Quickie : Les GANs

Generative Adversarial Networks

Generative Adversarial Network

Generative => Générateur

Adversarial => Discriminateur

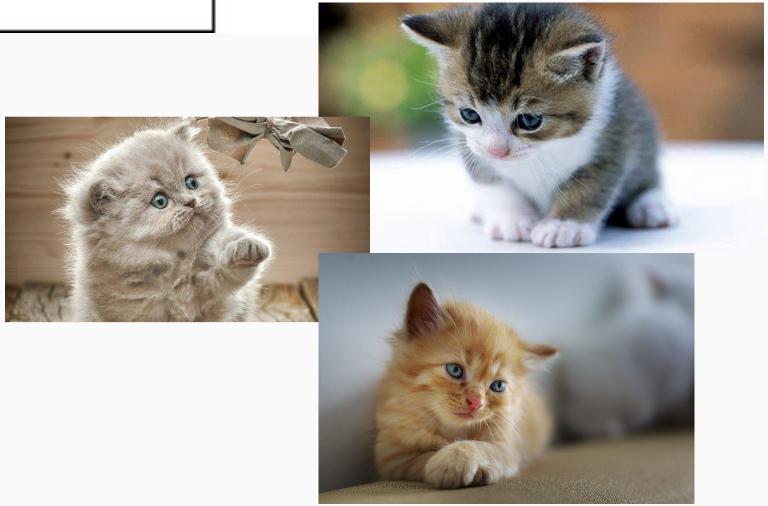
Exemples en images 2D mais applicables à d’autres domaines !

Qu’est-ce qu’un générateur ?

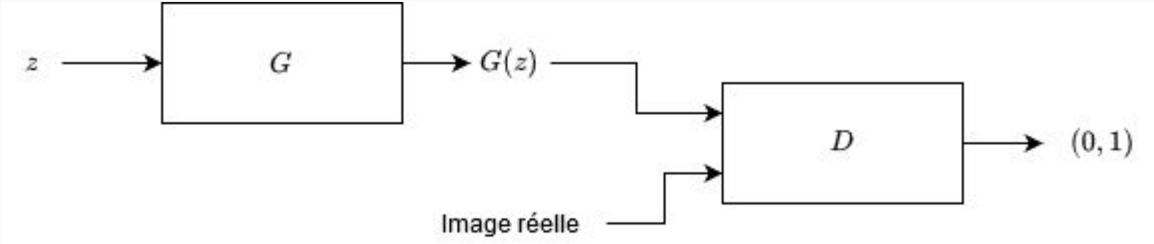
ex: $z \sim N(0,1) \; ou \; z \sim U(-1,1)$

$$z \longrightarrow \boxed{\quad G \quad} \longrightarrow G(z)$$

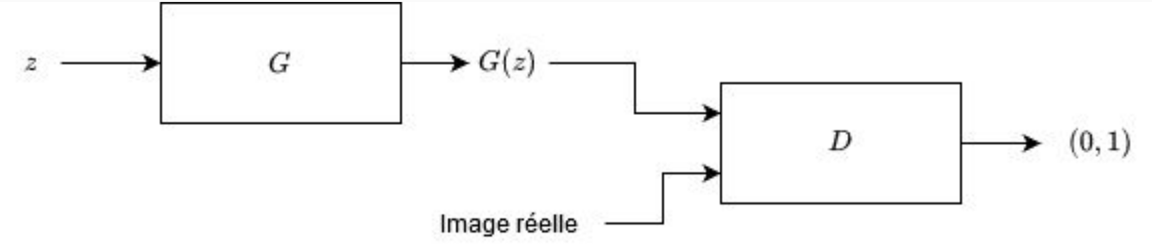
$$z = (0.1, 0.2, -0.3, \dots)$$



Que viens faire un discriminateur ?



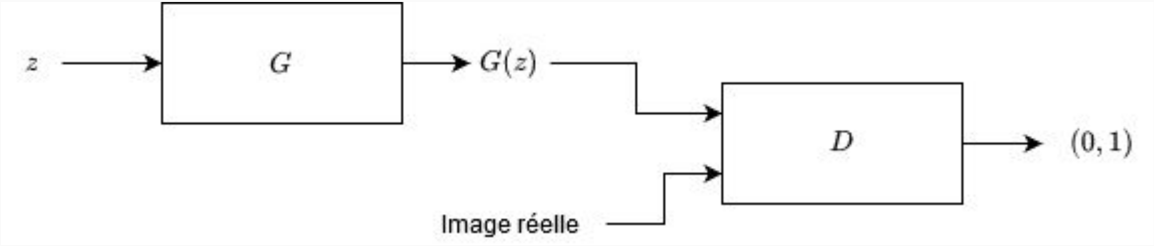
Que viens faire un discriminateur ?



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

$$\min_{G} V(G) = \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$$

Que viens faire un discriminateur ?



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

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 $\{\boldsymbol{z}^{(1)}, \dots, \boldsymbol{z}^{(m)}\}$ 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:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

end for

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \dots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

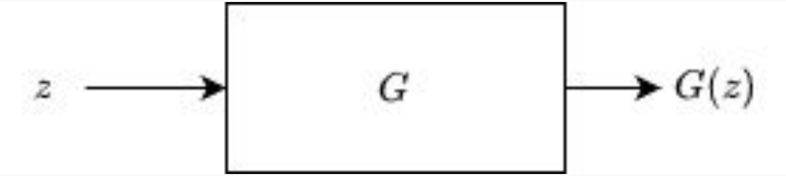
end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu,D. Warde-Farley, S. Ozair, A. Courville, and Y. Ben-gio. *Generative adversarial nets*. In NIPS, 2014.

Contrôle des features

Comment contrôler z ?



Contrôle des features : Conditionner z

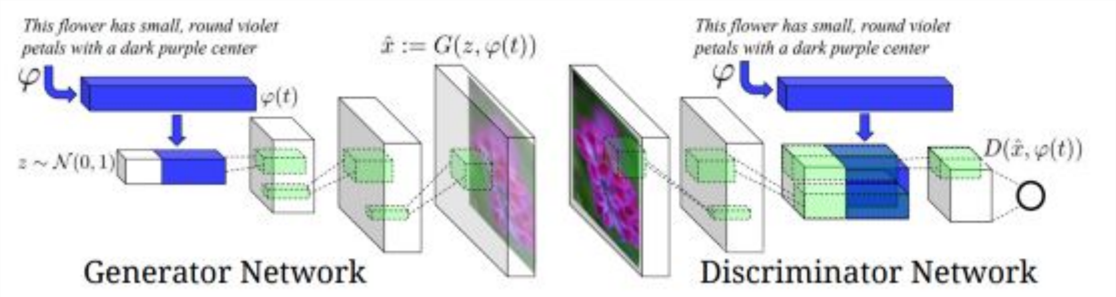
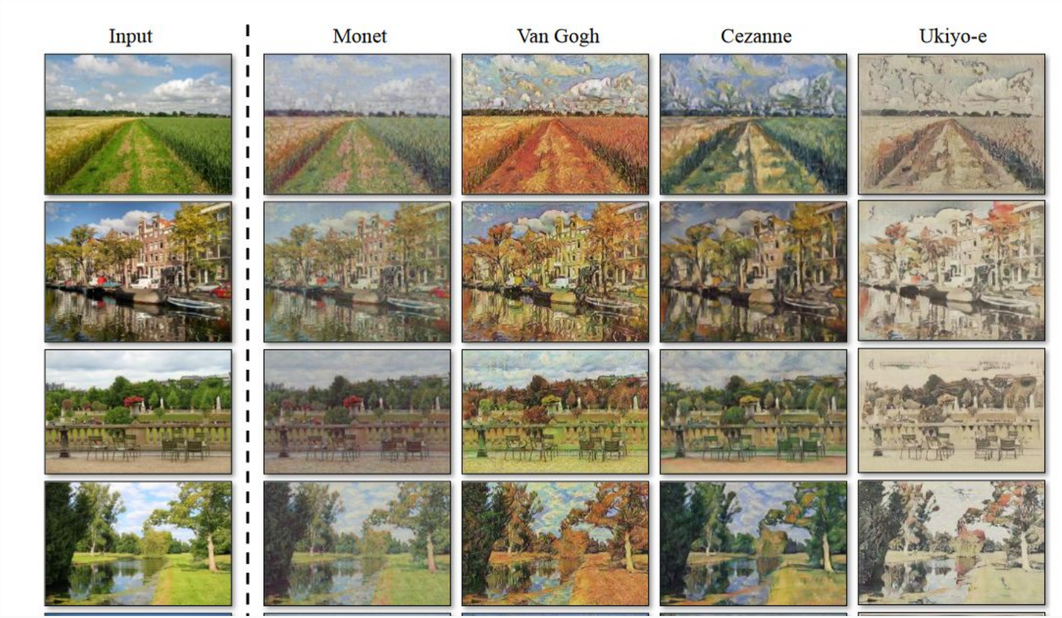


Figure 2. Our text-conditional convolutional GAN architecture. Text encoding $\varphi(t)$ is used by both generator and discriminator. It is projected to a lower-dimensions and depth concatenated with image feature maps for further stages of convolutional processing.

Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, Honglak Lee

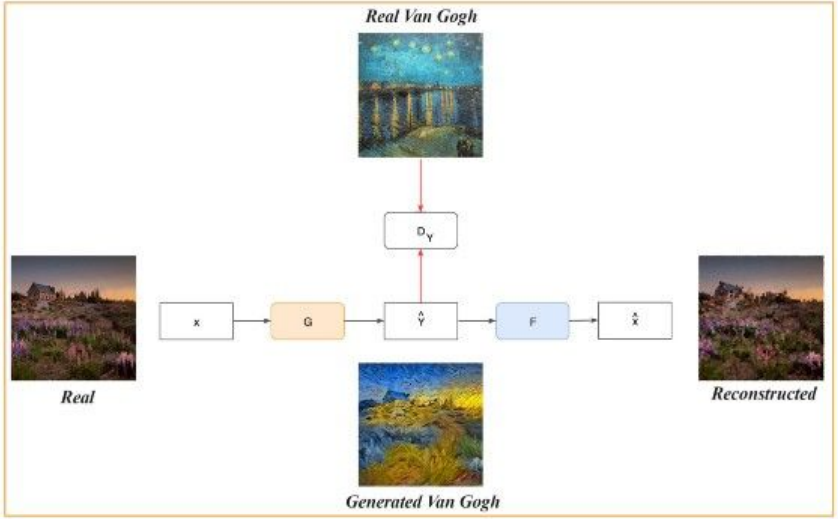
Generative Adversarial Text to Image Synthesis. 2016

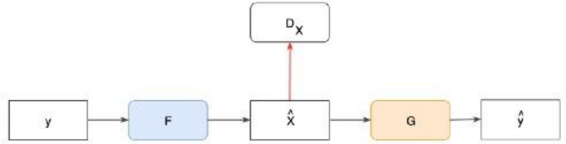
Autres types d’applications : CycleGan

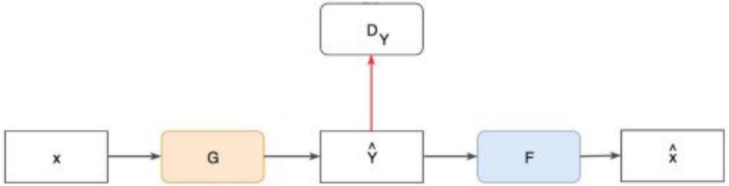


Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. 2017
Jun-Yan Zhu, Taesung Park, Phillip Isola, Alexei A. Efros

Autres types d’applications : CycleGan





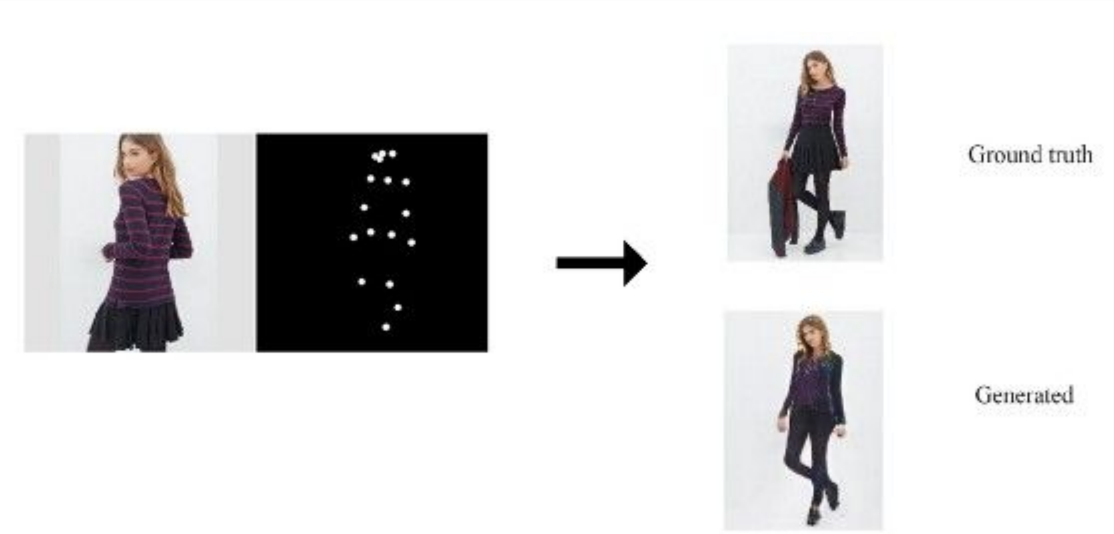


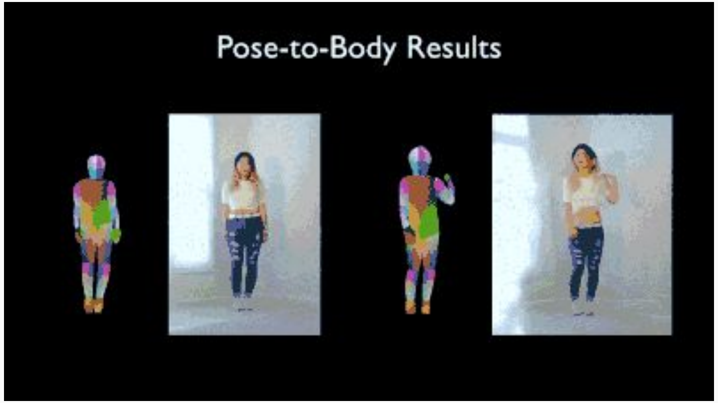
$$\begin{aligned}\mathcal{L}(G, F, D_X, D_Y) =& \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) \\ &+ \mathcal{L}_{\text{GAN}}(F, D_X, Y, X) \\ &+ \lambda\mathcal{L}_{\text{cyc}}(G, F),\end{aligned}$$

Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. 2017

Jun-Yan Zhu, Taesung Park, Phillip Isola, Alexei A. Efros

Autres types d’applications : Pose transfer





https://youtu.be/PCBTZh41Ris

Autres types d’applications : Résolution

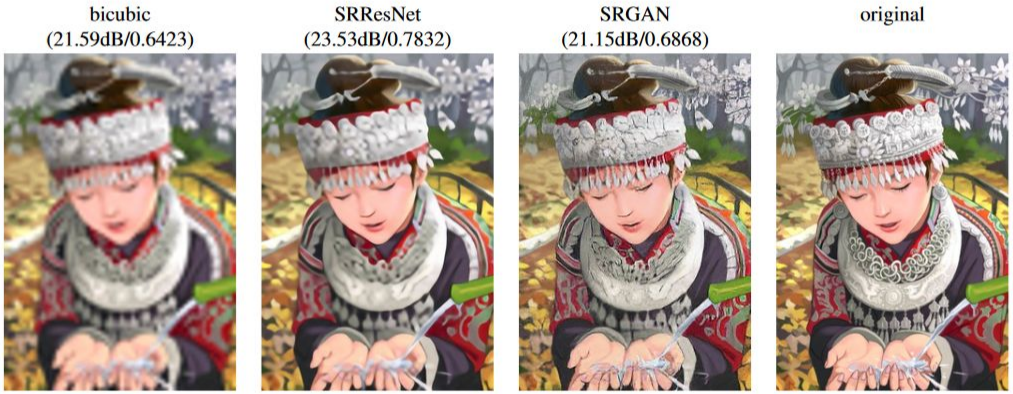


Figure 2: From left to right: bicubic interpolation, deep residual network optimized for MSE, deep residual generative adversarial network optimized for a loss more sensitive to human perception, original HR image. Corresponding PSNR and SSIM are shown in brackets. [4× upscaling]

Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network

Christian Ledig, Lucas Theis, Ferenc Husz´ar, Jose Caballero, Andrew Cunningham,Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, Wenzhe Shi

Générer des images de haute qualité : Stack / Progressive GAN

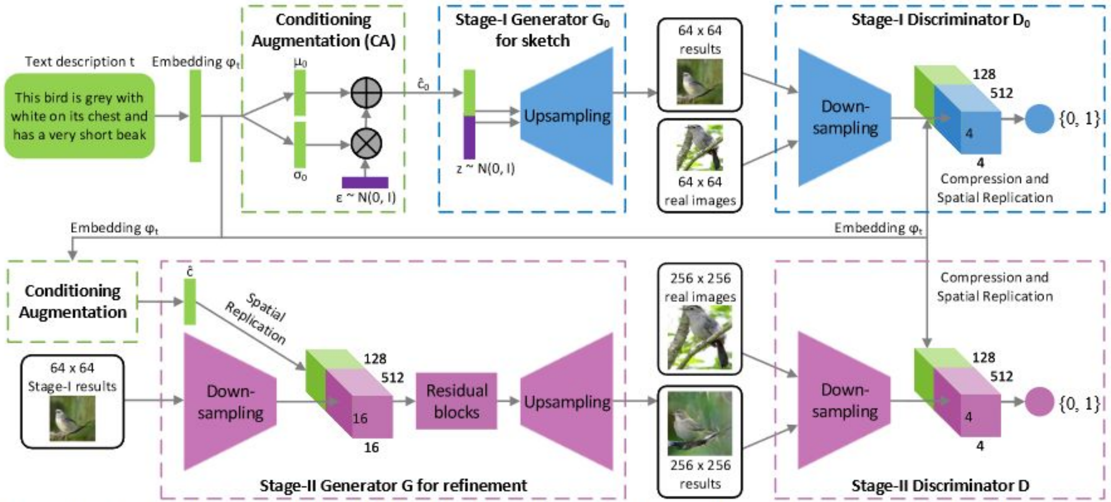
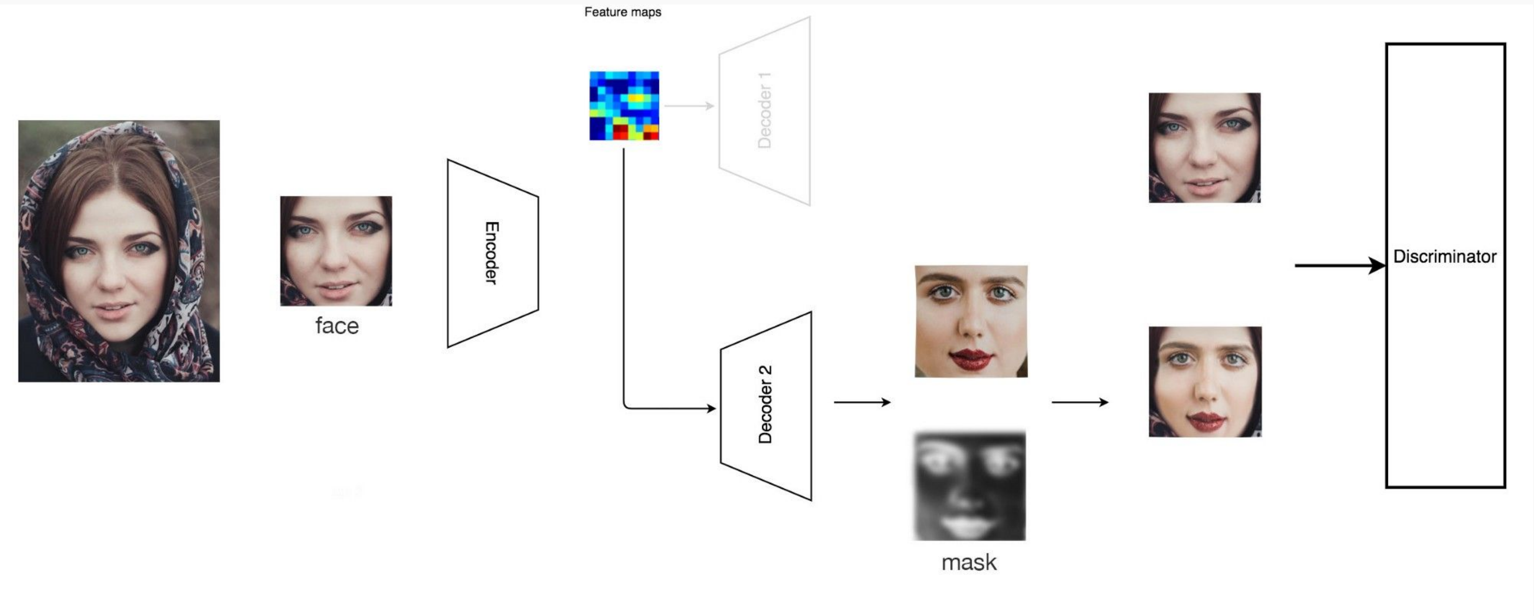


Figure 2. The architecture of the proposed StackGAN. The Stage-I generator draws a low-resolution image by sketching rough shape and basic colors of the object from the given text and painting the background from a random noise vector. Conditioned on Stage-I results, the Stage-II generator corrects defects and adds compelling details into Stage-I results, yielding a more realistic high-resolution image.

StackGAN: Text to Photo-realistic Image Synthesiswith Stacked Generative Adversarial Networks

Han Zhang, Tao Xu, Hongsheng Li,Shaoting Zhang, Xiaogang Wang3, Xiaolei Huang, Dimitris Metaxas

Autre types d'application : Deep Fake



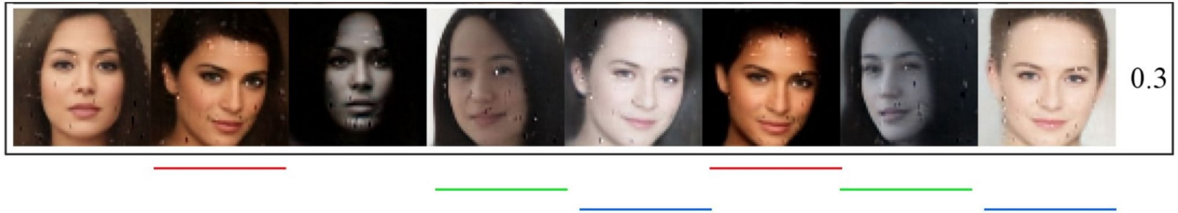
- GAN pour générer des deepfakes
- Lip Sync // Face detection en plus

Problèmes récurrents :

Gradient

$$-\nabla_{\theta_g} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \to 0 \quad \textit{change to} \quad \nabla_{\theta_g} \log\left(D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right)$$

Le mode collapse



Instabilités

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

Conclusion