

Apprentissage Automatique Auto-encodeurs & modèles génératifs

S. Herbin

stephane.herbin@onera.fr

Introduction

Rappel des cours précédents

- ▶ Algorithmes de classification/régression : $y = F(x, W)$
- ▶ Trouver la bonne représentation est une étape essentielle des chaînes d'interprétation de données !
 - ▶ x représente des données après extraction de caractéristiques : Arbres, modèles Bayésien, SVM...
 - ▶ x est la donnée brute : le Deep Learning permet de construire les caractéristiques.

Cours d'aujourd'hui

- ▶ Construire des représentations en résolvant des tâches
- ▶ Construire des représentations en apprenant à générer des données.
- ▶ Profiter du DL pour construire/naviguer dans des espaces de représentation performants (discriminants & invariants)

Auto-encodeurs

Apprentissage non supervisé

- ▶ On a vu (cours précédent) :
 - ▶ regroupement (kmeans, DBscan)
 - ▶ réduction de dimension (ACP)
 - ▶ construction de caractéristiques (auto-supervision)
- ▶ Aujourd'hui
 - ▶ Encodeur/décodeur : un schéma de conception générique.

L'ACP revisitée par les ANN I

Formulation de l'ACP comme minimisation quadratique

- ▶ Soient W_1 où W_2 deux matrices $p \times n$, $p < n$ et $\{x_i\}_{i=1}^K$ un ensemble de données.
- ▶ On cherche à minimiser le critère d'auto-association par rapport à W_1 et W_2 :

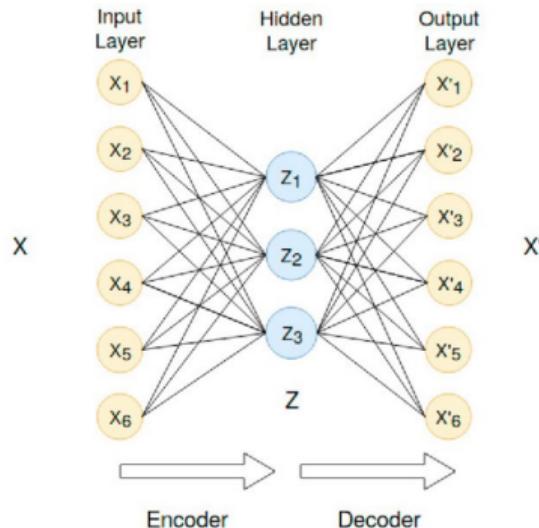
$$\sum_{i=1}^K \|x_i - W_2^T W_1 x_i\|^2 \quad (1)$$

- ▶ On peut montrer que les lignes de la matrice W_1 optimale contiennent les vecteurs propres de la matrice de covariance et que $W_1 = W_2$ à une matrice de taille $p \times p$ inversible près [6].
- ▶ La matrice W_1 contient les directions principales.

L'ACP revisitée par les ANN II

Auto-association et encodage

- ▶ Le produit $W_2^T W_1$ peut être considéré comme un réseau à deux couches « fully connected ».
- ▶ Minimiser l'équation (1) permet d'**encoder** les données dans $z = W_1 x$.
- ▶ La minimisation peut se faire par descente de gradient.
- ▶ La reconstruction ou **décodage** de z est $x' = W_2^T z$

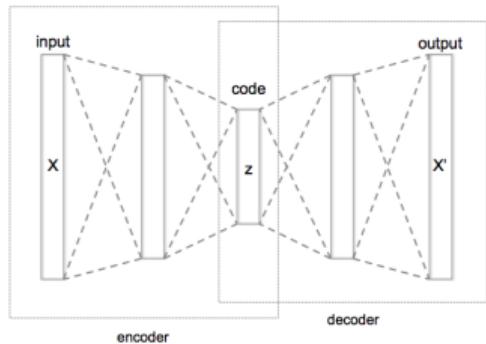


Architecture Encodeur-Décodeur

Généralisation

- ▶ $z = W_1x$ devient $z = F(x, W_1)$
- ▶ $x' = W_2^T z$ devient $x' = G(z, W_2)$

où F et G sont des réseaux de neurones de poids W_1 et W_2 , et où x' n'est pas nécessairement une reconstruction de x .



Rôle du code z

- ▶ Réduit la dimension des données → économie
- ▶ Extrait et représente leur information utile → meilleure exploitation

Apprentissage des AE I

Approche empirique

Minimisation par descente de gradient d'un critère du type :

$$\sum_{i=1}^K l(y_i, G(F(x_i, W_1), W_2)) + \text{régularisation} \quad (2)$$

Le rôle d'une régularisation est de contraindre la forme du code.

Elle peut prendre plusieurs formes :

- ▶ Parcimonie : de type $L_1 (|z_i|_1)$ [3] ou informationnelle ($KL[\hat{z}_j \parallel \epsilon]$) où \hat{z}_j est la moyenne des activations du code j et ϵ une petite probabilité [22]
- ▶ Contraction : $\|\nabla_x z\|^2$ pour améliorer la robustesse aux perturbations d'entrées
- ▶ Effet sur d'autres tâches ou objectifs ($\sum_{i=1}^K r(t_i, Q(z_i))$) pour améliorer l'extraction de l'information utile [21].

Apprentissage des AE II

Formulation variationnelle (Bayésienne)

Idée : considérer le code et la prédiction comme des variables aléatoires \Rightarrow les critères s'appliquent dans les espaces de distributions

$$z \sim q(z|x, W_1)$$

$$x \sim p(x|z, W_2)$$

On cherche une approximation du maximum de vraisemblance régularisé par un prior sur l'espace latent.

Apprentissage des AE III

On peut montrer que la loss définie pour chaque exemple x_i :

$$l_i(x_i|W_1, W_2) = \underbrace{-E_{z \sim q(z|x, W_1)}[\log p(x_i|z, W_2)]}_{\text{attache aux données}} + \underbrace{KL[q(z|x_i, W_1)\|p(z)]}_{\text{régularisation du code } z}$$

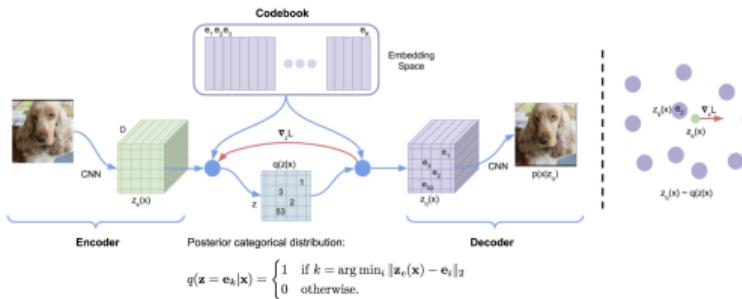
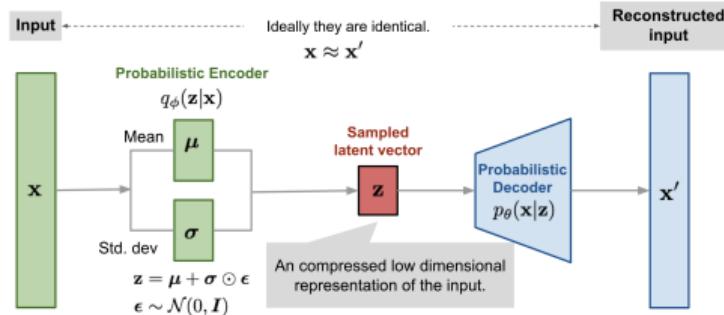
où la divergence de Kullback-Leibler est définie par

$$KL[q(z|x_i, W_1)\|p(z)] = \int q(z|x_i, W_1) \log \frac{q(z|x_i, W_1)}{p(z)} dz$$

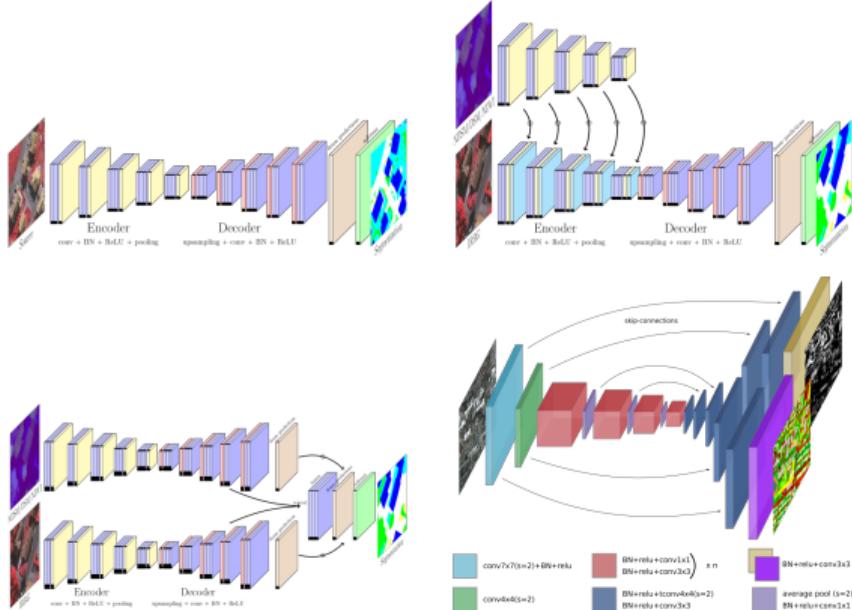
et où $p(z)$ est un prior sur le code implémenté une bonne approximation de cet objectif [20, 11].

Apprentissage des AE IV

L'intérêt d'une approche variationnelle est de pouvoir fixer globalement la forme de la distribution des codes z [34].



Généralisation du schéma encodeur/décodeur

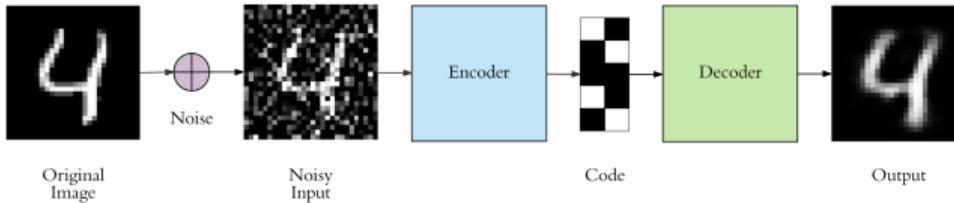


Segmentation [28, 5], estimation de profondeur [8], fusion [4] : même inspiration d'architecture « en sablier », mais pas d'exploitation du code.

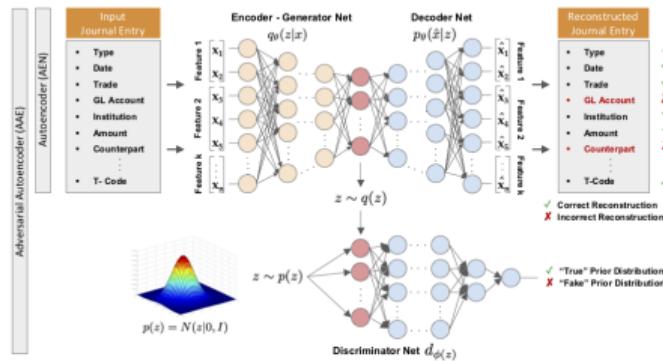
DL = « Mecano » de couches, connexions et fonctions de perte + optimisation générique (Gradient stochastique)
Auto-encodeurs et modèles génératifs

Quelques applications des auto-encodeurs I

► Débruiter [36]

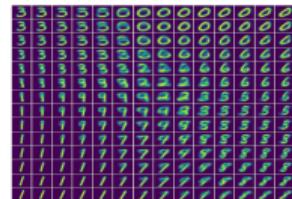


► Déetecter des anomalies [30]

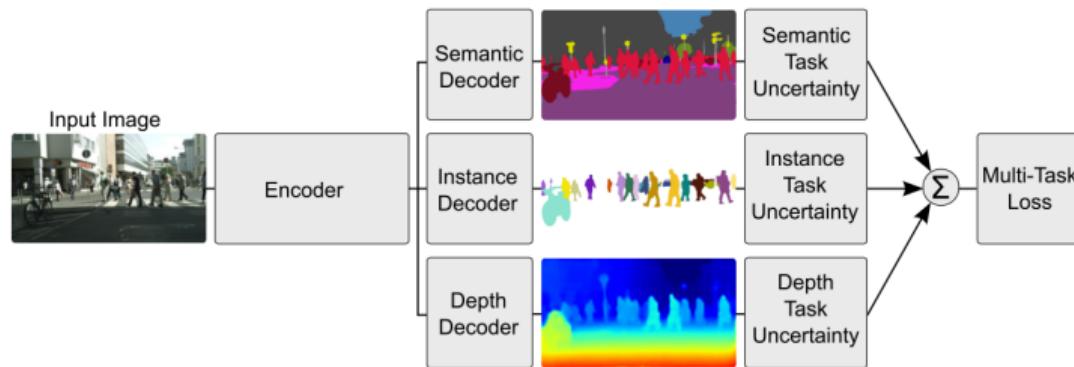


Quelques applications des auto-encodeurs II

- ▶ Interpréter/visualiser l'espace latent



- ▶ Encoder de manière générique (multi-tâche, pré-entraînement)



- ▶ Auto-supervision et tâche prétexte (cf. cours précédent)

Points clés des auto-encodeurs

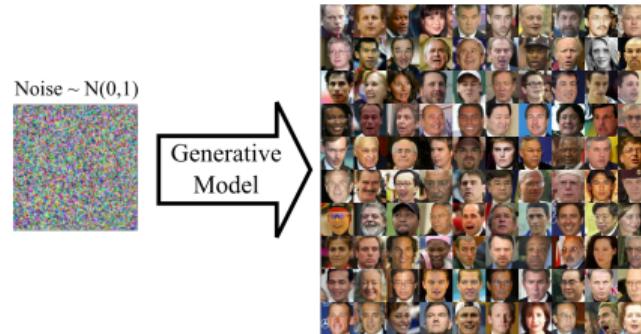
- + Non supervisé
- + Réduction de dimension
- + Construction d'un espace de représentation contrôlable/interprétable
- + Interprétation variationnelle = probabiliste
- + Flexibilité du schéma encodeur-décodeur

Utilisations

- ▶ Débruitage
- ▶ Détection d'anomalies
- ▶ Génération de données
- ▶ Pré-entraînement
- ▶ Encodage « universel »

Modèles génératifs

Produire des données vs. modéliser leur distribution



Problème

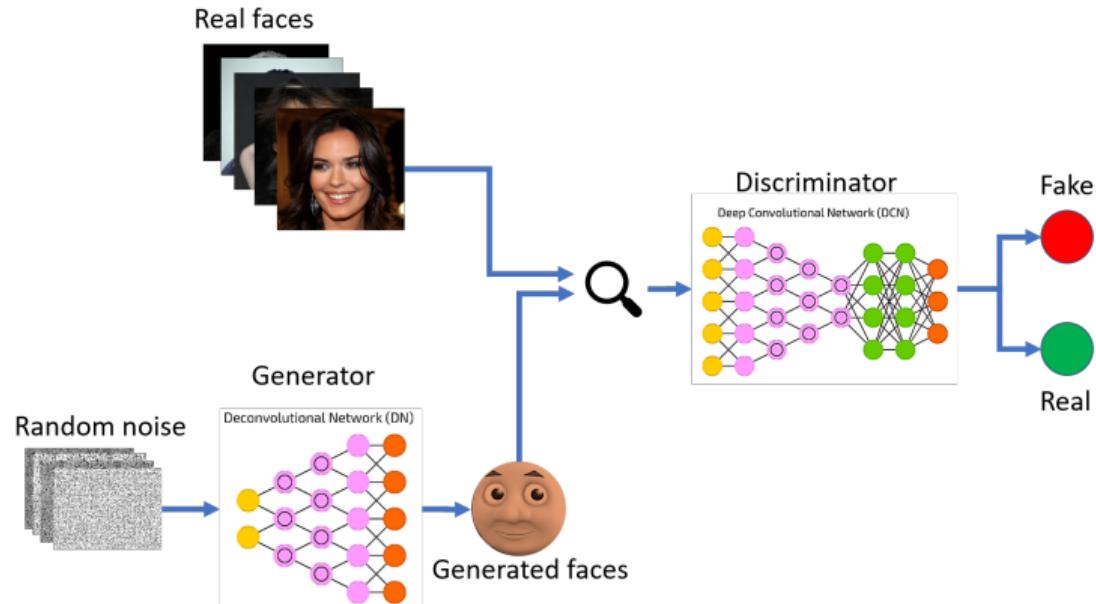
- ▶ Comment échantillonner une distribution sans la modéliser ?
- ▶ On s'intéresse à la partie décodeur (Générateur) :

$$z \sim p(z)$$

$$x \sim p(x|z, W)$$

- ▶ Peut-on apprendre directement un tel générateur/décodeur ?

Formulation adversaire I



Formulation adverse II

Principe

- ▶ Première idée : comparer les distributions générées et les données d'apprentissage (formulation variationnelle)
- ▶ Deuxième idée : apprendre comment bien comparer deux distributions (« Discriminateur »)
- ▶ Approche adverse : considérer le générateur comme un joueur essayant de flouer un détecteur d'erreur adapté
 - ⇒ principe min max

Formulation adversaire III

Critère

On cherche les poids du générateur W_G et du discriminateur W_D qui vérifient le critère (maximum de vraisemblance) :

$$\min_{W_G} \max_{W_D} \mathbb{E}_{x \sim p_{\text{data}}(x)} \left[\underbrace{\log D(x, W_D)}_{\text{discriminateur sur données réelles}} \right] + \mathbb{E}_{z \sim p(z)} \left[\underbrace{\log(1 - D(G(z, W_G), W_D))}_{\text{discriminateur sur données simulées}} \right]$$

Apprentissage des GAN I

Deux boucles imbriquées [12].

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 $\{\mathbf{z}^{(1)}, \dots, \mathbf{z}^{(m)}\}$ from noise prior $p_g(\mathbf{z})$.
- Sample minibatch of m examples $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\mathbf{x})$.
- Update the discriminator by ascending its stochastic gradient:

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

end for

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

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

end for

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

Apprentissage des GAN II



- ▶ Apprentissage difficile à contrôler [29, 1] : gradients évanescents ou divergents, disparition de modes, instabilités...
- ▶ Passage à l'échelle difficile.
- ▶ Impact des architectures neuronales (générateur et discriminateur)
- ▶ Mais améliorations constantes : Wasserstein GAN [2, 13], Boundary Equilibrium GAN [7], Least Squares GAN [24], Normalisation spectrale [25], Apprentissage progressif [15], ...

Wasserstein GAN [2, 13]

Idée : substituer au maximum de vraisemblance comme mesure d'écart entre distributions (réelle et simulée) une distance de transport (*Earth-Mover* ou *Wasserstein-1*) [35] :

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \Pi(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(x,y) \sim \gamma} [| | x - y | |]$$

On peut montrer (dualité de Kantorovich-Rubinstein) :

$$W(\mathbb{P}_r, \mathbb{P}_\theta) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x \sim \mathbb{P}_r} [f(x)] - \mathbb{E}_{x \sim \mathbb{P}_\theta} [f(x)]$$

Ce qui donne une nouvelle formulation du critère :

$$\min_{W_G} \max_{W_D} \mathbb{E}_{x \sim p_{\text{data}}(x)} [D(x, W_D)] - \mathbb{E}_{z \sim p(z)} [D(G(z, W_G), W_D)]$$

mais où il faut contrôler les variations de $D(x, W_D)$ (le $\|f\|_L \leq 1$).

Wasserstein GAN

Les fonctions de perte comparent directement les sorties (pas les log proba) et introduisent un terme de régularisation sur le gradient [13].

Algorithm 1 WGAN with gradient penalty. We use default values of $\lambda = 10$, $n_{\text{critic}} = 5$, $\alpha = 0.0001$, $\beta_1 = 0$, $\beta_2 = 0.9$.

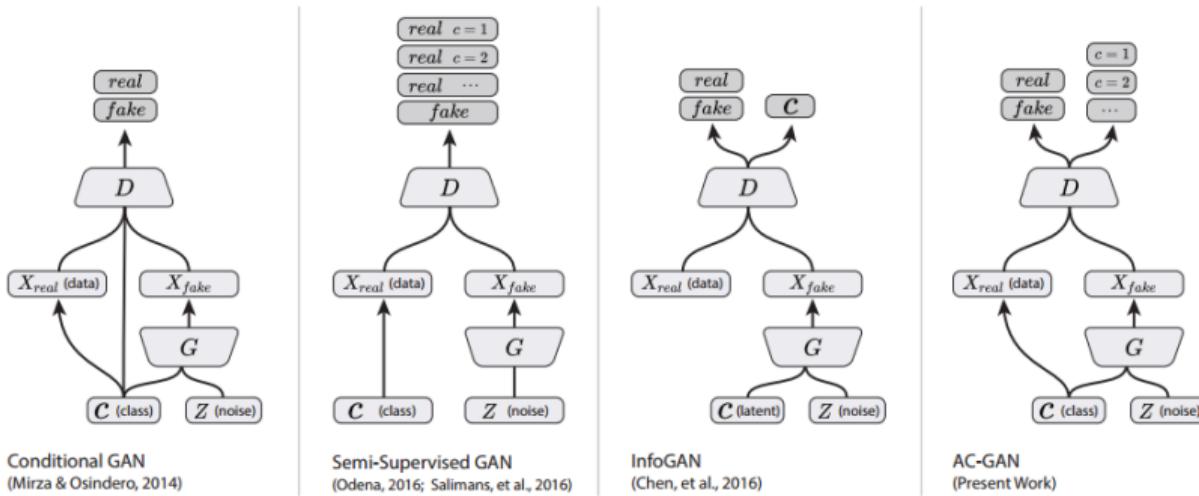
Require: The gradient penalty coefficient λ , the number of critic iterations per generator iteration n_{critic} , the batch size m , Adam hyperparameters α, β_1, β_2 .

Require: initial critic parameters w_0 , initial generator parameters θ_0 .

```
1: while  $\theta$  has not converged do
2:   for  $t = 1, \dots, n_{\text{critic}}$  do
3:     for  $i = 1, \dots, m$  do
4:       Sample real data  $\mathbf{x} \sim \mathbb{P}_r$ , latent variable  $\mathbf{z} \sim p(\mathbf{z})$ , a random number  $\epsilon \sim U[0, 1]$ .
5:        $\tilde{\mathbf{x}} \leftarrow G_\theta(\mathbf{z})$ 
6:        $\hat{\mathbf{x}} \leftarrow \epsilon \mathbf{x} + (1 - \epsilon) \tilde{\mathbf{x}}$ 
7:        $L^{(i)} \leftarrow D_w(\tilde{\mathbf{x}}) - D_w(\mathbf{x}) + \lambda(\|\nabla_{\hat{\mathbf{x}}} D_w(\hat{\mathbf{x}})\|_2 - 1)^2$ 
8:     end for
9:      $w \leftarrow \text{Adam}(\nabla_w \frac{1}{m} \sum_{i=1}^m L^{(i)}, w, \alpha, \beta_1, \beta_2)$ 
10:   end for
11:   Sample a batch of latent variables  $\{\mathbf{z}^{(i)}\}_{i=1}^m \sim p(\mathbf{z})$ .
12:    $\theta \leftarrow \text{Adam}(\nabla_\theta \frac{1}{m} \sum_{i=1}^m -D_w(G_\theta(\mathbf{z})), \theta, \alpha, \beta_1, \beta_2)$ 
13: end while
```

Variations autour des architectures I

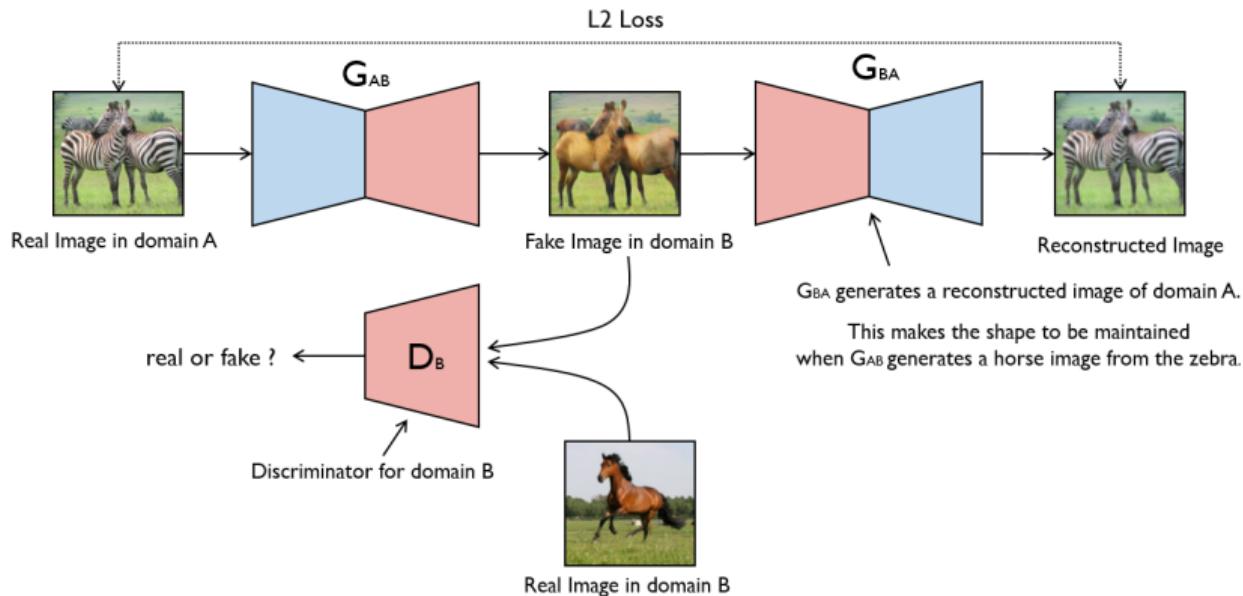
Conditionner la génération (par une classe, une image...)



D'après [26]

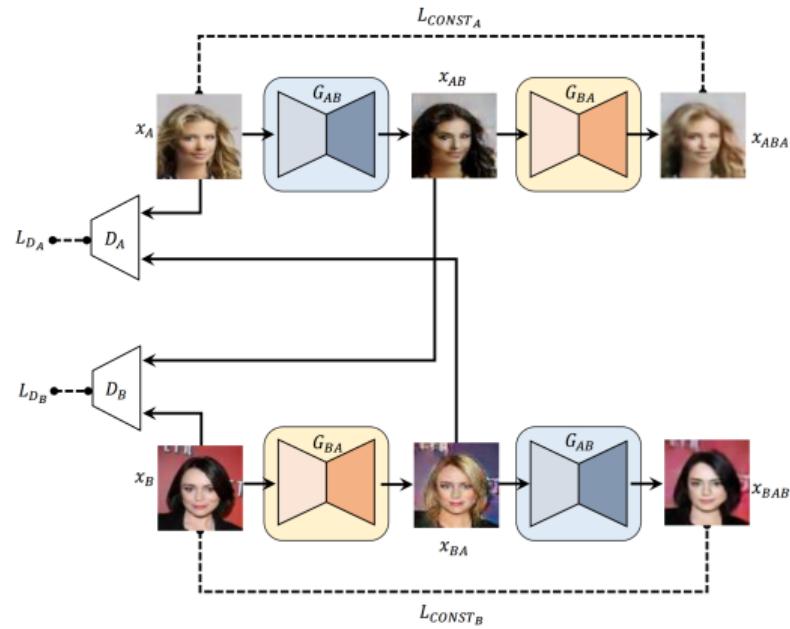
Variations autour des architectures II

Introduire une contrainte de reconstruction (cf. auto-encodeur) : Cycle-GAN [40]



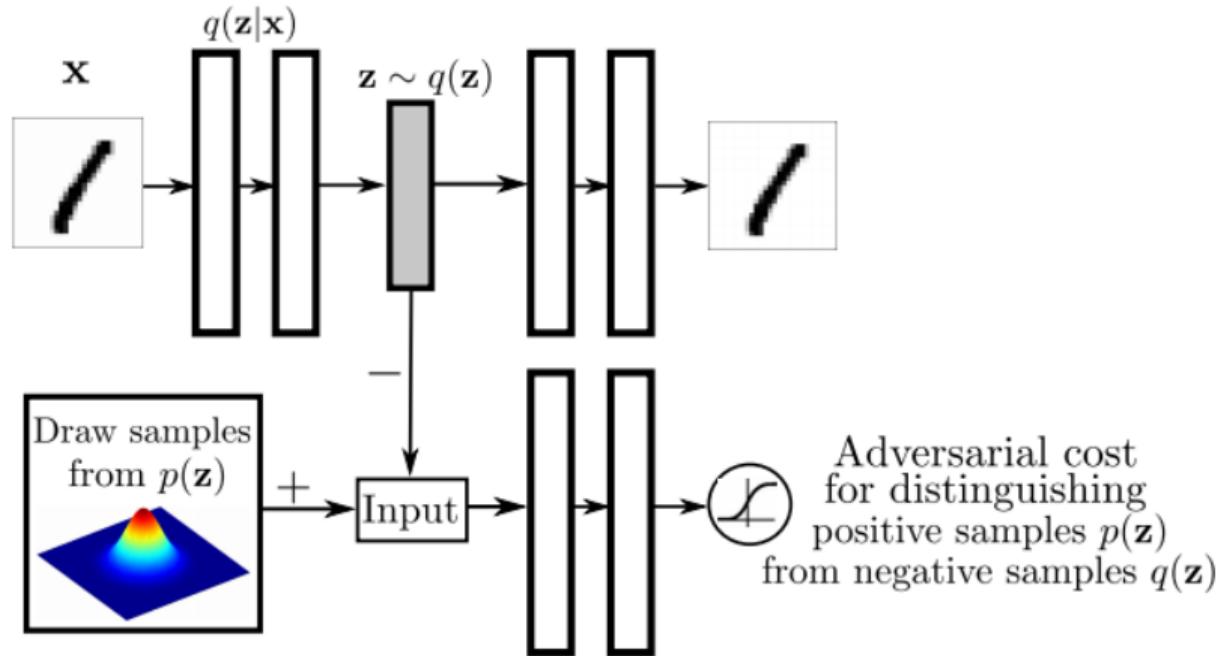
Variations autour des architectures III

Partager des espaces de représentation entre plusieurs populations de données :
DiscoGan [19]



Variations autour des architectures IV

Construire l'espace latent d'échantillonnage : Adversarial auto-encoder [23]



Modèles de diffusion I

Une autre stratégie de génération (qui semble détrôner les GAN...) [39]

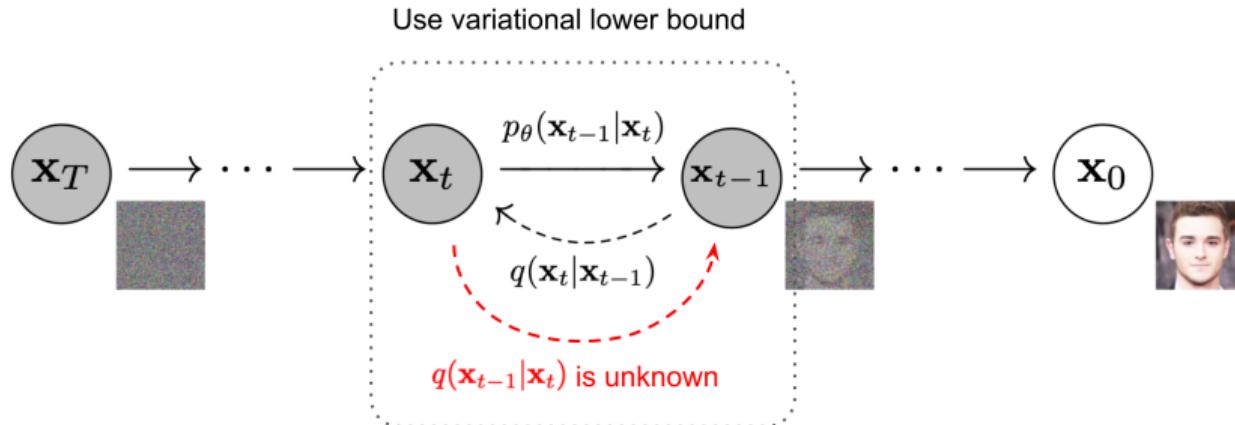
Principe

1. Dégrader séquentiellement et de manière connue une donnée source pour la transformer en un bruit = diffusion
2. Générer une nouvelle donnée en inversant la dégradation (reconstruction, débruitage) = c'est ce qu'il faut apprendre !

Deux problèmes

- ▶ Comment bien dégrader ?
- ▶ Comment reconstruire pour cette dégradation ?

Denoising Diffusion Probabilistic Models [14]



Modèles de diffusion III

Dégradation

- ▶ On part d'une image x_0 à laquelle on ajoute un bruit gaussien de variance connue β_t à chaque étape t pour générer une séquence d'images x_t .

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1 - \beta_t}x_{t-1}, \beta_t I)$$

Reconstruction

- ▶ On reconstruit en *inversant* la chaîne de Markov $q(x_t|x_{t-1})$ par une autre chaîne de paramètres inconnus :

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$$

C'est ce qu'il faut apprendre

Modèles de diffusion IV

Apprentissage

- ▶ On maximise l'entropie croisée : $\mathbb{E}_{q(x_0)}[\log p_\theta(x_0)]$ où $q(x_0)$ est la distribution source à générer et $p_\theta(x_0)$ est le modèle.
- ▶ On utilise une approximation variationnelle (comme pour les VAE) :

$$L_{\text{VLB}} = \mathbb{E}_{q(x_{0:T})} \left[\log \frac{q(x_{1:T}|x_0)}{p_\theta(x_{0:T})} \right] \geq -\mathbb{E}_{q(x_0)}[\log p_\theta(x_0)]$$

- ▶ Ce qui donne comme critère à minimiser

$$L_{\text{VLB}} = \text{Cste} + \sum_{t=2}^T \underbrace{D_{\text{KL}}(q(x_{t-1}|x_t, x_0) \parallel p_\theta(x_{t-1}|x_t))}_{L_{t-1}} - \underbrace{\log p_\theta(x_0|x_1)}_{L_0}$$

Toutes les probabilités sont gaussiennes \implies expression analytique !

Algorithmes

Algorithm 1 Training

```

1: repeat
2:    $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ 
3:    $t \sim \text{Uniform}(\{1, \dots, T\})$ 
4:    $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
5:   Take gradient descent step on
        $\nabla_{\theta} \|\epsilon - \epsilon_{\theta}(\sqrt{\alpha_t} \mathbf{x}_0 + \sqrt{1 - \alpha_t} \epsilon, t)\|^2$ 
6: until converged

```

Algorithm 2 Sampling

```

1:  $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
2: for  $t = T, \dots, 1$  do
3:    $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$  if  $t > 1$ , else  $\mathbf{z} = \mathbf{0}$ 
4:    $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \alpha_t}} \epsilon_{\theta}(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$ 
5: end for
6: return  $\mathbf{x}_0$ 

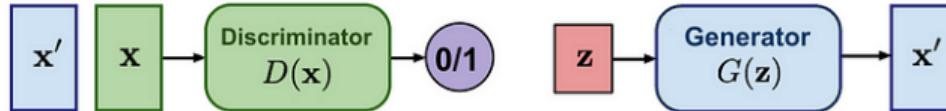
```

avec la reparamétrisation $\alpha_t = 1 - \beta_t$ et $\bar{\alpha}_t = \prod_{i=1}^t \alpha_i$

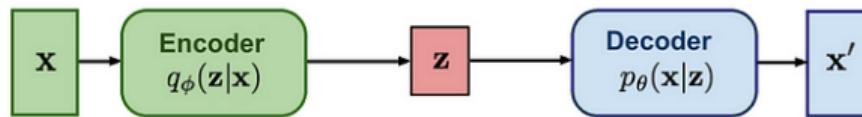
- ▶ Remarque : l'apprentissage et la prédiction sont stochastiques \Rightarrow temps de calcul important.
- ▶ Des stratégies d'amélioration des temps de calcul : rendre l'inversion déterministe [31], améliorer les stratégies d'échantillonnage [16], générer dans l'espace latent d'un décodeur [27]...

GAN vs. VAE vs. Diffusion I

GAN: Adversarial training

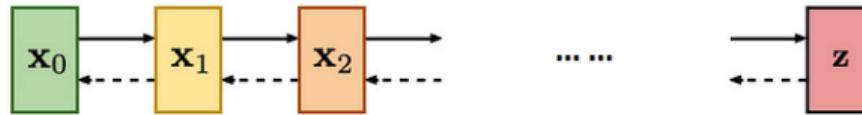


VAE: maximize variational lower bound



Diffusion models:

Gradually add Gaussian noise and then reverse



GAN vs. VAE vs. Diffusion II

GAN

- ▶ Qualité de rendu
- ▶ Manque de diversité (« Mode collapse »)
- ▶ Calcul rapide
- ▶ Apprentissage délicat
- ▶ Espace latent accessible

VAE

- ▶ Rendu de faible qualité (images floues)
- ▶ Grande diversité
- ▶ Calcul rapide
- ▶ Facile à apprendre
- ▶ Espace latent accessible

Diffusion models

- ▶ Qualité de rendu
- ▶ Grande diversité
- ▶ Calcul coûteux
- ▶ Apprentissage coûteux mais sûr
- ▶ Espace latent complexe (toute la séquence de débruitage)

Mais des convergences...

- ▶ *Diffusion Models Beat GANs on Image Synthesis* [10]
- ▶ *Tackling the Generative Learning Trilemma with Denoising Diffusion GANs* [38]
- ▶ *Diffusion-GAN : Training GANs with Diffusion* [37]

Quelques applications

- ▶ Génération de visage photo-réalistes
 - [DiscoFaceGAN \[9\]](#)
 - [Style GAN \[17, 18\]](#)
- ▶ Transfert de style
 - [Pix2Pix](#)
- ▶ Manipulation d'image (Inpainting, super résolution, rendu photo [33]...)
 - [Compétition AIM](#), [Compte-rendu Inpainting](#), [Proceedings Image relighting \[32\]](#)
- ▶ Deep Fake
 - [DeepFaceLab](#)
- ▶ Génération conditionnelle
 - texte -> image : [DALL-E](#), [DALL-E3](#), [Stable diffusion](#), [RPG...](#)
 - texte -> texte : [chatGPT](#)

Modèles génératifs

Points clés des GAN

- + Génération de données « réalistes »
- + Flexibilité fonctionnelle
- Apprentissage délicat et coûteux
- Beaucoup de travaux
- Critères de validation ?

Utilisations

- ▶ Deux familles principales : GAN et modèles de diffusion
- ▶ Génération de données multimédia : images, vidéo, texte, son...
- ▶ Filtrage : super-résolution, transfert de style, rendu

Références |

- [1] Martin Arjovsky and Léon Bottou.
Towards principled methods for training generative adversarial networks.
arXiv preprint arXiv :1701.04862, 2017.
- [2] Martin Arjovsky, Soumith Chintala, and Léon Bottou.
Wasserstein gan.
In *International Conference on Machine Learning*, 2017.
- [3] Devansh Arpit, Yingbo Zhou, Hung Ngo, and Venu Govindaraju.
Why regularized auto-encoders learn sparse representation ?
In *International conference on machine learning*, pages 136–144. PMLR, 2016.
- [4] Nicolas Audebert, Bertrand Le Saux, and Sébastien Lefèvre.
Beyond RGB : Very high resolution urban remote sensing with multimodal deep networks.
ISPRS Journal of Photogrammetry and Remote Sensing, 140 :20–32, 2018.
Publisher : Elsevier.
- [5] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla.
Segnet : A deep convolutional encoder-decoder architecture for image segmentation.
IEEE transactions on pattern analysis and machine intelligence, 39(12) :2481–2495, 2017.
Publisher : IEEE.
- [6] Pierre Baldi and Kurt Hornik.
Neural networks and principal component analysis : Learning from examples without local minima.
Neural networks, 2(1) :53–58, 1989.
Publisher : Elsevier.
- [7] David Berthelot, Thomas Schumm, and Luke Metz.
Began : Boundary equilibrium generative adversarial networks.
arXiv preprint arXiv :1703.10717, 2017.

Références II

- [8] Marcela Carvalho, Bertrand Le Saux, Pauline Trouve-Peloux, Frederic Champagnat, and Andres Almansa.
Multitask learning of height and semantics from aerial images.
IEEE Geoscience and Remote Sensing Letters, 17(8) :1391–1395, 2020.
- [9] Yu Deng, Jiaolong Yang, Dong Chen, Fang Wen, and Xin Tong.
Disentangled and controllable face image generation via 3d imitative-contrastive learning.
In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5154–5163, 2020.
- [10] Prafulla Dhariwal and Alexander Nichol.
Diffusion Models Beat GANs on Image Synthesis.
In *Advances in Neural Information Processing Systems*, volume 34, pages 8780–8794. Curran Associates, Inc., 2021.
- [11] Carl Doersch.
Tutorial on variational autoencoders.
arXiv preprint arXiv :1606.05908, 2016.
- [12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio.
Generative adversarial nets.
In *Advances in neural information processing systems*, pages 2672–2680, 2014.
- [13] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville.
Improved training of wasserstein GANs.
In *Proceedings of the 31st international conference on neural information processing systems*, pages 5769–5779, 2017.
- [14] Jonathan Ho, Ajay Jain, and Pieter Abbeel.
Denoising Diffusion Probabilistic Models.
In *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020.

Références III

- [15] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen.
Progressive growing of GANs for improved quality, stability, and variation.
In International conference on learning representations, 2018.
- [16] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine.
Elucidating the Design Space of Diffusion-Based Generative Models.
Advances in Neural Information Processing Systems, 35 :26565–26577, December 2022.
- [17] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila.
Training generative adversarial networks with limited data.
arXiv preprint arXiv :2006.06676, 2020.
- [18] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila.
Analyzing and Improving the Image Quality of StyleGAN.
pages 8110–8119, 2020.
- [19] Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jung Kwon Lee, and Jiwon Kim.
Learning to discover cross-domain relations with generative adversarial networks.
In International conference on machine learning, pages 1857–1865. PMLR, 2017.
- [20] Diederik P Kingma and Max Welling.
Auto-encoding variational bayes.
arXiv preprint arXiv :1312.6114, 2013.
- [21] Lei Le, Andrew Patterson, and Martha White.
Supervised autoencoders : Improving generalization performance with unsupervised regularizers.
Advances in neural information processing systems, 31 :107–117, 2018.
- [22] Alireza Makhzani and Brendan Frey.
K-sparse autoencoders.
arXiv preprint arXiv :1312.5663, 2013.

Références IV

- [23] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey.
Adversarial autoencoders.
arXiv preprint arXiv :1511.05644, 2015.
- [24] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley.
Least squares generative adversarial networks.
In *Proceedings of the IEEE international conference on computer vision*, pages 2794–2802, 2017.
- [25] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida.
Spectral normalization for generative adversarial networks.
In *International conference on learning representations*, 2018.
- [26] Augustus Odena, Christopher Olah, and Jonathon Shlens.
Conditional image synthesis with auxiliary classifier GANs.
In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th international conference on machine learning*, volume 70 of *Proceedings of machine learning research*, pages 2642–2651, International Convention Centre, Sydney, Australia, August 2017. PMLR.
- [27] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer.
High-Resolution Image Synthesis With Latent Diffusion Models.
pages 10684–10695, 2022.
- [28] Olaf Ronneberger, Philipp Fischer, and Thomas Brox.
U-net : Convolutional networks for biomedical image segmentation.
In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.
- [29] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen.
Improved techniques for training GANs.
In *Proceedings of the 30th international conference on neural information processing systems*, pages 2234–2242, 2016.

Références V

- [30] Marco Schreyer, Timur Sattarov, Christian Schulze, Bernd Reimer, and Damian Borth.
Detection of accounting anomalies in the latent space using adversarial autoencoder neural networks.
arXiv preprint arXiv :1908.00734, 2019.
- [31] Jiaming Song, Chenlin Meng, and Stefano Ermon.
Denoising Diffusion Implicit Models.
October 2020.
- [32] Tiancheng Sun, Jonathan T Barron, Yun-Ta Tsai, Zexiang Xu, Xueming Yu, Graham Fyffe, Christoph Rhemann, Jay Busch, Paul Debevec, and Ravi Ramamoorthi.
Single image portrait relighting.
ACM Transactions on Graphics (TOG), 38(4) :1–12, 2019.
Publisher : ACM New York, NY, USA.
- [33] Ayush Tewari, Ohad Fried, Justus Thies, Vincent Sitzmann, Stephen Lombardi, Kalyan Sunkavalli, Ricardo Martin-Brualla, Tomas Simon, Jason Saragih, Matthias Nießner, and others.
State of the art on neural rendering.
In *Computer graphics forum*, volume 39, pages 701–727. Wiley Online Library, 2020.
Number : 2.
- [34] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu.
Neural discrete representation learning.
In *Proceedings of the 31st international conference on neural information processing systems*, pages 6309–6318, 2017.
- [35] Cédric Villani.
Optimal transport : old and new, volume 338.
Springer Science & Business Media, 2008.
- [36] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou.
Stacked denoising autoencoders : Learning useful representations in a deep network with a local denoising criterion.
Journal of machine learning research, 11(12), 2010.

Références VI

- [37] Zhendong Wang, Huangjie Zheng, Pengcheng He, Weizhu Chen, and Mingyuan Zhou.
Diffusion-GAN : Training GANs with Diffusion, August 2023.
arXiv :2206.02262 [cs, stat].
- [38] Zhisheng Xiao, Karsten Kreis, and Arash Vahdat.
Tackling the Generative Learning Trilemma with Denoising Diffusion GANs.
October 2021.
- [39] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang.
Diffusion Models : A Comprehensive Survey of Methods and Applications.
ACM Computing Surveys, 56(4) :105 :1–105 :39, November 2023.
- [40] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros.
Unpaired image-to-image translation using cycle-consistent adversarial networks.
In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.