



Réseaux Antagonistes Génératifs (GANs)

Yassine Ben Acha

Abdellah Ennajari

December 22, 2024



- ▶ Introduction
- ▶ Comprendre les GANs
- ▶ Avantages des GANs
- ▶ Le principe du GAN
- ▶ Exemples de GANs
- ▶ Le principe de GAN
- ▶ Comment entraîner ce modèle ?
- ▶ Comment mettre à jour le discriminateur ?
- ▶ Comment fonctionnent le générateur et le discriminateur ?
- ▶ Le flux du générateur
- ▶ Le flux du discriminateur
- ▶ Comment mettre à jour le générateur ?
- ▶ GAN Original
- ▶ Fonctions de Perte et Optimisation
- ▶ Conclusion

Introduction

1 Introduction

- **Présentation générale des GANs :** Les Réseaux Antagonistes Génératifs (GANs) sont une classe de modèles d'apprentissage profond inventée par Ian Goodfellow en 2014. Ils se composent de deux réseaux neuronaux : un générateur et un discriminateur, qui s'affrontent dans un jeu à somme nulle. L'objectif principal est de permettre au générateur de produire des données artificielles réaliste... en trompant le discriminateur.
- **Applications et motivations :** Les GANs trouvent des applications dans divers domaines tels que la création d'images réaliste...s, l'amélioration de la qualité des images, la synthèse de données, et la recherche médicale.

Comprendre les GANs en 3 points clés

2 Comprendre les GANs

- **G comme Génératif :** C'est comme un artiste qui apprend à créer de nouvelles images.
- **A comme Antagoniste :** Deux réseaux qui s'affrontent : l'un crée, l'autre juge.
- **N comme Réseaux de Neurones :** Utilisation d'intelligence artificielle avancée.



Les avantages des GANs

3 Avantages des GANs

- **Création de contenu :** Générer de nouvelles images, vidéos, musiques, textes, etc.
- **Apprentissage non supervisé/supervisé :** Apprendre directement à partir des données, sans étiquettes ou avec des données étiquetées (cGANs ou CycleGANs).
- **Meilleure compréhension des données :** Modéliser la distribution complète des données ($P(X)$).



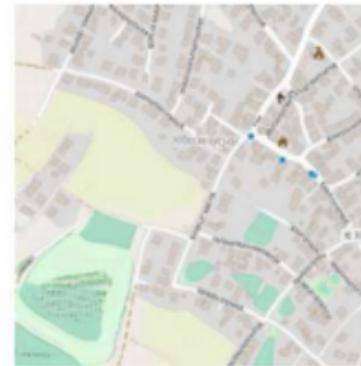
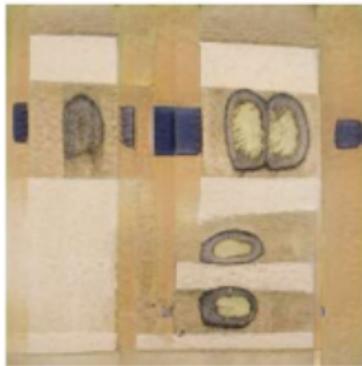
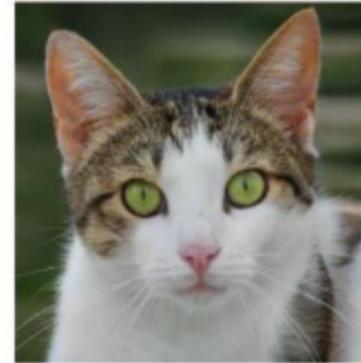
Le principe du GAN

4 Le principe du GAN

- **Le Générateur :** Crée des images artificielles de plus en plus réalistes.
- **Le Discriminateur :** Devine si une image est réelle ou artificielle.
- **L'apprentissage :** Les deux réseaux s'améliorent en s'affrontant, comme dans un jeu.

Que peut faire un GAN ?

5 Exemples de GANs



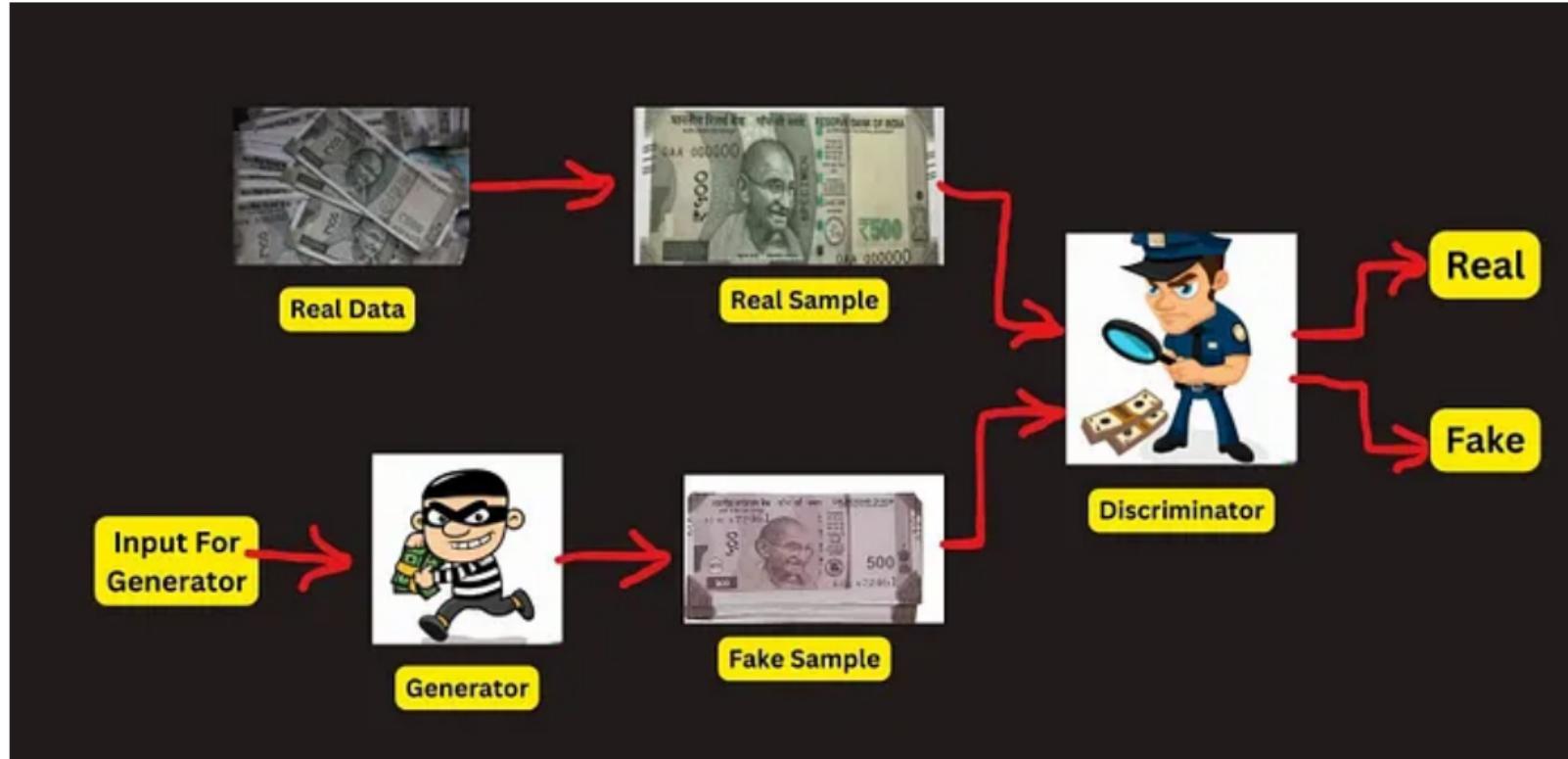
Quel visage est réel ?

5 Exemples de GANs



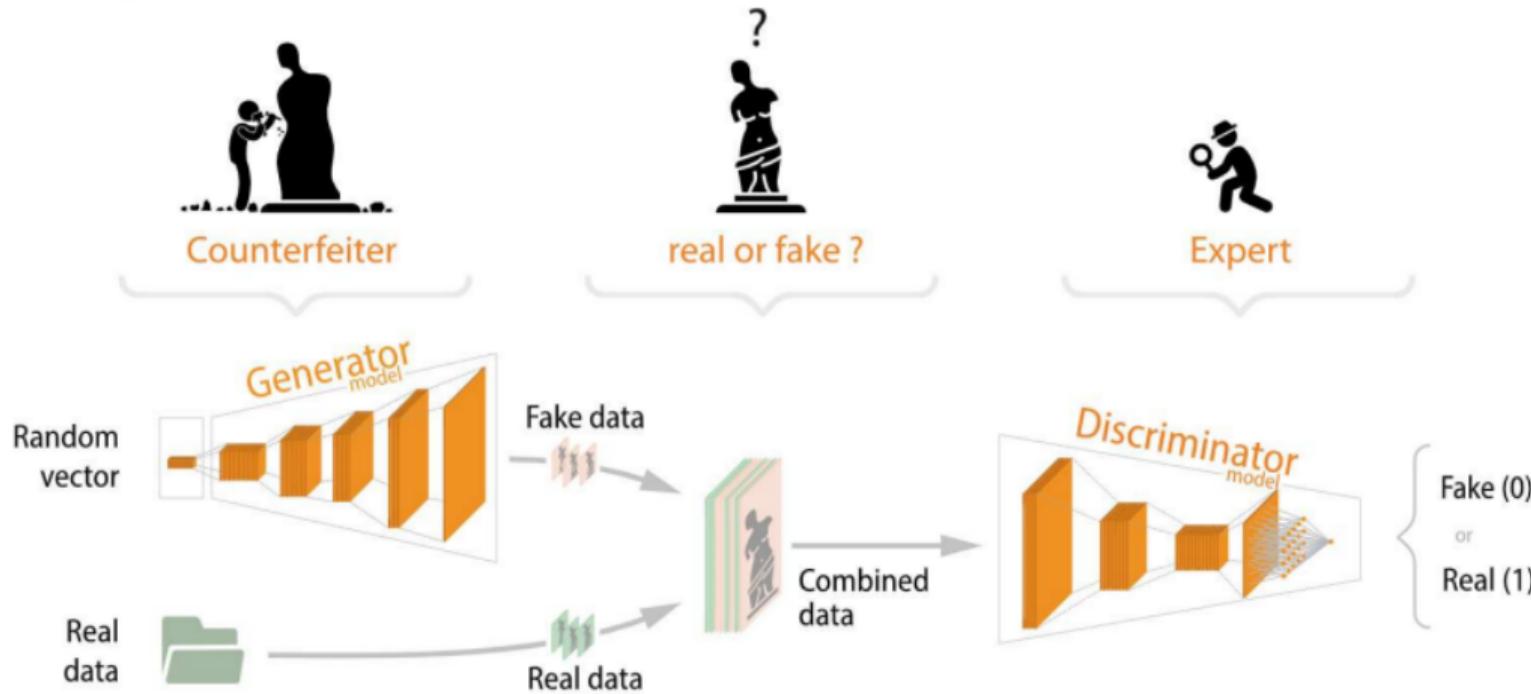
Exemple 1

5 Exemples de GANs



Principe

6 Le principe de GAN



Entraînement

7 Comment entraîner ce modèle ?

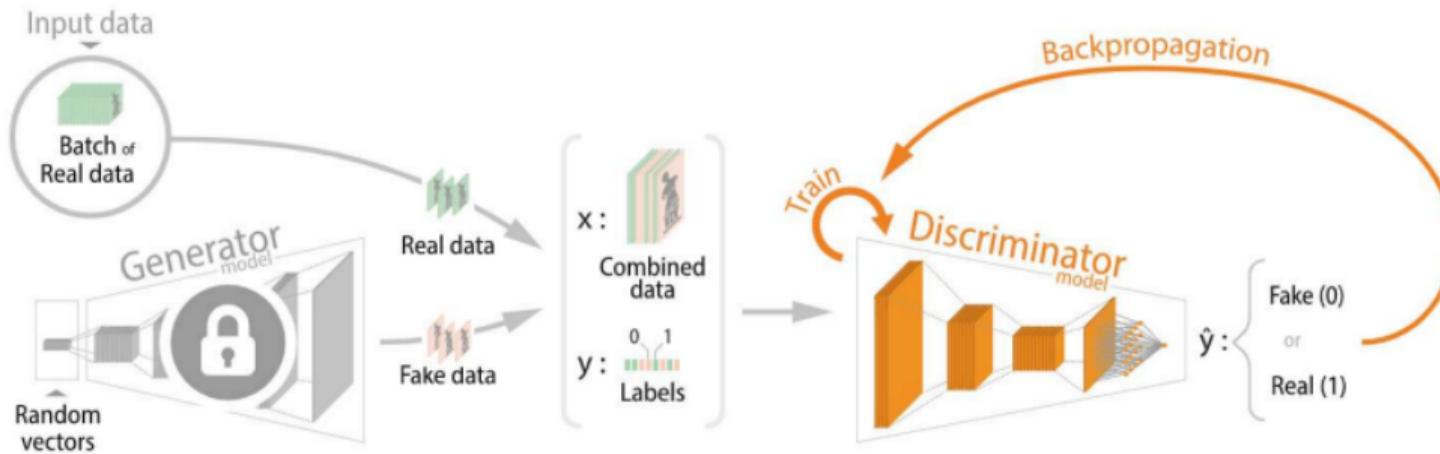


Entraînement

8 Comment mettre à jour le discriminateur ?

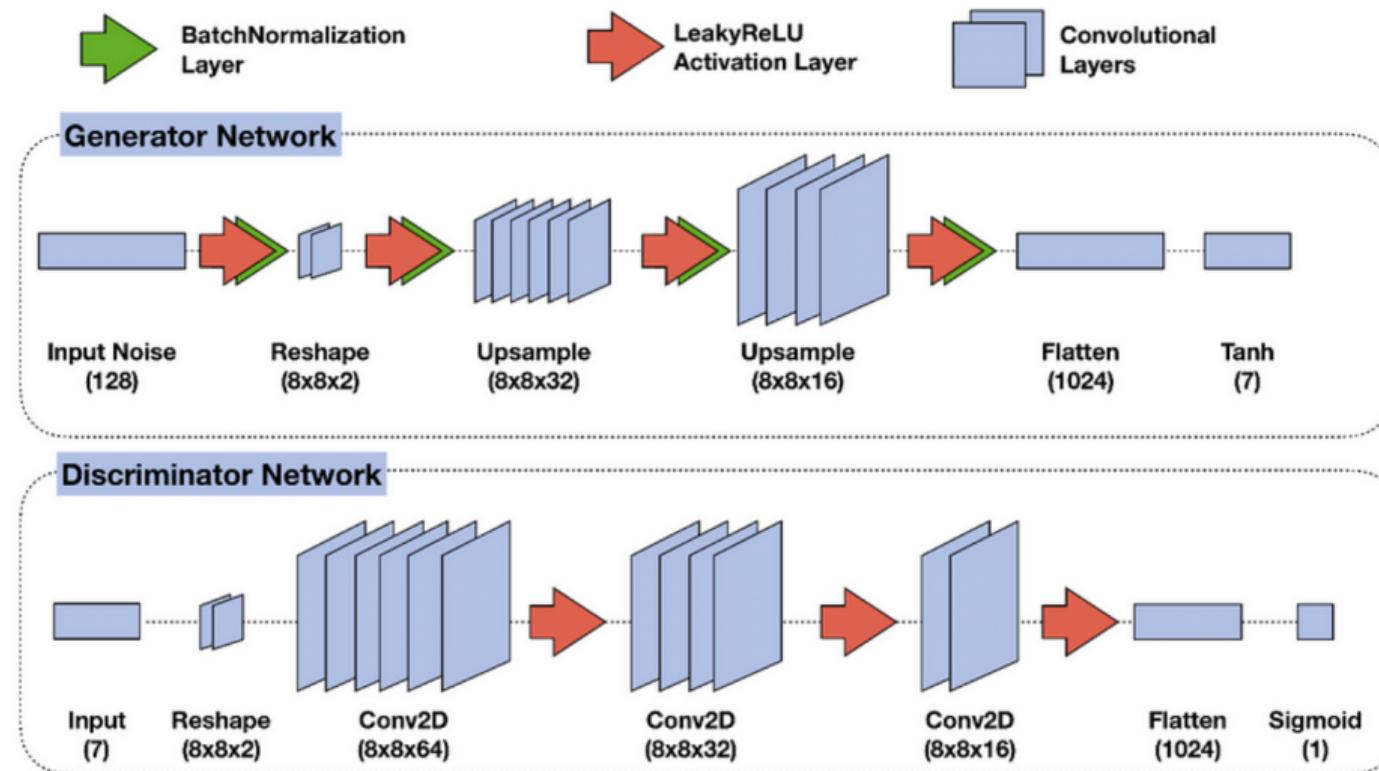
Step1 - Update Discriminator

Generator is locked



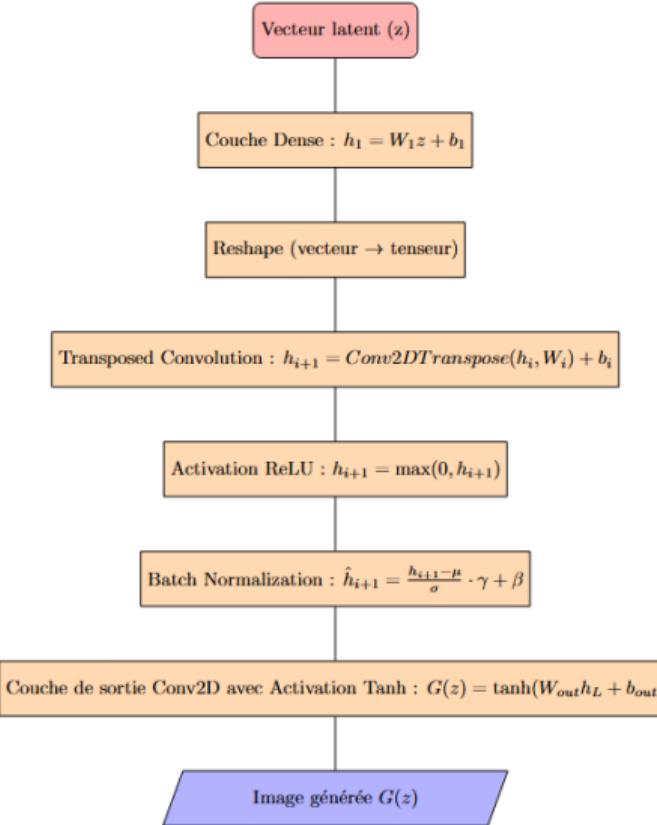
Entraînement

9 Comment fonctionnent le générateur et le discriminateur ?



Flux de Génération d'Images avec un GAN

10 Le flux du générateur



Objectif mathématique du générateur :

Minimiser la fonction de coût pour tromper le discriminateur :

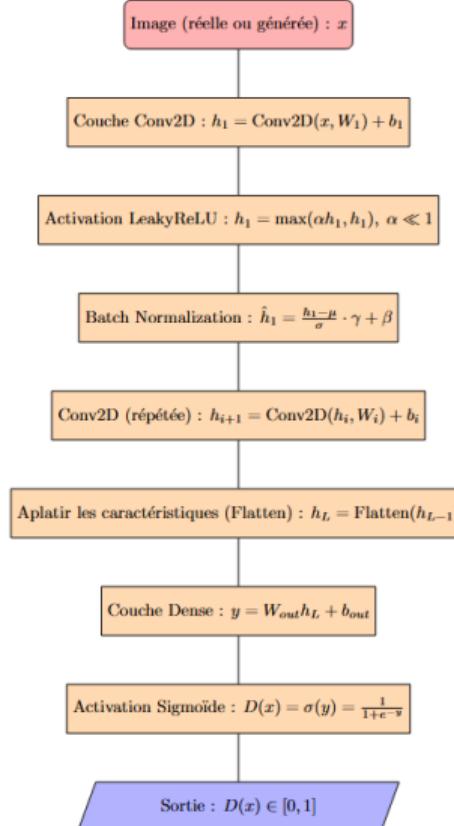
$$L_G = \mathbb{E}_{z \sim p_z(z)} [\log (1 - D(G(z)))]$$

ou dans certaines variantes :

$$L_G = -\mathbb{E}_{z \sim p_z(z)} [\log (D(G(z)))]$$

Flux de discriminateur avec un GAN

11 Le flux du discriminateur



Objectif mathématique du discriminateur :
 Maximiser la fonction de coût pour bien classer les images réelles et générées :

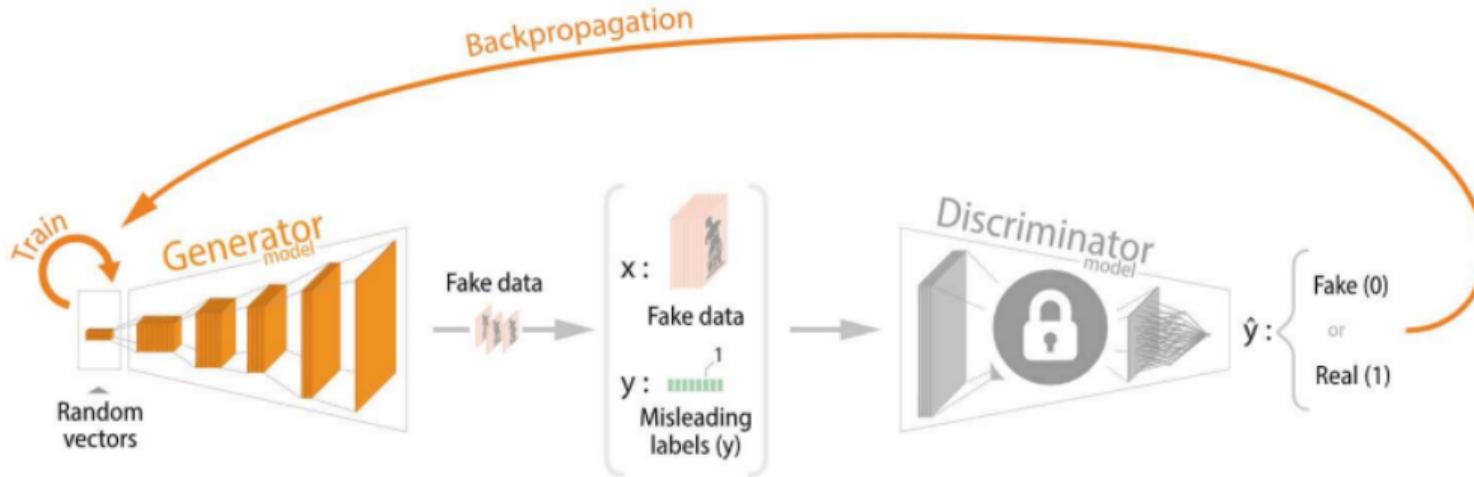
$$L_D = -\mathbb{E}_{x \sim p_{\text{data}}(x)} [\log(D(x))] - \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))]$$

Backpropagation

12 Comment mettre à jour le générateur ?

Step2 - Update Generator

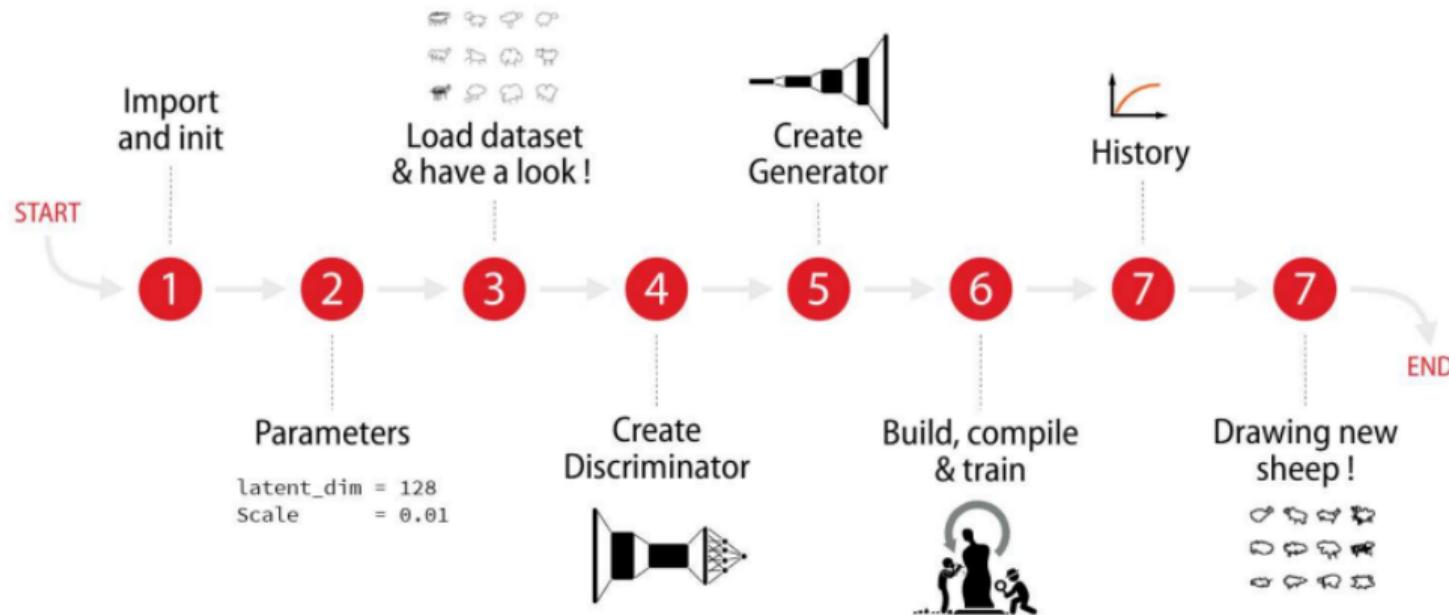
Discriminator is locked





Exemple

12 Comment mettre à jour le générateur ?



Relation entre Générateur et Discriminateur

12 Comment mettre à jour le générateur ?

Le GAN entier est un jeu à somme nulle entre le générateur et le discriminateur. Leur objectif commun peut être exprimé comme suit :

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

Explications :

- Le générateur G essaie de minimiser cette fonction en trompant D pour qu'il classe $G(z)$ comme réel.
- Le discriminateur D essaie de maximiser cette fonction en distinguant les vraies images des images générées.

Cross-Entropie Catégorique :

$$H(y, \hat{y}) = -\frac{1}{n_{\text{obs}}} \sum_{i=1}^{n_{\text{obs}}} \sum_{c=1}^{n_{\text{class}}} y_c \cdot \log(\hat{y}_c)$$

Rôle : Mesure l'erreur de classification.

Fonctions de perte GAN :

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

$$L_G = \frac{1}{m} \sum_{i=1}^m \left[\log \left(1 - D \left(G(z^{(i)}) \right) \right) \right]$$

Rôle : Optimise la génération d'images.

Définitions :

- $x^{(i)}$: données réelles provenant d'un ensemble de m valeurs.
- $z^{(i)}$: vecteur latent aléatoire.
- $D(x)$: sortie du discriminateur pour x (probabilité qu'une image réelle soit jugée réelle).
- $G(z^{(i)})$: sortie du générateur à partir d'un espace latent z (image générée).
- $D(G(z^{(i)}))$: sortie du discriminateur pour une image générée.

Le Discriminateur cherche à maximiser L_D :

$$\forall x \in X, D(x) \approx 1 \quad \text{et} \quad \forall z \in Z, D(G(z)) \approx 0$$

Le Générateur cherche à minimiser L_G :

$$\forall z \in Z, D(G(z)) \approx 1$$

Où :

- *faux* : 0
- *réel* : 1

Algorithm 1 : WGAN avec pénalité de gradient (Boucle du critique)

13 GAN Original

Algorithm 1 WGAN avec pénalité de gradient - Mise à jour du critique

Require: Coefficient de pénalité de gradient λ , nombre d'itérations du critique par itération du générateur n_{critic} , taille du lot m , hyperparamètres Adam α, β_1, β_2
Require: Paramètres initiaux du critique w_0 , paramètres initiaux du générateur θ_0

```
1: while  $\theta$  n'a pas convergé do
2:   for  $t = 1, \dots, n_{\text{critic}}$  do
3:     for  $i = 1, \dots, m$  do
4:       Échantillonner des données réelles  $\mathbf{x} \sim \mathbb{P}_r$ , variable latente  $\mathbf{z} \sim p(\mathbf{z})$ ,
    $\epsilon \sim U[0, 1]$ 
5:        $\hat{\mathbf{x}} \leftarrow G_\theta(\mathbf{z})$ 
6:        $\tilde{\mathbf{x}} \leftarrow \epsilon \mathbf{x} + (1 - \epsilon) \hat{\mathbf{x}}$ 
7:        $L^{(i)} \leftarrow D_w(\tilde{\mathbf{x}}) - D_w(\mathbf{x}) + \lambda (\|\nabla_{\tilde{\mathbf{x}}} D_w(\tilde{\mathbf{x}})\|_2 - 1)^2$ 
8:     end for
9:      $w \leftarrow \text{Adam}(\nabla_w \frac{1}{m} \sum_{i=1}^m L^{(i)}, w, \alpha, \beta_1, \beta_2)$ 
```

Algorithm 1 : WGAN avec pénalité de gradient (Mise à jour du générateur)

13 GAN Original

Algorithm 2 WGAN avec pénalité de gradient - Mise à jour du générateur

- 1: **while** θ n'a pas convergé **do**
 - 2: Échantillonner un lot de variables latentes $\{\mathbf{z}^{(i)}\}_{i=1}^m \sim p(\mathbf{z})$
 - 3: $\theta \leftarrow \text{Adam}(\nabla_{\theta} \frac{1}{m} \sum_{i=1}^m -D_w(G_{\theta}(\mathbf{z})), \theta, \alpha, \beta_1, \beta_2)$
 - 4: **end while**=0
-

Étape 1 : Initialisation

13 GAN Original

- **Initialiser les paramètres des modèles :**
 - θ_G pour le Générateur G
 - θ_D pour le Discriminateur D
- **Fixer les distributions :**
 - $p_{\text{data}}(x)$: Distribution des vraies données
 - $p_z(z)$: Distribution de bruit aléatoire (par exemple, gaussienne ou uniforme)
- **Fixer les hyperparamètres :**
 - Nombre d'itérations N
 - Taille du mini-lot m

Étape 2 : Boucle d'entraînement

13 GAN Original

Pour chaque itération i de 1 à N :

1. Entraîner le discriminateur D :

- Prendre un mini-lot de données réelles $\{x^{(1)}, \dots, x^{(m)}\} \sim p_{\text{data}}(x)$
- Générer un mini-lot de données synthétiques $\{G(z^{(1)}), \dots, G(z^{(m)})\}$ en échantillonnant $z \sim p_z(z)$
- Mettre à jour θ_D en maximisant :

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

2. Entraîner le générateur G :

- Échantillonner un nouveau mini-lot de bruit $\{z^{(1)}, \dots, z^{(m)}\} \sim p_z(z)$
- Mettre à jour θ_G en minimisant :

$$-\mathbb{E}_{z \sim p_z(z)}[\log D(G(z))]$$

3. Optionnel : Afficher les pertes de D et G pour suivre la convergence.

Étape 3 : Génération de nouvelles données

13 GAN Original

1. Après l'entraînement, utiliser uniquement le générateur G :
 - Échantillonner un bruit $z \sim p_z(z)$
 - Calculer $x_{\text{fake}} = G(z)$ pour générer des données synthétiques.
2. Si nécessaire, post-traiter x_{fake} pour correspondre au format attendu.

Fonctions de Perte

14 Fonctions de Perte et Optimisation

- **Fonction de perte du discriminateur :**

$$L_D = - \sum_{x \in \chi, z \in \zeta} \log(D(x)) + \log(1 - D(G(z)))$$

- **Fonction de perte du générateur :**

$$L_G = - \sum_{z \in \zeta} \log(D(G(z)))$$

Optimisation du Modèle

14 Fonctions de Perte et Optimisation

- **Optimisation du discriminateur :**

$$\max_D \left\{ \mathbb{E}_{x \sim p_{\text{data}}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))] \right\}$$

- **Optimisation du générateur :**

$$\min_G \left\{ \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))] \right\}$$

Entraînement du Discriminateur

14 Fonctions de Perte et Optimisation

- **Objectif :** Maximiser la fonction de perte du discriminateur.
- **Mise à jour des poids :**

$$\theta_D \leftarrow \theta_D + \eta \nabla_{\theta_D} \left\{ \mathbb{E}_{x \sim p_{\text{data}}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))] \right\}$$

Entraînement du Générateur

14 Fonctions de Perte et Optimisation

- **Objectif :** Minimiser la fonction de perte du générateur.
- **Mise à jour des poids :**

$$\theta_G \leftarrow \theta_G - \eta \nabla_{\theta_G} \left\{ \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))] \right\}$$

Conclusion

15 Conclusion

- Les GANs sont des modèles génératifs puissants pour apprendre la distribution sous-jacente des données.
- L'entraînement des GANs repose sur un jeu adversarial entre le générateur et le discriminateur.
- Les fonctions de perte et les techniques d'optimisation sont cruciales pour l'entraînement efficace des GANs.