

# Apprentissage Automatique Auto-encodeurs & modèles génératifs

**S. Herbin, A. Chan Hon Tong**

[stephane.herbin@onera.fr](mailto:stephane.herbin@onera.fr)

# Introduction

## Rappel des cours précédents

- ▶ Algorithmes de classification/régression :  $y = F(x, W)$
- ▶ Deux situations :
  - ▶  $x$  représente des données après extraction de caractéristiques : Arbres, modèles Bayésien, SVM...
  - ▶  $x$  est la donnée brute : Deep Learning

## Cours d'aujourd'hui

- ▶ Apprentissage non-supervisé pour construire des caractéristiques
- ▶ Représenter, coder, générer les données
- ▶ Profiter du DL pour construire/naviguer dans des espaces de représentation plus expressifs

# Auto-encodeurs et modèles génératifs : plan

Auto-encodeurs

Principes

Apprentissage

Extensions et applications

Modèles génératifs

Principes

Apprentissage

Extensions et applications

# Auto-encodeurs

# L'ACP revisitée par les RN 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  $W_2$  :

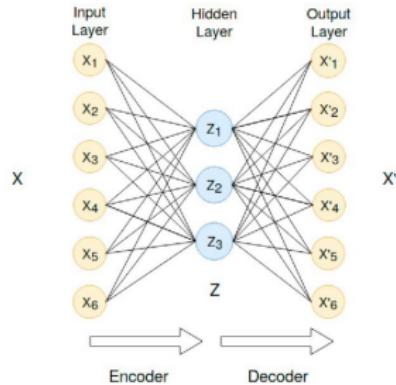
$$\sum_{i=1}^K \|x_i - W_2^T W_1 x_i\|^2 \quad (1)$$

- ▶ On peut montrer que les lignes de la matrice  $W_1$  optimale contient 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 RN 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  $x' = W_2^T z$

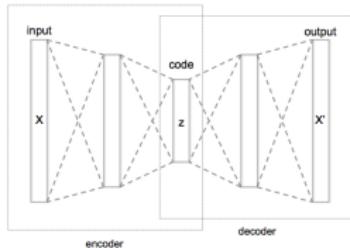


# Architecture Encodeur-Décodeur

## 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  $\rightarrow$  économie
- ▶ Extrait et représente leur information utile  $\rightarrow$  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  $\epsilon$  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  $x_i$  :

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

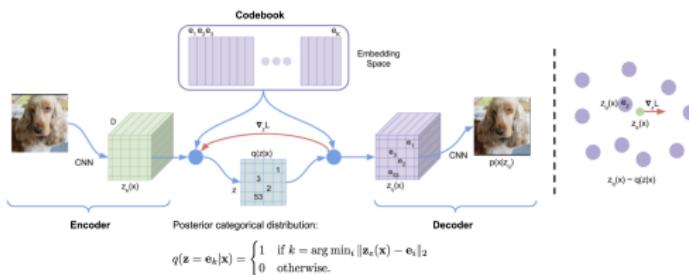
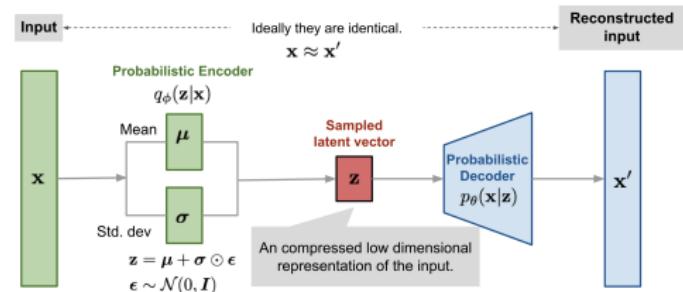
avec la divergence de Kullback-Leibler

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

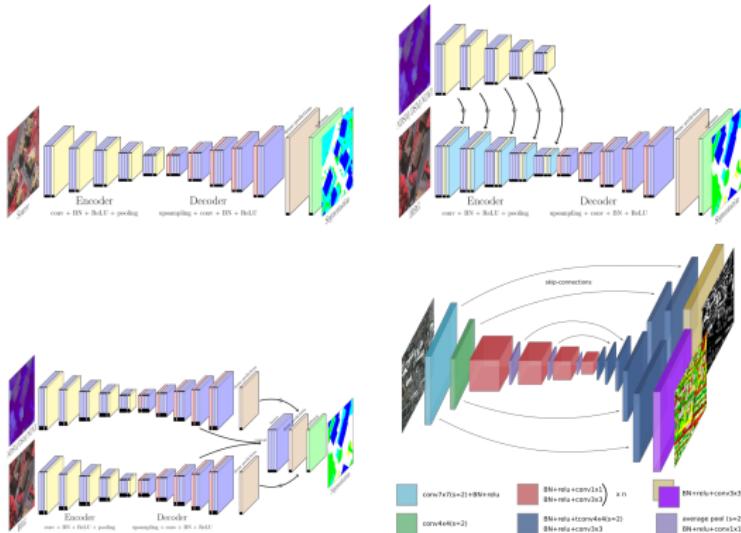
et où  $p(z)$  est un prior sur le code implémente une bonne approximation de cet objectif [20, 12].

# Apprentissage des AE IV

L'intérêt d'une approche variationnelle est de pouvoir fixer globalement la forme de la distribution des codes  $z$  [32].



# Généralisation du schéma encodeur/décodeur

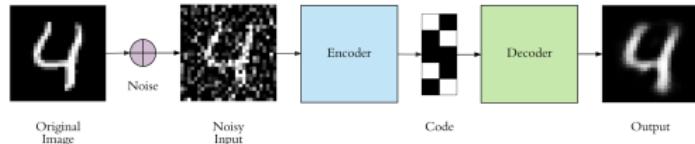


Segmentation[27, 5], estimation de profondeur[8], fusion[4] : même inspiration d'architecture « en sablier », mais pas d'exploitation du code.

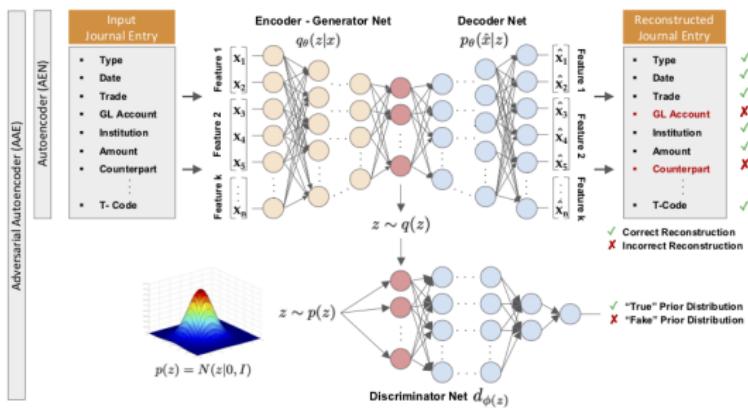
DL = « Mecano » de couches, connexions et fonctions de perte + optimisation générique (Gradient stochastique)

# Quelques applications des auto-encodeurs I

## ► Débruitier [35]

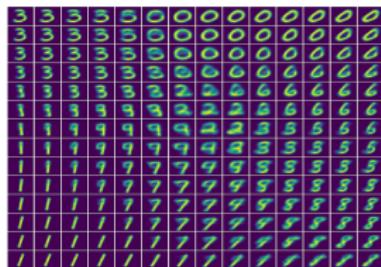


## ► Détecter des anomalies [29]

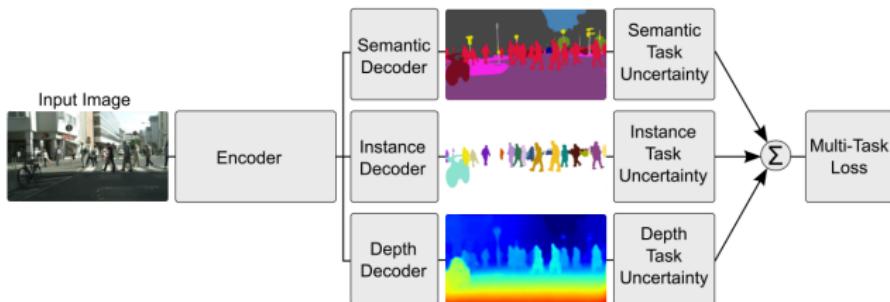


# Quelques applications des auto-encodeurs II

- ▶ Interpréter/visualiser l'espace latent

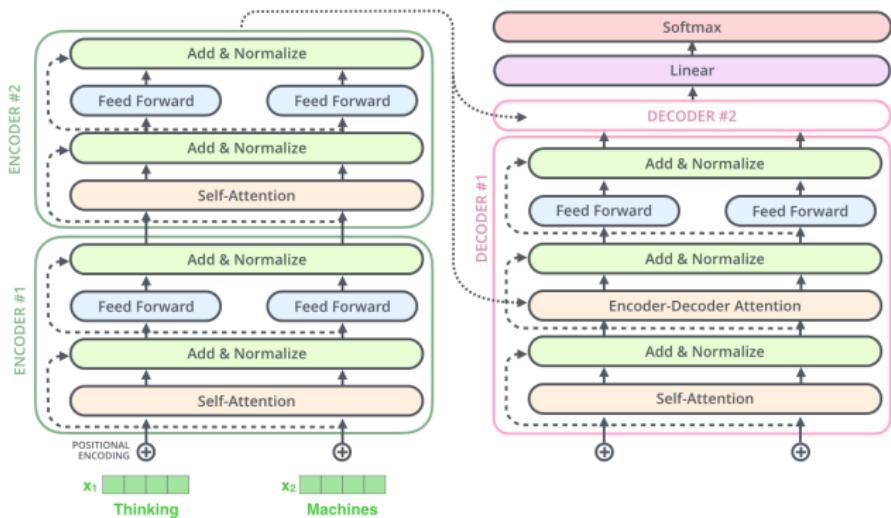


- ▶ Encoder de manière générique (multi-tâche, pré-entraînement)



# Des architectures de plus en plus complexes

Le modèle "transformer" [33] : utilisé pour la traduction de texte [11].



# Auto-encodeurs & Cie : Résumé

## Points clés des auto-encodeurs

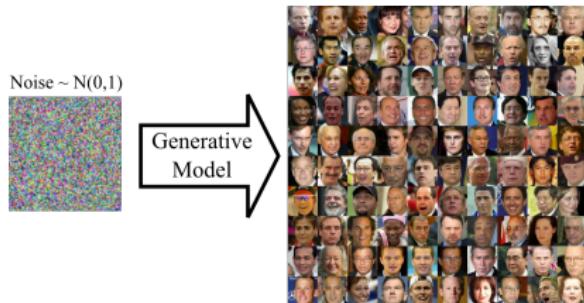
- + Non supervisé
- + Réduction de dimension
- + Construction d'un espace de représentation contrôlable/interprétable
- + Interprétation variationnelle = probabiliste
- + Flexibilité du schéma encodeur-décodeur

## Utilisations

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

# Modèles génératifs

# Produire des données vs. modéliser leur distribution



## Problème

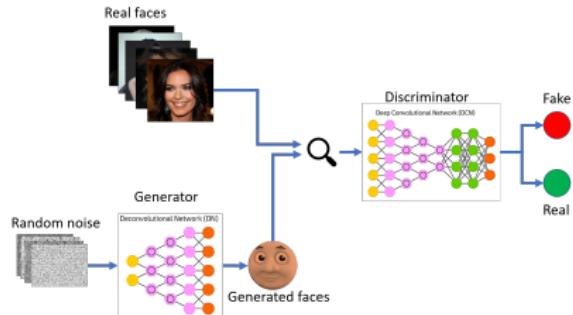
- ▶ Comment échantillonner une distribution sans la modéliser ?
- ▶ On s'intéresse à la partie décodeur (Générateur) :

$$z \sim p(z)$$

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

- ▶ Peut-on apprendre un tel générateur ?

# Formulation adversaire I



## 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 adverse II

### Critère

On cherche les poids du générateur  $W_G$  et du discriminateur  $W_D$  qui vérifient le critère :

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

# Apprentissage |

Deux boucles imbriquées s'appuyant sur des ensembles de données générées et échantillonnées [13].

---

**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:

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

**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:

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

**end for**

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

---

# Apprentissage II



- ▶ Apprentissage difficile à contrôler [28, 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, 14], Boundary Equilibrium GAN [7], Least Squares GAN [24], Normalisation spectrale [25], Apprentissage progressif [16], ...

## Apprentissage III

### Exemple d'amélioration : Wasserstein GAN [2, 14]

Idée : substituer au maximum de vraisemblance comme mesure d'écart entre distributions (réelle et simulée) une distance de transport (*Earth-Mover* ou *Wasserstein-1*) [34] :

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

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  $\|f\|_L \leq 1$ ).

# Apprentissage IV

Nouvel algorithme : les fonctions de perte comparent directement les sorties (pas les log) et introduisent un terme de régularisation sur le gradient [14].

---

**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  $\lambda$ , the number of critic iterations per generator iteration  $n_{\text{critic}}$ , the batch size  $m$ , Adam hyperparameters  $\alpha, \beta_1, \beta_2$ .

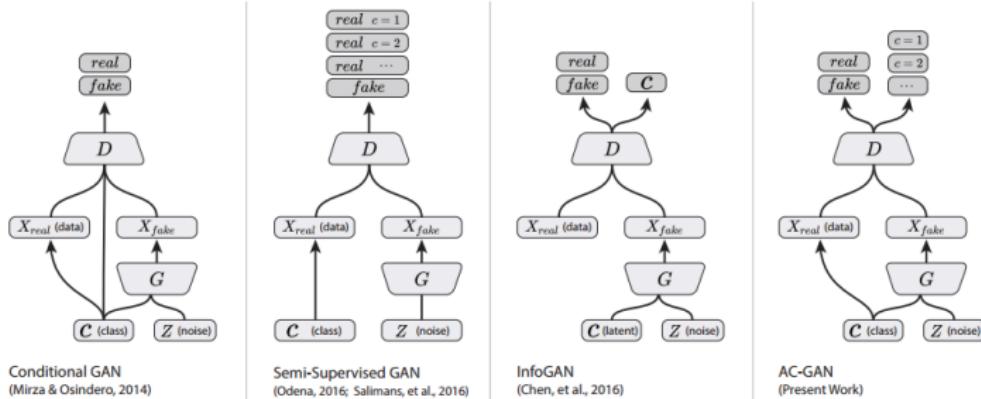
**Require:** initial critic parameters  $w_0$ , initial generator parameters  $\theta_0$ .

```
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

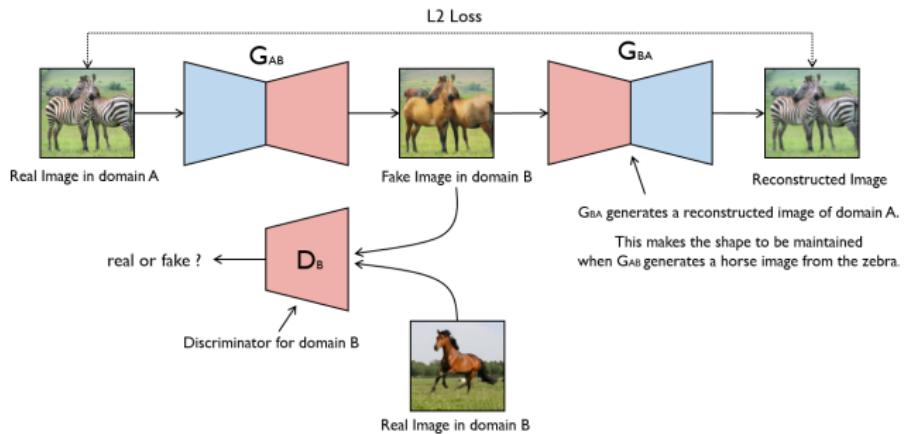
Conditionner la génération (par une classe, une image...)



D'après [26]

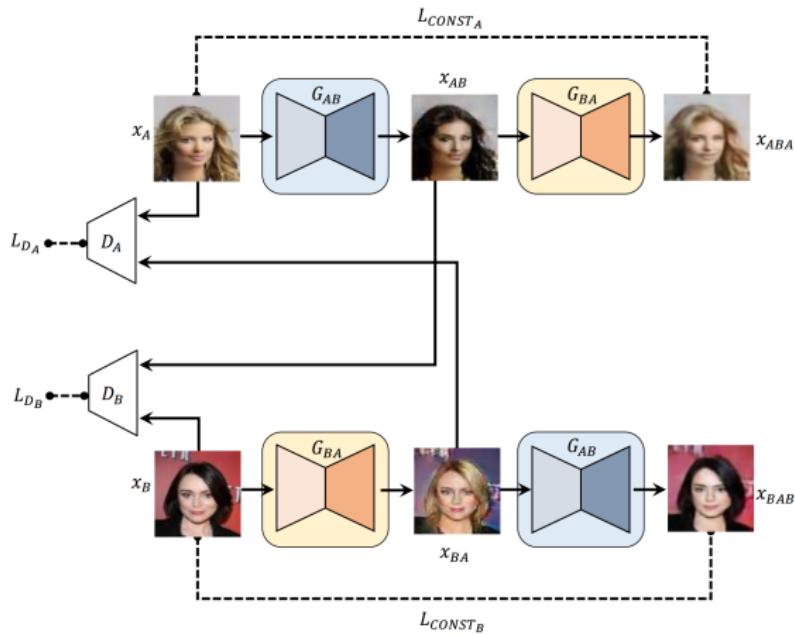
# Variations autour des architectures II

Introduire une contrainte de reconstruction (cf. auto-encodeur) :  
Cycle-GAN [36]



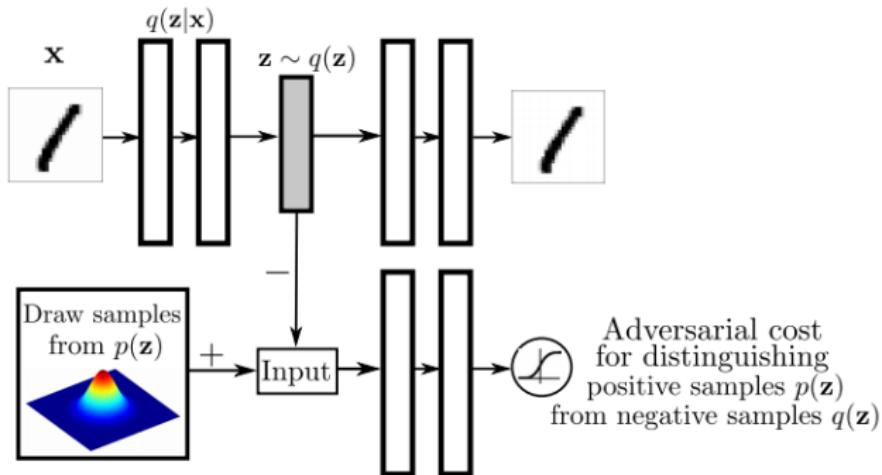
# Variations autour des architectures III

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]



## Quelques applications

- ▶ Génération de visage photo-réalistes  
[DiscoFaceGAN \[10\]](#)  
[Style GAN \[17, 18\]](#)
- ▶ Transfert de style  
[Pix2Pix](#)
- ▶ Manipulation d'image (Inpainting, super résolution, rendu photo [31]...)  
[Compétition AIM](#), [Compte-rendu Inpainting](#), [Proceedings Image relighting \[30\]](#)
- ▶ Deep Fake  
[DeepFaceLab](#)
- ▶ Génération conditionnelle
  - texte -> image : [DALL-E](#)
  - texte -> texte : [GPT-2](#)

# Modèles génératifs adversaires

## Points clés des GAN

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

## Utilisations

- ▶ 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 generative adversarial networks.  
In *International conference on machine learning*, pages 214–223. PMLR, 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.
- [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.
- [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.
- [7] David Berthelot, Thomas Schumm, and Luke Metz.  
Began : Boundary equilibrium generative adversarial networks.  
*arXiv preprint arXiv :1703.10717*, 2017.
- [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.

# Références II

- [9] Javiera Castillo-Navarro, Bertrand Le Saux, Alexandre Boulch, Nicolas Audebert, and Sébastien Lefèvre.  
Semi-supervised semantic segmentation in earth observation : The minifrance suite, dataset analysis and multi-task network study.  
*arXiv preprint arXiv :2010.07830*, 2020.
- [10] 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.
- [11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova.  
Bert : Pre-training of deep bidirectional transformers for language understanding.  
*arXiv preprint arXiv :1810.04805*, 2018.
- [12] Carl Doersch.  
Tutorial on variational autoencoders.  
*arXiv preprint arXiv :1606.05908*, 2016.
- [13] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio.  
Generative adversarial networks.  
*arXiv preprint arXiv :1406.2661*, 2014.
- [14] 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.
- [15] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros.  
Image-to-image translation with conditional adversarial networks.  
*In Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

# Références III

- [16] 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.
- [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.  
In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 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.
- [23] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey.  
Adversarial autoencoders.  
*arXiv preprint arXiv :1511.05644*, 2015.

# Références IV

- [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, 06–11 Aug 2017. PMLR.
- [27] 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.
- [28] 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.
- [29] 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.
- [30] 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.

# Références V

- [31] Ayush Tewari, Ohad Fried, Justus Thies, Vincent Sitzmann, Stephen Lombardi, Kalyan Sunkavalli, Ricardo Martin-Brualla, Tomas Simon, Jason Saragih, Matthias Nießner, et al.  
**State of the art on neural rendering.**  
In *Computer Graphics Forum*, volume 39, pages 701–727. Wiley Online Library, 2020.
- [32] 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.
- [33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin.  
**Attention is all you need.**  
In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6000–6010, 2017.
- [34] Cédric Villani.  
*Optimal transport : old and new*, volume 338.  
Springer Science & Business Media, 2008.
- [35] 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.
- [36] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros.  
**Unpaired image-to-image translation using cycle-consistent adversarial networks.**  
In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017.