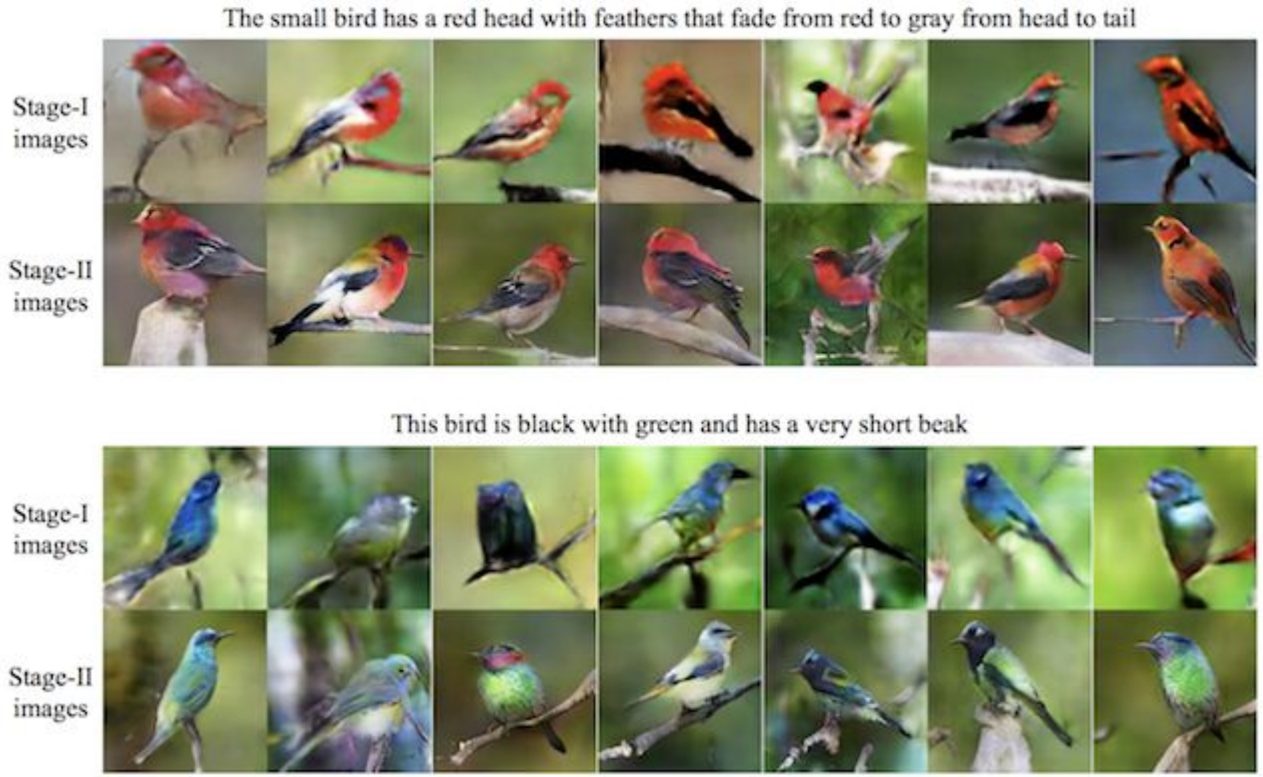
Style transfer



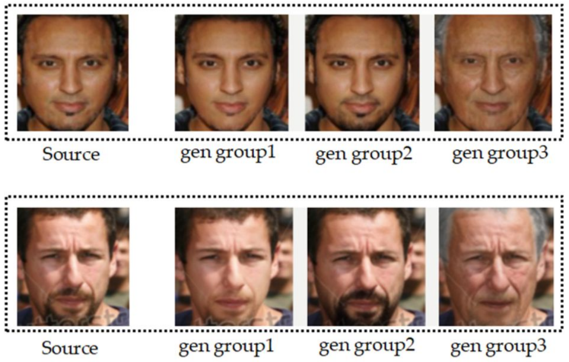
Super resolution



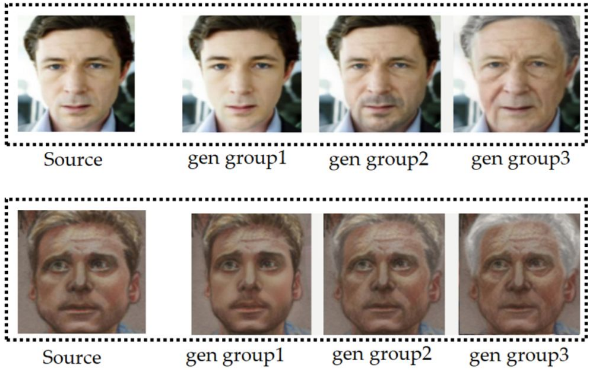
text2image


The small bird has a red head with feathers that fade from red to gray from head to tail
Stage-I images
Stage-II images
This bird is black with green and has a very short beak
Stage-I images
Stage-II images

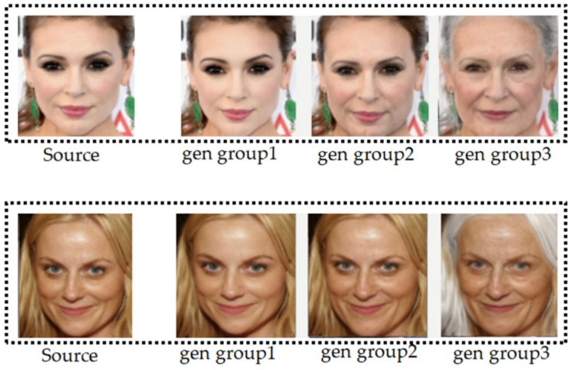
Facial rejuvenation and aging



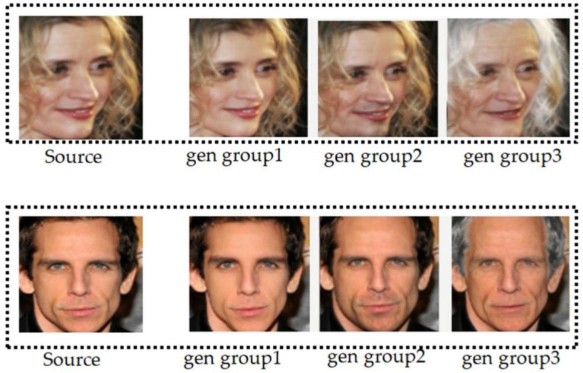
(a)



(b)

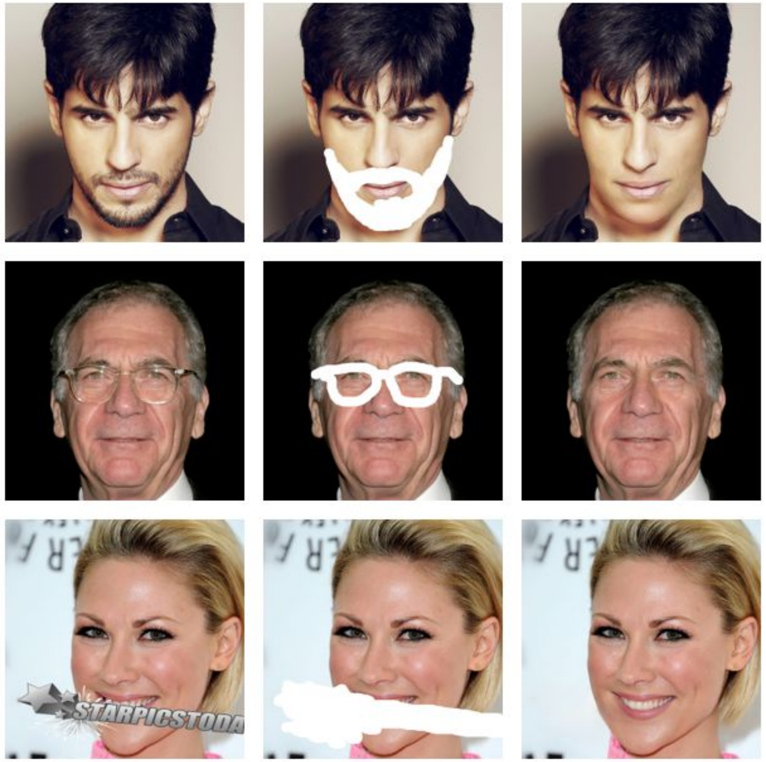


(c)

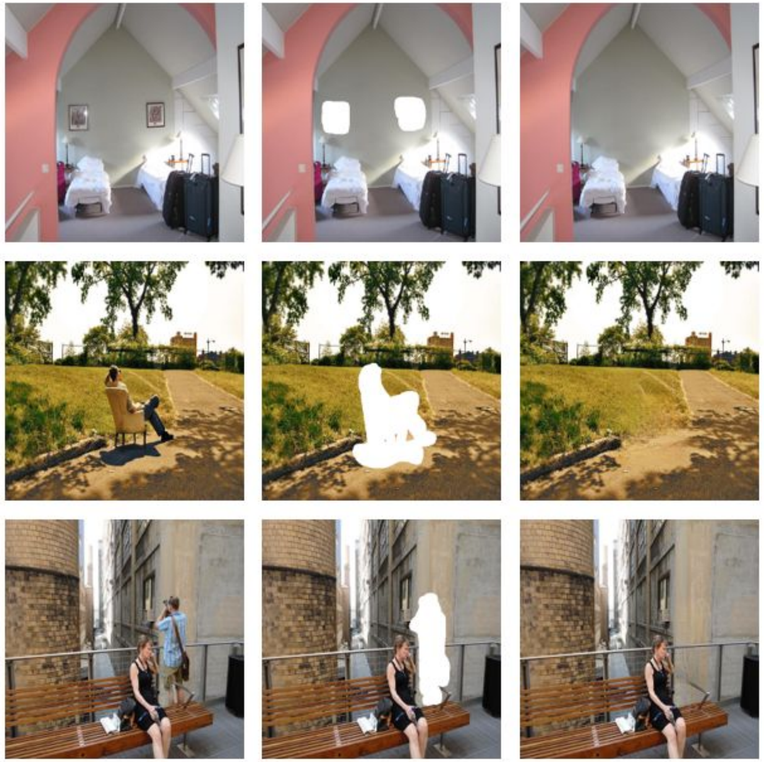


(d)

Fill missing parts of the image (inpainting)

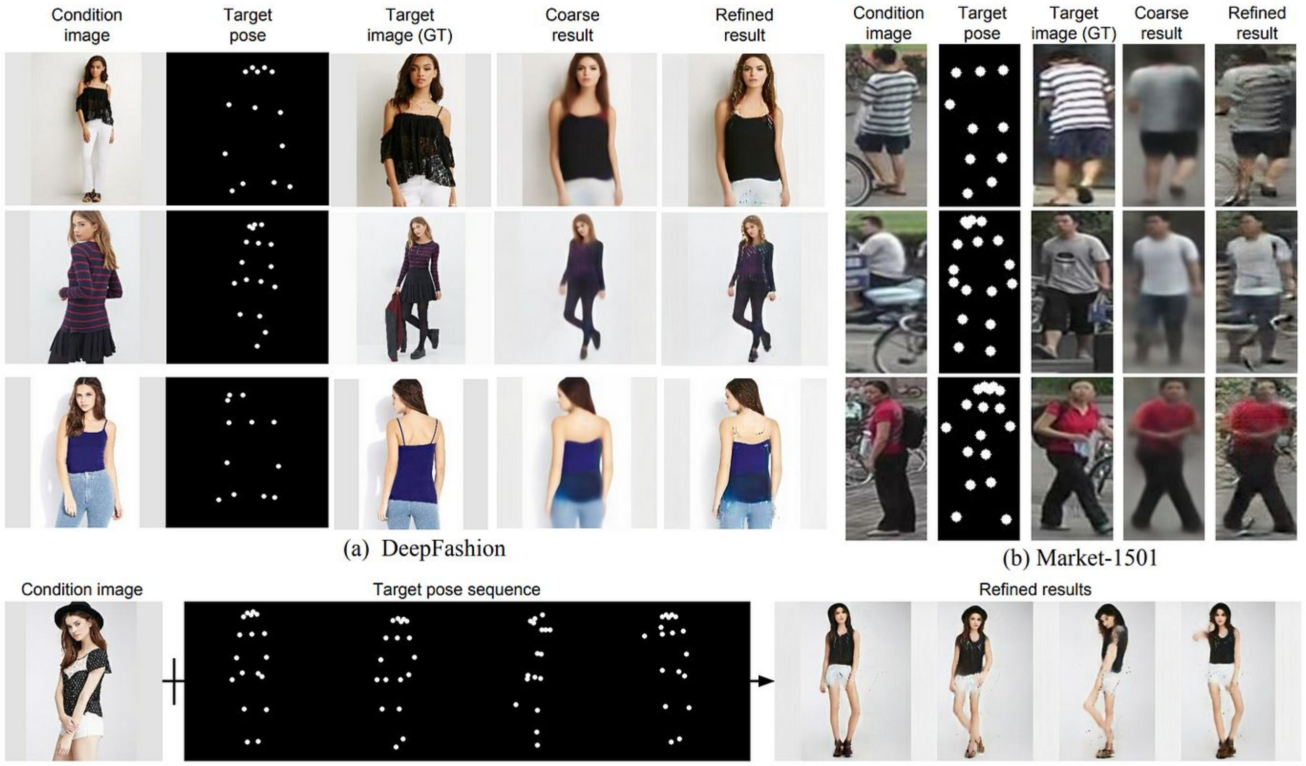


(a) Face Editing



(b) Object Removal

New human poses generation



(a) DeepFashion

(b) Market-1501

(c) Generating from a sequence of poses

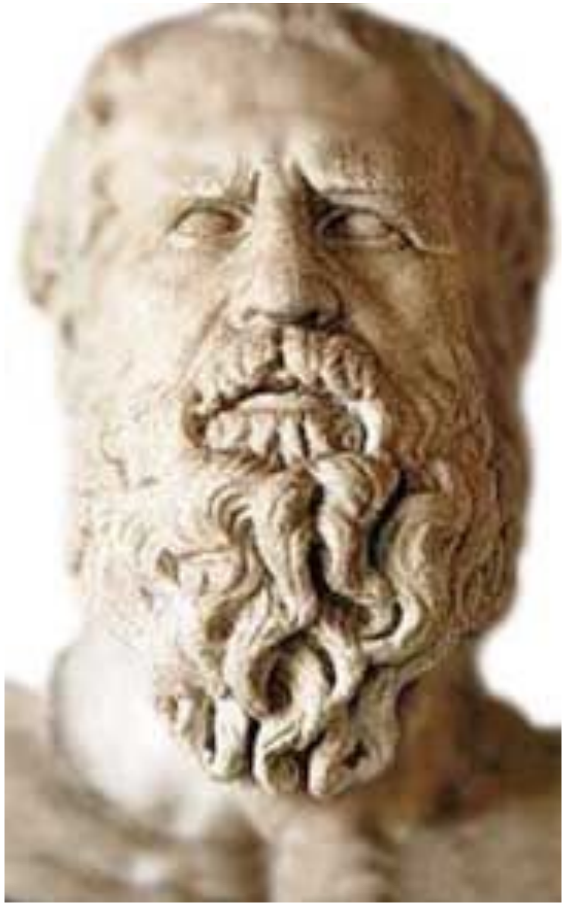
Great-grandfather of GANs

HERACLITUS OF EPHESUS (c. 535 – 480 BCE)

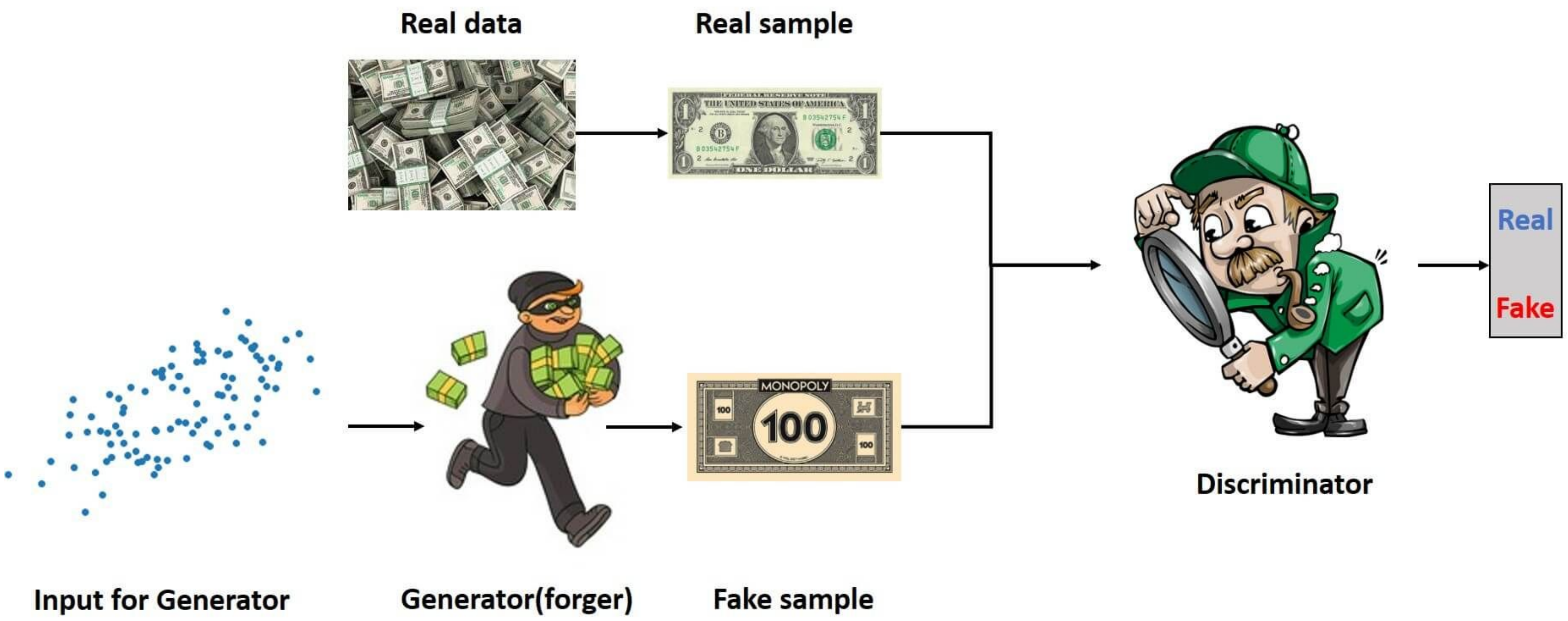
Opposition is the father of all things, the king of all.

Opposition brings concord. Out of discord comes the fairest harmony.

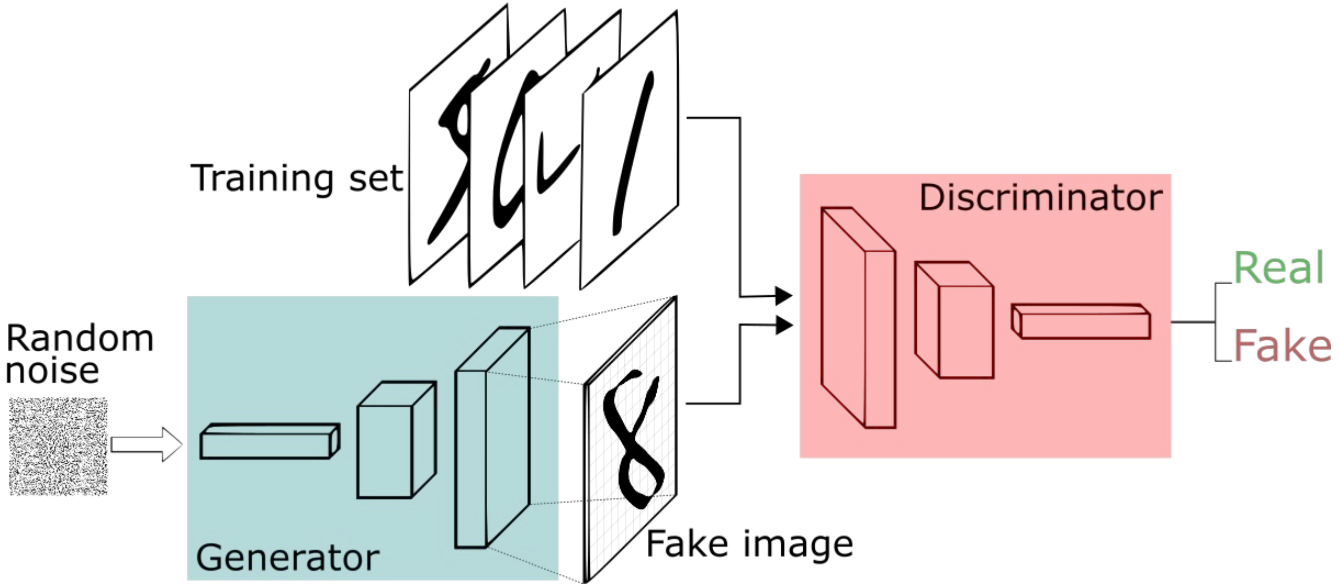
Everything changes, nothing remains the same.



How GANs work - Idea



How GANs work - Idea

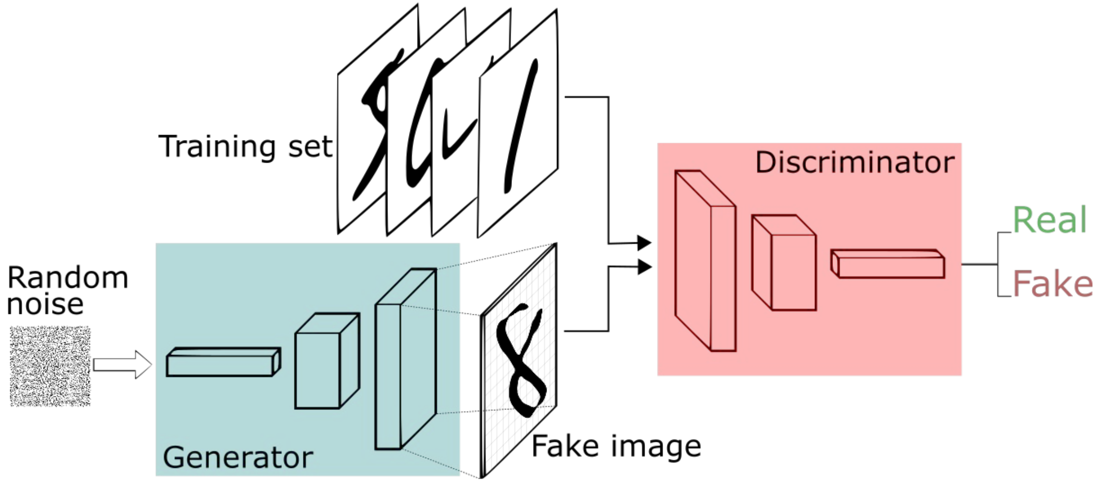


How GANs are trained - The minimax game

Generator: $$\frac{1}{m}\sum_{i=1}^{m} log(1 - D(G(z^{(i)})))$$

Minimize

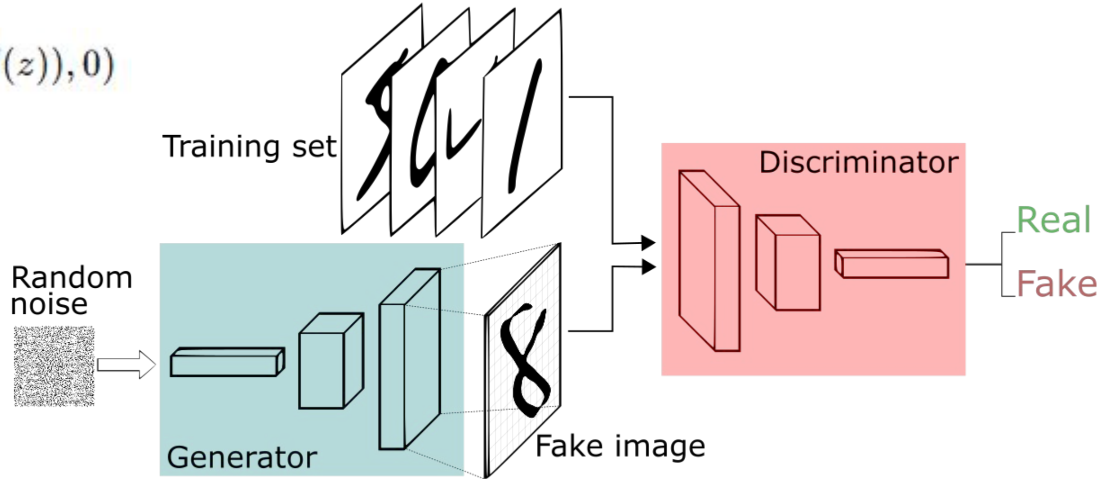
$$\mathcal{L}_G = \mathrm{BCE}(D(G(z)), 1)$$



How GANs are trained - The minimax game

Discriminator: $\frac{1}{m}\sum_{i=1}^{m}[logD(x^{(i)}) + log(1 - D(G(z^{(i)})))]$ *Maximize*

$$\mathcal{L}_D = \mathrm{BCE}(D(x), 1) + \mathrm{BCE}(D(G(z)), 0)$$



How GANs are trained - The minimax game

$$\min_G \max_D V(D, G) = \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

The Nash equilibrium of this particular game is achieved at:

- $P_{data}(x) = P_{gen}(x) \ \forall x$
- $D(x) = \frac{1}{2} \ \forall x$

Common training challenges

- **Mode collapse**: *Generator outputs lack diversity.*

→ Use minibatch discrimination, feature matching, WGAN

- **Vanishing gradients**: *Discriminator becomes too strong, generator stops learning.*

→ WGAN, label smoothing, or modify the loss.

- **Unstable training**: *Training fails to converge or oscillates.*

→ Use spectral normalization, gradient penalties, and tune hyperparameters.

Brief history (2014 - 2021)

🔹 **2014**
Ian Goodfellow et al. introduce GANs in *"Generative Adversarial Nets"* → breakthrough in unsupervised learning.
📄 Goodfellow, I. et al. (2014). *Generative Adversarial Nets*. NeurIPS. https://arxiv.org/abs/1406.2661

🔹 **2015–2016**
DCGAN improves image quality.
📄 Radford, A., Metz, L., & Chintala, S. (2016). *Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks*. ICLR. https://arxiv.org/abs/1511.06434
cGAN allows controlled generation.
📄 Mirza, M., & Osindero, S. (2014). *Conditional Generative Adversarial Nets*. arXiv preprint. https://arxiv.org/abs/1411.1784

🔹 **2017**
CycleGAN enables image-to-image translation without paired data.
📄 Zhu, J.-Y., Park, T., Isola, P., & Efros, A.A. (2017). *Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks*. ICCV. https://arxiv.org/abs/1703.10593
Wasserstein GAN (WGAN) introduces a more stable training method.
📄 Arjovsky, M., Chintala, S., & Bottou, L. (2017). *Wasserstein GAN*. ICML. https://arxiv.org/abs/1701.07875

🔹 **2018–2021**
StyleGAN / StyleGAN2 / StyleGAN3 raise the bar in photo-realistic synthesis.
📄 StyleGAN (2019): Karras, T. et al. *A Style-Based Generator Architecture for GANs*. CVPR. https://arxiv.org/abs/1812.04948
📄 StyleGAN2 (2020): Karras, T. et al. *Analyzing and Improving the Image Quality of StyleGAN*. CVPR. https://arxiv.org/abs/1912.04958
📄 StyleGAN3 (2021): Karras, T. et al. *Alias-Free Generative Adversarial Networks*. NeurIPS. https://arxiv.org/abs/2106.12423
BigGAN enables high-res generation at scale.
📄 Brock, A., Donahue, J., & Simonyan, K. (2019). *Large Scale GAN Training for High Fidelity Natural Image Synthesis*. ICLR. https://arxiv.org/abs/1809.11096

DCGAN (Deep Convolutional GAN)

📄 Radford, A., Metz, L., & Chintala, S. (2016). Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. ICLR. https://arxiv.org/abs/1511.06434

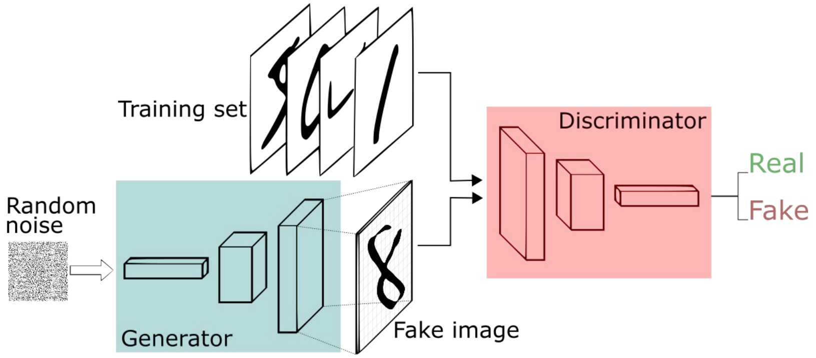
Key idea: Use CNN architectures for both Generator and Discriminator

Main Contributions:

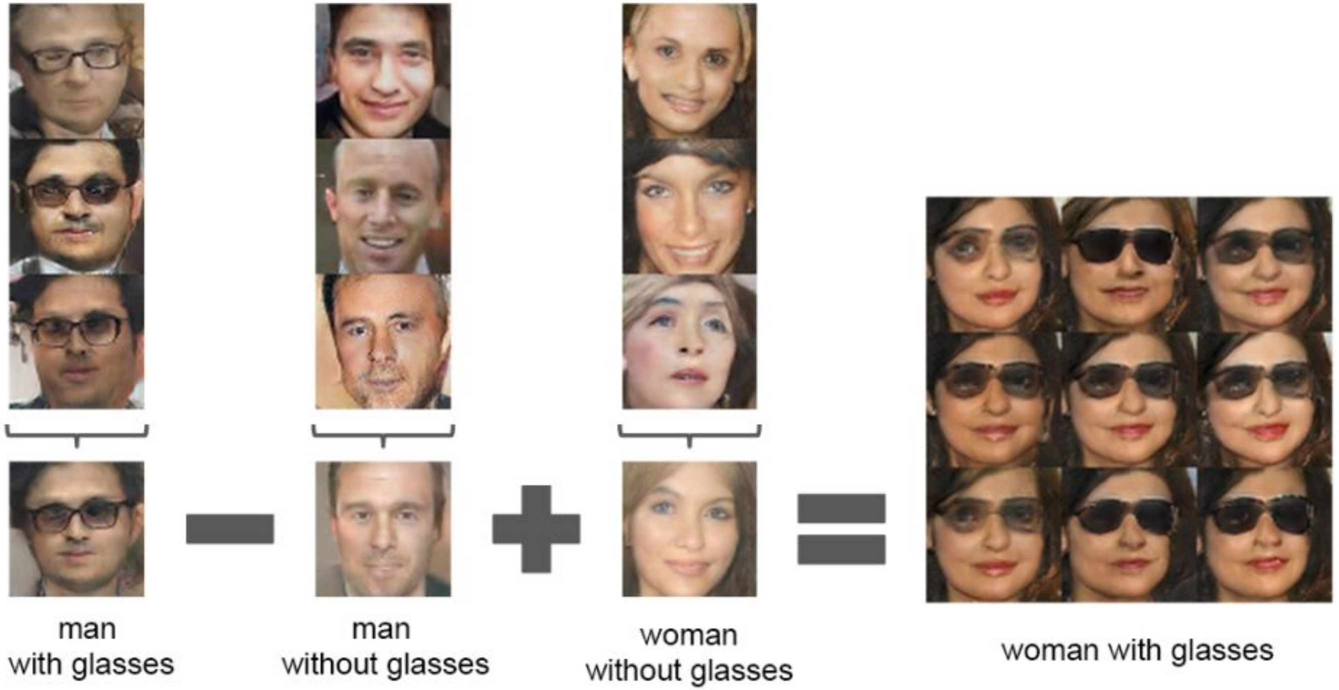
- Replace fully connected layers with convolutions / transposed convolutions
- Use batch normalization for stable training
- Apply ReLU in Generator, LeakyReLU in Discriminator
- Remove pooling layers → let strides do the down/up-sampling

Advantages:

- More stable training
- Better image quality
- Simple and effective architecture for image generation tasks



DCGAN (Deep Convolutional GAN)



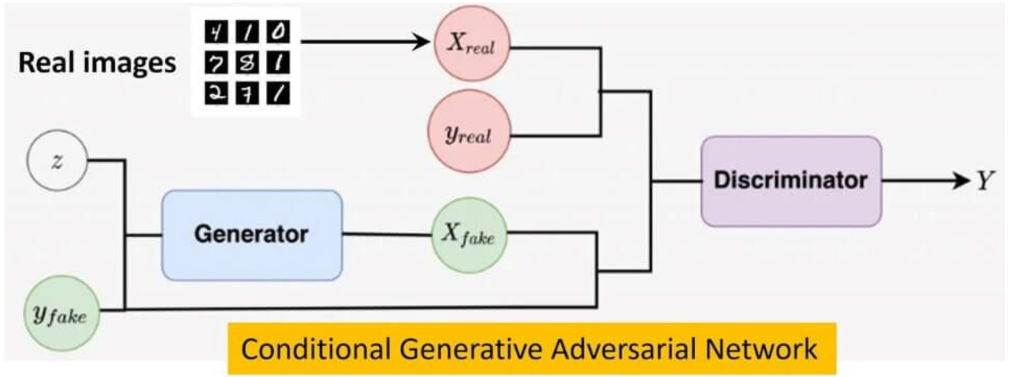
cGAN (conditional GAN)

📄 Mirza, M., & Osindero, S. (2014). Conditional Generative Adversarial Nets. arXiv preprint. https://arxiv.org/abs/1411.1784

Key idea: Add conditioning information (e.g., class labels) to both Generator and Discriminator

How it works

- Generator receives input noise z and label y → generates image G(z | y)
- Discriminator evaluates whether an image is real/fake given label y → D(x | y)



CycleGAN

📄 Zhu, J.-Y., Park, T., Isola, P., & Efros, A.A. (2017). Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. ICCV. https://arxiv.org/abs/1703.10593

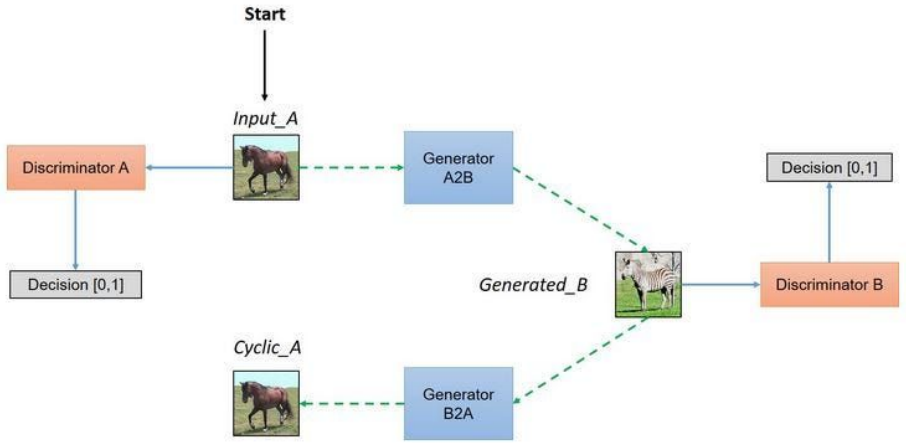
Key idea: Learn image-to-image translation between two domains **without paired data**

Losses:

- Adversarial loss: Two GAN losses (one for each translation direction) encourage the generated images to be indistinguishable from real images in the target domain.
- Cycle-consistency loss: Ensures that translating an image to the other domain and back reconstructs the original (e.g., A → B → A ≈ A). This enforces content preservation.
- Identity loss (optional): Penalizes changes when the input image already belongs to the target domain, helping preserve colors and structure.

Applications:

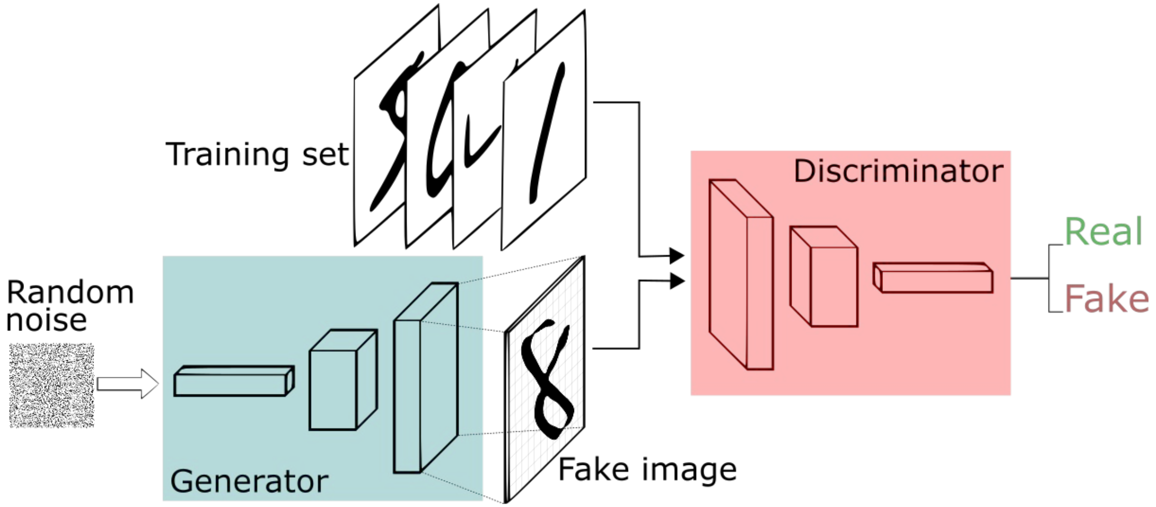
- Artistic style transfer (painting ↔ photo)
- Object translation (apples ↔ oranges)
- Medical image translation (MRI ↔ CT)



WGAN (Wasserstein GAN)

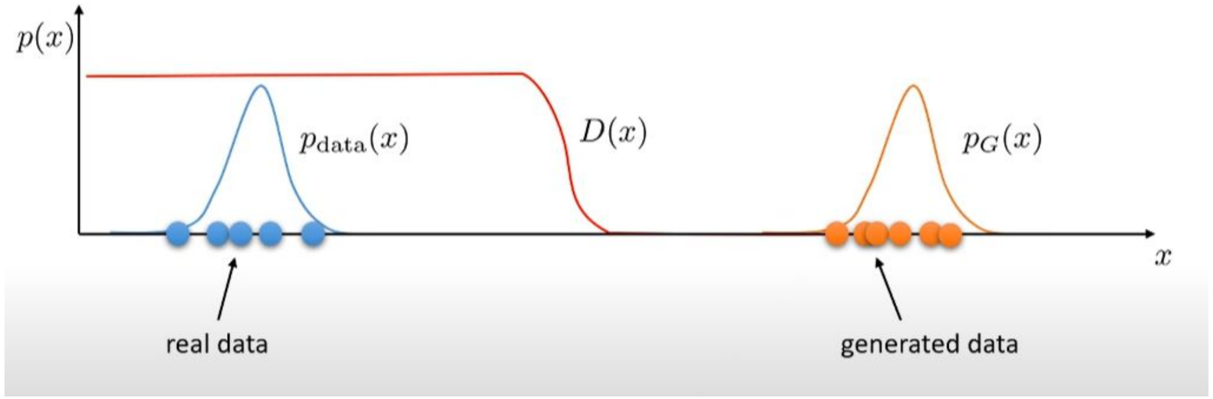
📄 Arjovsky, M., Chintala, S., & Bottou, L. (2017). Wasserstein GAN. ICML. https://arxiv.org/abs/1701.07875

Problem: When the discriminator gets too good, it stops giving useful feedback to the generator—gradients vanish, learning slows down, and training becomes unstable.



WGAN (Wasserstein GAN)

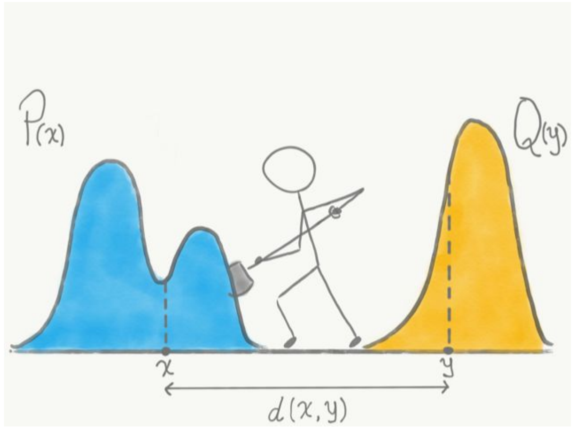
Problem: Vanishing gradients when Generator is not good



$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

WGAN (Wasserstein GAN)

Wasserstein distance (Earth mover’s distance): It measures the minimum effort required to transform one distribution into another



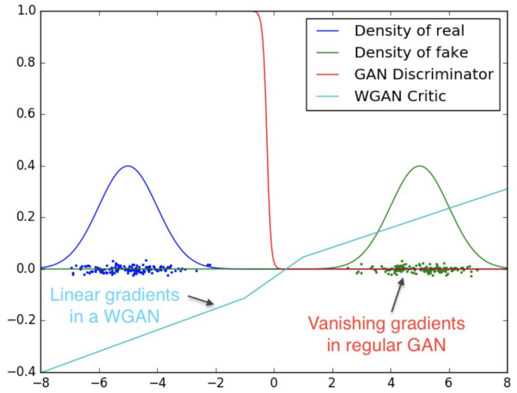
$$W(p_{\text{data}}, p_G) = \inf_{\gamma} E_{(x,y)\sim\gamma(x,y)}[||x-y||]$$

$\uparrow$

$\gamma(x,y)$ is a *distribution* over x, y
with marginals $\gamma_X(x) = p_{\text{data}}(x)$ and $\gamma_Y(x) = p_G(x)$

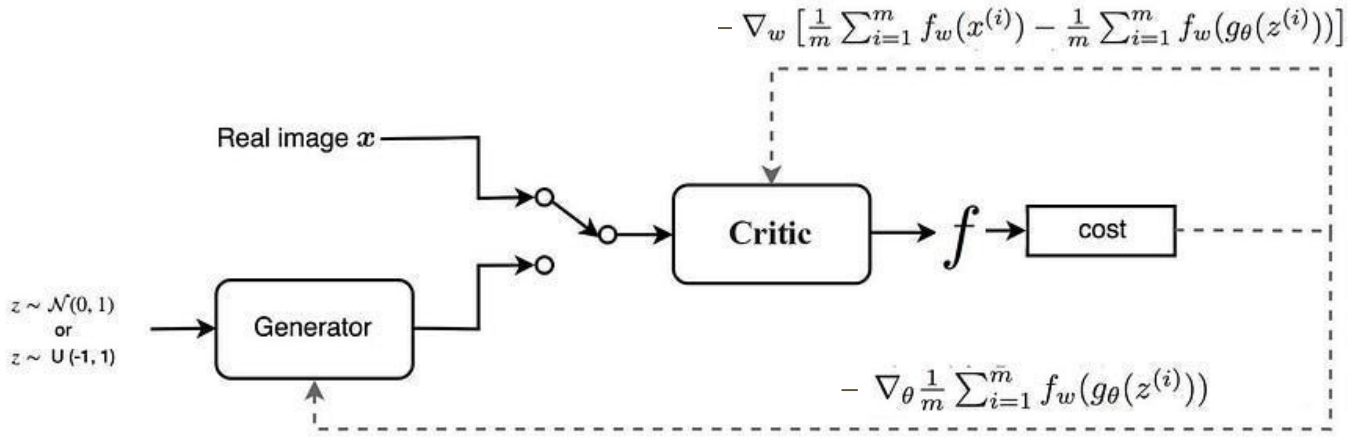
dualidad de Kantorovich-Rubinstein

$$W(p_{\text{data}}, p_G) = \sup_{||f||_L \leq 1} E_{p_{\text{data}}}[f(x)] - E_{p_G(x)}[f(x)]$$



WGAN (Wasserstein GAN)

Discriminator now is called Critic: The critic assigns higher scores to more realistic images and lower scores to fake ones, reflecting their distance from the real data distribution.



$-\nabla_w \left[\frac{1}{m}\sum_{i=1}^{m} f_w(x^{(i)}) - \frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)}))\right]$
Real image x
Critic
f
cost
$z \sim \mathcal{N}(0,1)$
or
$z \sim$ U (-1, 1)
Generator
$-\nabla_\theta \frac{1}{m}\sum_{i=1}^{m} f_w(g_\theta(z^{(i)}))$

1-Lipschitz Constraint in WGANs

To approximate the Wasserstein distance correctly, the critic must be a 1-Lipschitz function. Different techniques have been proposed to enforce this:

- **Weight Clipping** (WGAN, 2017)

  Clip critic weights to a fixed range (e.g., [−0.01, 0.01])

  Simple but can lead to capacity issues and poor gradients

- **Gradient Penalty** (WGAN-GP, 2017)

  Add a penalty term to the loss to enforce that the gradient norm is close to 1:

  $$\lambda \cdot (\|\nabla_{\hat{x}} f(\hat{x})\|_2 - 1)^2$$

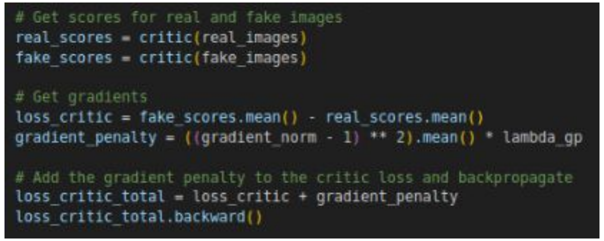
  where $\hat{x}$ is interpolated between real and fake samples

  More stable and widely used

- **Spectral Normalization** (2018)

  Normalize each layer's weight matrix by its largest singular value (spectral norm)

  Lightweight and effective; used in many modern GANs



```
# Get scores for real and fake images
real_scores = critic(real_images)
fake_scores = critic(fake_images)

# Get gradients
loss_critic = fake_scores.mean() - real_scores.mean()
gradient_penalty = ((gradient_norm - 1) ** 2).mean() * lambda_gp

# Add the gradient penalty to the critic loss and backpropagate
loss_critic_total = loss_critic + gradient_penalty
loss_critic_total.backward()
```

GANs in 2025

Today GANs are no longer dominant in image generation.

Where GANs remain useful:

- ⚡ Fast inference for real-time applications
- 🧠 Low-data regimes where training data is limited
- 🔧 Custom setups with tight resource constraints
- 🎨 Creative tools in art, design, and music

$\hat{x}$

Challenges remain:

- Training instability
- Mode collapse
- Difficulty scaling to complex data

Competing with:

- Diffusion Models → better quality & stability
- Transformers → better semantic control
- Latent models → efficient + high-fidelity

- More than **55,000 people** have been reported **killed.**
- **80% of Palestinians killed are civilians**.
- **70% of the Palestinians killed in residential buildings or similar housing were women and children.**

Wikipedia

RAHMET
OBRIGADO
GRAZIE
HVALA
谢谢
KIITOS
DANKE
TAK
DANK JE
THANK YOU
SHUKRAAN
ευχαριστώ
GRACIAS
BARKA
MERCI
СПАСИБО
ARIGATO
TAKK
TACK

Jupyter notebook

https://www.kaggle.com/code/rafat97/pytorch-wasserstein-gan-wgan