

Apprentissage & Reconnaissance des Formes: Auto-encodeurs & modèle génératifs (GANs) (9/10)

S. Herbin, **B. Le Saux**, A. Boulch

`bertrand.le_saux@onera.fr`

11 février 2018

Introduction

Introduction

- ▶ Jusqu'ici, beaucoup de classification...
- ▶ et depuis peu la régression (et l'étude de concepts de régularisation)

Objectifs

- ▶ Aujourd'hui : de la régression encore plus poussée, sous forme d'apprentissage profond;
- ▶ Auto-encodeurs \Rightarrow apprendre un espace de représentation;
- ▶ Modèles Génératifs Adversaires (GANs) \Rightarrow régression par apprentissage profond.

Arbres de décision et méthodes ensemblistes : plan

Auto-encodeurs

- Principes de l'auto-encodage

- Variantes d'auto-encodeurs

- Auto-encodeurs de débruitage

Modèles Génératifs Adversaires

- GANs

- Générateurs : encodeurs-décodeurs

- Conditional GANs

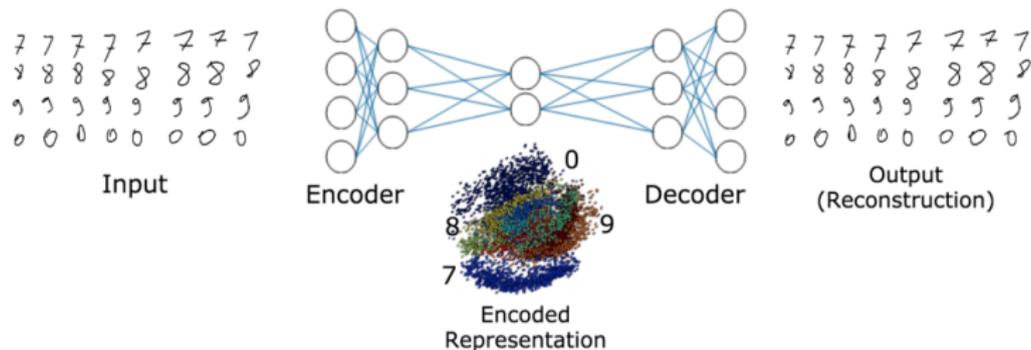
Conclusion

Auto-encodeurs

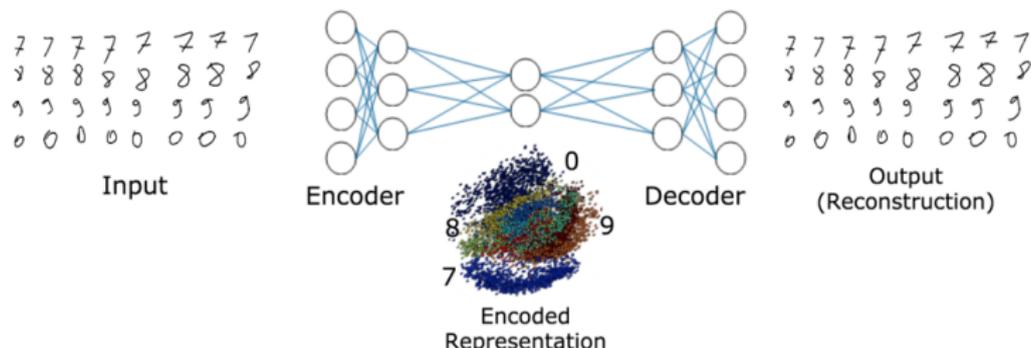
Auto-encodeurs : principes

Définition

Réseau de neurones capable d'apprendre un code représentatif des données de manière non-supervisée.

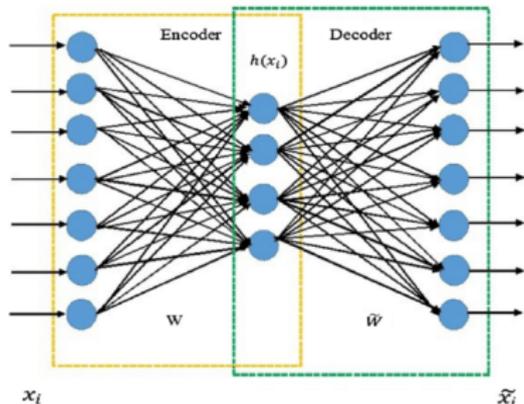


Auto-encodeurs : principes



- ▶ Apprend un code capable de représenter complètement le signal d'entrée, c'est à dire suffisant pour revenir aux données d'origine;
- ▶ Non-supervisé ? ➡ ou bien supervisé avec l'entrée (reconstruction);
- ▶ Intuition: les données peuvent contenir du bruit, des valeurs non représentatives, et la transformation apprise permet d'extraire l'information importante.

Auto-encodeurs : principes

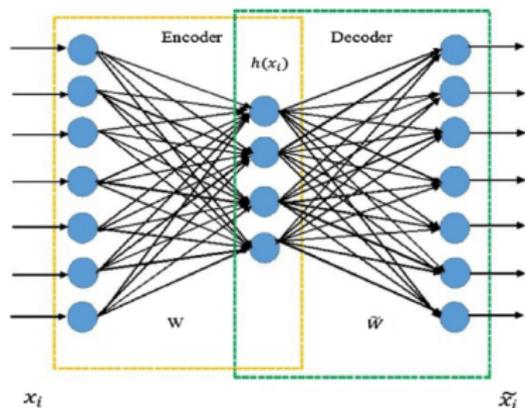


Partie encodeur

Historiquement, apprendre le code consiste à :

- ▶ Compresser les données. \implies réduction de dimensionalité (lien avec l'ACP)
- ▶ Extraire des caractéristiques représentatives (*features*) \implies sélection de caractéristiques

Auto-encodeurs : principes

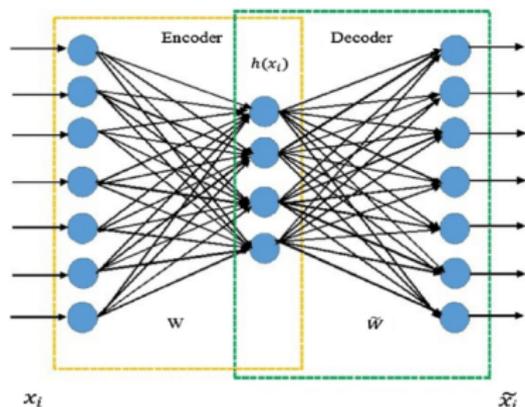


Partie encodeur

Aujourd'hui, apprendre le code, c'est :

- ▶ Construire un espace de représentation correspondant à des variables cachées \Rightarrow *deep learning*
- ▶ Auto-encodeur = cas spécial de réseau *feed-forward*, qui peut être appris par les techniques classiques (descente de gradient par minibatches et rétro-propagation)

Auto-encodeurs : principes



Partie décodeur

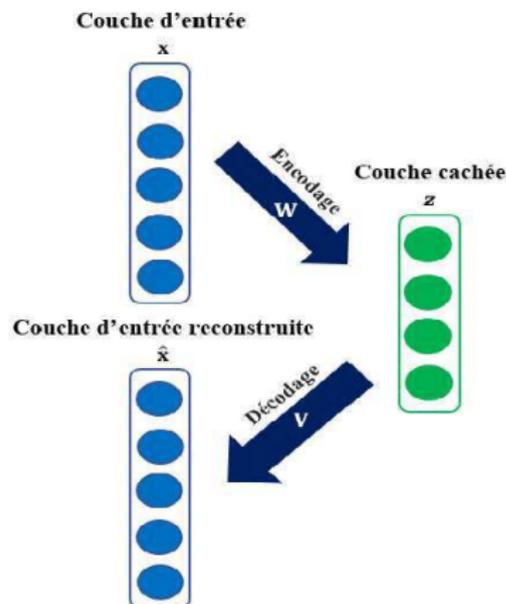
(Re-)construire à partir d'un code :

- ▶ Permet de reconstruire le plus fidèlement possible une donnée (signal, image, etc.)
- ▶ Générer une donnée jamais vue en combinant deux codes
 - ⇒ Modèle génératif

Auto-encodeurs : principes

Structure

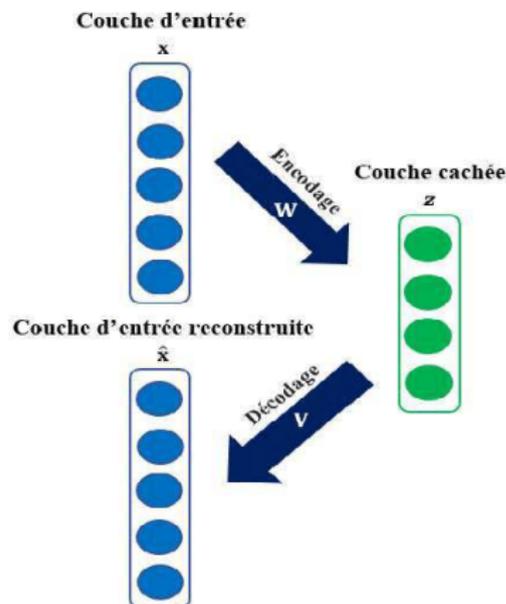
- ▶ **Encoder** $\Phi : \mathcal{X} \rightarrow \mathcal{F}$
- ▶ **Encoder** $\Psi : \mathcal{F} \rightarrow \mathcal{X}$
- ▶ **Code** $z = \sigma(Wx + b)$
- ▶ **Reconstruction**
 $\hat{x} = \sigma'(Vz + b')$
- ▶ **Fonction de pénalité:**
erreur de reconstruction
 $\mathcal{L}(x, \hat{x}) = \|x - \hat{x}\|^2$



Auto-encodeurs : principes

Structure

- ▶ **Fonction de pénalité:**
erreur de reconstruction
$$\mathcal{L}(x, \hat{x}) = \|x - \sigma'(V\sigma(Wx + b) + b')\|^2$$
- ▶ Si $\dim(\mathcal{F}) \leq \dim(\mathcal{X})$,
alors *compression*
- ▶ Si $\dim(\mathcal{F}) \geq \dim(\mathcal{X})$,
alors risque d'apprendre la
fonction *identité*...

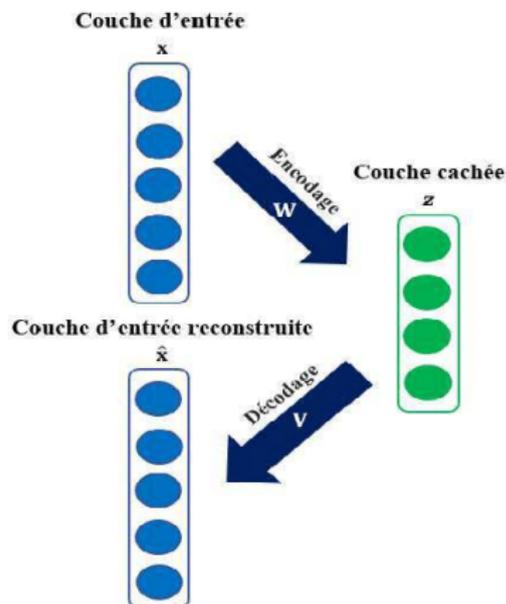


Auto-encodeurs : apprentissage

Entraînement

Pour chaque échantillon x :

1. Passe *feed-forward* pour activer les couches cachées et calculer \hat{x}
2. Mesure de l'erreur $\mathcal{L}(x, \hat{x})$
3. Rétro-propagation de l'erreur à travers le réseau et mise-à-jour des poids (W, V)



Auto-encodeurs parcimonieux

Principe général

Parcimonie (*sparsity* ou parfois *sparsité* par anglicisme) : principe consistant à n'utiliser qu'un minimum de causes pour expliquer un phénomène.

⇒ Objectif : Chercher des codes compacts

k-sparse auto-encoders

1. Identifier les k plus fortes activations des couches cachées
2. Mettre à 0 les autres activations

Auto-encodeurs parcimonieux

⇒ Objectif : Chercher des codes compacts

Régularisation parcimonieuse

- ▶ Soit $\dot{z}_j = \frac{1}{m} \sum_1^m z_j$ moyenne des activations cachées
- ▶ Objectif : $\dot{z}_j \simeq \rho$ avec ρ proche de 0
- ▶ Régularisation avec le terme de parcimonie:

$$\text{KL}(\dot{z}||\rho) = \sum_j \left(\rho \log\left(\frac{\rho}{\dot{z}_j}\right) + (1 - \rho) \log\left(\frac{1-\rho}{1-\dot{z}_j}\right) \right)$$

$$\Rightarrow \mathcal{L}(x, \hat{x}) = \|x - \hat{x}\|^2 + \text{KL}(\dot{z}||\rho)$$

Auto-encodeurs de débruitage

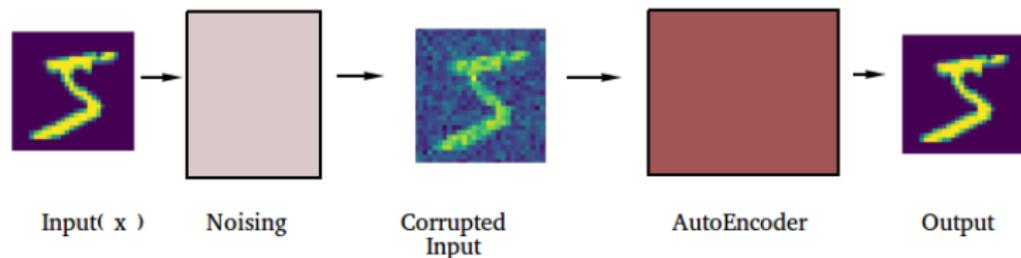
Principe général

Utilisation des auto-encodeurs pour apprendre à débruiter un signal, une image.



⇒ Objectif : apprendre une *bonne* représentation, c'à-d. une représentation *robuste* à un signal d'entrée corrompu.

Auto-encodeurs de débruitage



Denosing auto-encoder

1. Bruiter les échantillons : $x \mapsto \tilde{x}$
2. Optimiser les paramètres de l'auto-encodeur pour $\mathcal{L}(x, \hat{x})$

Auto-encodeurs : Résumé

Points clés des auto-encodeurs

- + Non-supervisé
- + Réduction de dimensionalité

Utilisations

- + Encodage (apprentissage de représentations)
- + Modèles génératifs
- + Débruitage

Modèles Génératifs Adversaires

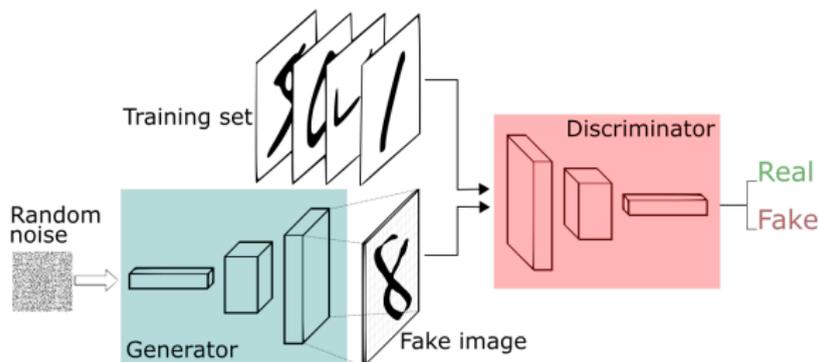
Modèles Génératifs Adversaires



Problème

- ▶ Décodeur = Modèle génératif (ou **Générateur**)
- ▶ Peut-on entraîner un modèle génératif qui générerait des signaux x à partir d'un code aléatoire z ?
- ▶ *Par exemple par régression avec une norme $L2$?*
- ▶ *Pas vraiment...*

Modèles Génératifs Adversaires



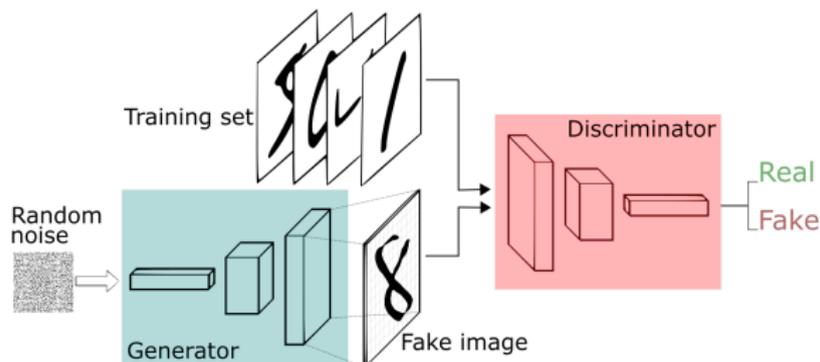
Solution

Goodfellow et al, "Generative Adversarial Networks", NIPS'2016

- ▶ Un réseau **générateur** G apprend à générer des images...
- ▶ Un réseau **discriminateur** D apprend à valider la vraisemblance des images produites (*log loss*).
- ▶ Les 2 réseaux rivalisent dans un cycle d'optimisations alternées:

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

Modèles Génératifs Adversaires



Pénalités permettant une optimisation plus stable

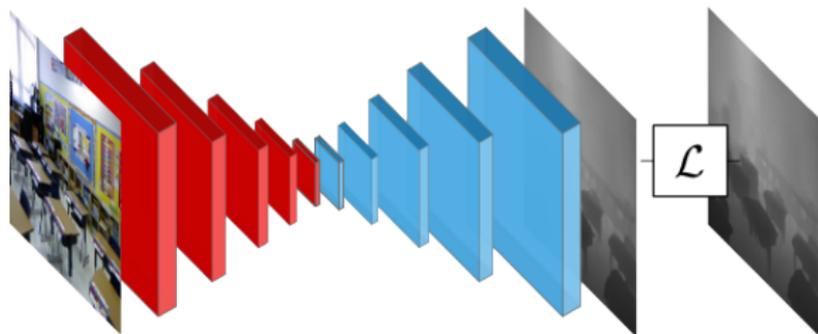
- ▶ *Least-square loss*:

$$\min_D V(D) = \mathbb{E}_{x \sim p_{data}(x)} [(D(x) - 1)^2] + \mathbb{E}_{z \sim p_z(z)} [(D(G(z)))^2]$$

$$\min_G V(G) = \mathbb{E}_{z \sim p_z(z)} [(D(G(z)) - 1)^2]$$

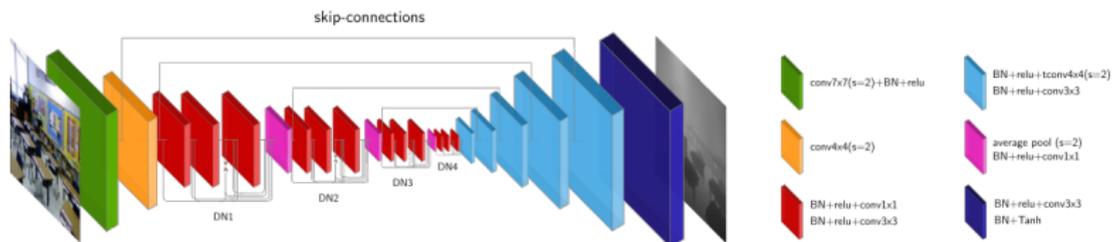
- ▶ *Wasserstein distance*...

Générateurs: encodeurs-décodeurs



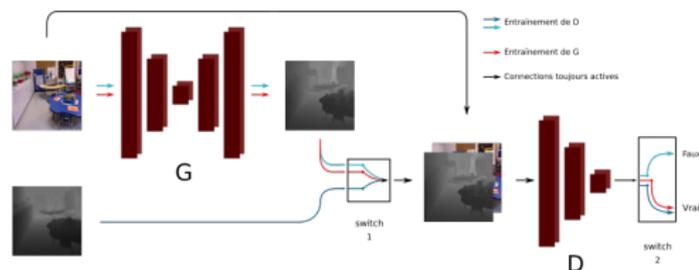
- ▶ Un réseau encodeur-décodeur a la même structure qu'un auto-encodeur...
- ▶ mais l'espace cible n'est pas le même !
 - ▶ **En classification** : segmentation sémantique (voir cours deep learning)
 - ▶ **En régression** : estimation d'une fonction du signal d : signal débruité, image super-résolue, profondeur...

Générateurs: encodeurs-décodeurs



- ▶ Seule la fonction de perte \mathcal{L} change, en régression :
 - ▶ L2 : $\mathcal{L}(x, d) = \|x - G(z)\|^2$
 - ▶ L1 : $\mathcal{L}(x, d) = |x - G(z)|$
- ▶ Concernant le réseau, toutes les techniques sont utilisables : *skip connections* (U-Net), couches résiduelles (resnet), blocs denses (DenseNet), etc...

Conditional GANs (cGANs)



Pour que G traduise une image d'un domaine à un autre (*Pix2pix*):

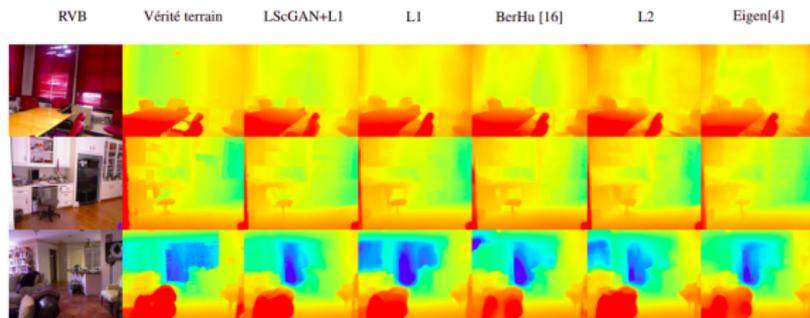
- ▶ Conditionner l'image générée $G(z)$ a une image existante z
- ▶ D estime la validité de la paire d'image $(z, G(x))$ ou (z, x)
- ▶ Pénalité:

$$\min_D V(D) = \mathbb{E}_{x \sim p_{data}(x)} [(D(x) - 1)^2] + \mathbb{E}_{z \sim p_z(z)} [(D(G(z)))^2] + \lambda |x - G(z)|$$

$$\min_G V(G) = \mathbb{E}_{z \sim p_z(z)} [(D(G(z)) - 1)^2]$$

Isola et al., *Image-to-Image Translation with Conditional Adversarial Networks*, CVPR'2017.

Application : estimation de la 3D (profondeur)



- ▶ Entraînement sur des paires (image, profondeur)
- ▶ L_1 suffit, mais L_{LSGAN} si suffisamment d'exemples...
- ▶ Vidéo : [img-AE-GANs/D3Net.mp4]

Carvalho et al., *On Regression Losses for Deep Depth Estimation*, ICIP'2018.

Application : transfert de style



- ▶ Générer des visages ressemblants à une personne, vieillir ...
- ▶ Vidéo : [img-AE-GANs/karras2018stylegan-video.mp4]

Karras et al., *A Style-Based Generator Architecture for Generative Adversarial Networks*, NIPS'2018.

GANs : Résumé

Points clés des GANs

- + Modèles génératifs
- + Rendu réaliste par optimisation de la vraisemblance du résultat
- Entraînement facilement instable \Rightarrow LS-GANs, Wassersteins-GANs
- Détails incertains \Rightarrow Version multi-échelle
- *Mode collapse*

Utilisations

- ▶ Régression, régression conditionnée à un signal
- ▶ Qualité image : super-résolution, débruitage...
- ▶ CGI : transfert de style, morphing, synthèse de films...
- ▶ Shopping online : essai de vêtements...
- ▶ Traduction de texte en image...

Conclusion

Cours n°9: Auto-encodeurs et GANs

Notions phares du jour

- ▶ Auto-encodeurs
- ▶ *Generative Adversarial Networks*

Concepts généraux

- ▶ Modèles génératifs
- ▶ Apprentissage de code (représentations)
- ▶ Apprentissage adversaire (lien avec le test de Turing)