Séance 1 : Théorie des Modèles de Diffusion et Équations

Julien Perez

November 25, 2024

Introduction aux Modèles de Diffusion

Objectifs de la séance :

- Comprendre les concepts de base des modèles génératifs.
- Explorer les principes des équations différentielles stochastiques.
- Analyser le processus de diffusion et sa pertinence en modélisation.

Contexte des modèles génératifs :

- Les modèles génératifs apprennent à imiter la distribution des données.
- Ils sont utilisés pour créer de nouvelles données réalistes (images, textes, sons).
- *Applications* : Génération d'images, synthèse vocale, etc.

Introduction à l'IA Générative

Qu'est-ce que l'IA Générative ?

- L'IA générative est une branche de l'intelligence artificielle qui se concentre sur la création de nouvelles données réalistes.
- Elle utilise des modèles statistiques génératifs pour apprendre la distribution des données et générer de nouvelles instances.

Applications :

- Génération d'images, de texte, de musique, de vidéos, etc.
- Utilisée dans les arts, le design, la médecine, et bien d'autres domaines.

Modèles Génératifs

Objectif des Modèles Génératifs :

- Apprendre la distribution sous-jacente des données d'entraînement.
- Générer de nouvelles données qui suivent cette distribution.

Types de Modèles Génératifs :

- Variational Autoencoders (VAE)
- Generative Adversarial Networks (GAN)
- Modèles de Diffusion
- Autoregressive Models

Applications de l'IA Générative

Génération d'Images :

- Création d'images réalistes de visages, de paysages, et d'objets.
- Utilisation dans les arts et le design.

Synthèse Audio :

- Création de voix synthétiques réalistes.
- Utilisation dans les assistants vocaux et les systèmes de synthèse vocale.

Génération de Texte :

- Création de textes cohérents et réalistes.
- Utilisation dans la création de contenu et les chatbots.

Structure Générale d'un VAE

Composants principaux :

1. **Encodeur :** Mappe les données dans un espace latent probabiliste.
2. **Réparamétrisation :** Permet un échantillonnage différentiable.
3. **Decodeur :** Reconstruit les données à partir de l'espace latent.

Architecture d'un VAE

1. **Encodeur :** Estime les paramètres $\mu(x)$ et $\sigma(x)$ de $q(z|x)$.
2. **Réparamétrisation :** Génère un vecteur latent z.
3. **Decodeur :** Reconstruit x à partir de z via $p(x|z)$.

Formulation Mathématique des VAE

Objectif des VAE :

- Maximiser la vraisemblance des données $p(x)$ en passant par un espace latent z.

Approche Bayésienne :

$$p(x) = \int p(x|z)p(z)dz$$

Approximation avec une distribution $q(z|x)$:

$$\log p(x) \geq \mathbb{E}_{q(z|x)} [\log p(x|z)] - \mathrm{KL}(q(z|x) \| p(z))$$

Fonction de Perte :

$$\mathcal{L}(x, z) = \underbrace{-\mathbb{E}_{q(z|x)} [\log p(x|z)]}_{\text{Perte de reconstruction}} + \underbrace{\mathrm{KL}(q(z|x) \| p(z))}_{\text{Régularisation KL}}$$

Variational Autoencoders (VAE)

Principe des VAE :

- Encode les données dans un espace latent probabiliste.
- Ajoute un terme de régularisation pour rendre l'espace latent continu.

Fonction de Perte des VAE :

$$\mathcal{L}(x, z) = \underbrace{-\mathbb{E}_{q(z|x)}\left[\log p(x|z)\right]}_{\text{Perte de reconstruction}} + \underbrace{\mathrm{KL}(q(z|x)\|p(z))}_{\text{Régularisation KL}}$$

- $p(x|z)$: Distribution conditionnelle des données générées.
- $q(z|x)$: Distribution conditionnelle de l'espace latent.
- $p(z)$: Distribution a priori de l'espace latent.

Reparametrization Trick

Problème :

- La rétropropagation ne peut pas être effectuée directement sur $z \sim q(z|x)$ à cause de la nature non différentiable de l'échantillonnage.

Solution :

- Introduire une transformation différentiable :

$$z = \mu + \sigma \cdot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$$

Avantages :

- μ et σ deviennent différentiables, ce qui permet un entraînement fluide des réseaux.

Variational Autoencoders (VAE)

Principe des VAE :

- Encode les données dans un espace latent probabiliste.
- Ajoute un terme de régularisation pour rendre l'espace latent continu.

Fonction de Perte des VAE :

- **Perte de reconstruction** : Distance entre les données originales et reconstruites.
- **Divergence KL** : Mesure la distance entre la distribution latente et une distribution gaussienne.

Limites des VAE

Problèmes des VAE :

- Reconstruction souvent floue, car les hypothèses gaussiennes limitent la capacité du modèle.
- Difficulté à modéliser des distributions de données complexes ou multimodales.

Impact sur la Génération :

- Génération d'images de moindre qualité par rapport aux GAN et aux modèles de diffusion.

Processus d'Entraînement des VAE

Étapes principales :

- **Encodage :** L'encodeur génère les paramètres $\mu(x)$ et $\sigma^2(x)$ d'une distribution latente gaussienne $q(z|x)$.
- **Echantillonnage :** Utilisation du **reparametrization trick** :

$$z = \mu(x) + \sigma(x) \cdot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$$

- **Décodage :** Le décodeur génère $p(x|z)$ à partir de l'échantillon latent z.
- **Optimisation :** Minimisation de la fonction de perte $\mathcal{L}(x, z)$ à l'aide de la descente de gradient.

Architecture typique :

- Encodeur : Réseau de neurones qui apprend $\mu(x)$ et $\sigma^2(x)$.
- Décodeur : Réseau de neurones pour générer x à partir de z.

Utilisation des VAE

Applications principales :

- **Génération de données :** Création d'images, de sons ou de textes synthétiques.
- **Interpolation dans l'espace latent :** Génération fluide entre deux points dans l'espace latent.
- **Détection d'anomalies :** Identifier les échantillons qui ne s'adaptent pas bien à l'espace latent appris.
- **Compression de données :** Encodage efficace dans un espace de faible dimension.

Cas d'usage :

- Génération de visages réalistes (images de faible résolution).
- Représentation compacte dans les systèmes de recommandation.
- Reconstruction de données médicales pour la détection d'artefacts.

Exemple en PyTorch : Modèle VAE

```
import torch
from torch import nn

class Encoder(nn.Module):
    def __init__(self, input_dim, latent_dim):
        super().__init__()
        self.fc = nn.Linear(input_dim, 128)
        self.mu = nn.Linear(128, latent_dim)
        self.log_var = nn.Linear(128, latent_dim)

    def forward(self, x):
        h = torch.relu(self.fc(x))
        return self.mu(h), self.log_var(h)

class Decoder(nn.Module):
    def __init__(self, latent_dim, output_dim):
        super().__init__()
        self.fc = nn.Linear(latent_dim, 128)
        self.recon = nn.Linear(128, output_dim)

    def forward(self, z):
        h = torch.relu(self.fc(z))
        return torch.sigmoid(self.recon(h))
```

Exemple en PyTorch : VAE complet

```
class VAE(nn.Module):
    def __init__(self, input_dim, latent_dim):
        super().__init__()
        self.encoder = Encoder(input_dim, latent_dim)
        self.decoder = Decoder(latent_dim, input_dim)

    def reparameterize(self, mu, log_var):
        std = torch.exp(0.5 * log_var)
        epsilon = torch.randn_like(std)
        return mu + std * epsilon

    def forward(self, x):
        mu, log_var = self.encoder(x)
        z = self.reparameterize(mu, log_var)
        recon_x = self.decoder(z)
        return recon_x, mu, log_var
```

Exemple en PyTorch : Fonction de Perte

Fonction de perte :

```
def vae_loss(recon_x, x, mu, log_var):
    recon_loss = nn.functional.binary_cross_entropy(recon_x, x,
                                                    reduction='sum')
    kl_div = -0.5 * torch.sum(1 + log_var - mu.pow(2) - log_var.exp())
    return recon_loss + kl_div
```

Boucle d'entraînement :

```
optimizer = torch.optim.Adam(vae.parameters(), lr=1e-3)
for epoch in range(num_epochs):
    for x in dataloader:
        x = x.view(-1, input_dim)
        recon_x, mu, log_var = vae(x)
        loss = vae_loss(recon_x, x, mu, log_var)

        optimizer.zero_grad()
        loss.backward()
        optimizer.step()
```

Generative Adversarial Networks (GAN)

Principe des GAN :

- Utilise deux réseaux de neurones : un générateur et un discriminateur.
- Le générateur crée des données synthétiques, tandis que le discriminateur essaie de distinguer les données réelles des données générées.

Fonction de Perte des GAN :

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

- $D(x)$: Probabilité que x soit une donnée réelle.
- $G(z)$: Donnée générée à partir du bruit z.

GAN - Generative Adversarial Networks

Concept de Base :

- Les GAN sont des modèles génératifs composés de deux réseaux de neurones : un générateur et un discriminateur.
- Le générateur crée des données synthétiques, tandis que le discriminateur essaie de distinguer les données réelles des données générées.

Objectif :

- Le générateur apprend à produire des données de plus en plus réalistes pour tromper le discriminateur.
- Le discriminateur apprend à mieux distinguer les données réelles des données générées.

Architecture des GAN

Composants Principaux :

- **Générateur (G)** : Réseau de neurones qui génère des données à partir d'un vecteur de bruit aléatoire.
- **Discriminateur (D)** : Réseau de neurones qui classe les données comme réelles ou générées.

Processus d'Entraînement :

- Le générateur et le discriminateur sont entraînés simultanément dans un jeu à somme nulle.
- Le générateur essaie de maximiser l'erreur du discriminateur, tandis que le discriminateur essaie de minimiser sa propre erreur.

Fonction de Perte des GAN

Objectif d'Apprentissage :

- Le générateur essaie de minimiser la probabilité que le discriminateur classe correctement les données générées.
- Le discriminateur essaie de maximiser la probabilité de classer correctement les données réelles et générées.

Formulation Mathématique :

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

- $D(x)$: Probabilité que x soit une donnée réelle.
- $G(z)$: Donnée générée à partir du bruit z.

Défis des GAN

Problèmes Courants :

- **Collapse de Mode** : Le générateur produit des échantillons avec peu de diversité.
- **Instabilité de l'Entraînement** : Les GAN peuvent être difficiles à entraîner et à converger.
- **Équilibre entre G et D** : Le générateur et le discriminateur doivent être équilibrés pour un apprentissage efficace.

Solutions :

- Utilisation de techniques comme le bruitage des labels, l'ajout de bruit aux données, et l'utilisation de fonctions de perte alternatives.

Variantes des GAN

DCGAN (Deep Convolutional GAN) :

- Utilise des réseaux de convolution profonds pour le générateur et le discriminateur.
- Améliore la stabilité de l'entraînement et la qualité des images générées.

WGAN (Wasserstein GAN) :

- Utilise la distance de Wasserstein pour mesurer la différence entre les distributions réelles et générées.
- Améliore la stabilité de l'entraînement et réduit le collapse de mode.

StyleGAN :

- Introduit des blocs de style pour contrôler les caractéristiques des images générées.
- Permet une génération d'images de haute qualité avec un contrôle fin des styles.

Applications des GAN

Génération d'Images :

- Création d'images réalistes de visages, de paysages, et d'objets.
- Utilisation dans les arts et le design.

Super-résolution :

- Amélioration de la résolution des images basse qualité.
- Applications en photographie et en imagerie médicale.

Traduction Image-à-Image :

- Conversion d'images d'un domaine à un autre (ex : photo en peinture).
- Utilisation dans les effets spéciaux et la post-production.

GAN pour la Synthèse Audio

Génération de Voix :

- Création de voix synthétiques réalistes.
- Utilisation dans les assistants vocaux et les systèmes de synthèse vocale.

Amélioration de la Qualité Audio :

- Restauration et amélioration de la qualité des enregistrements audio.
- Applications dans la musique et les podcasts.

Conversion de Voix :

- Transformation de la voix d'une personne en celle d'une autre.
- Utilisation dans les films et les jeux vidéo.

GAN pour la Génération de Texte

Création de Texte Réaliste :

- Génération de textes cohérents et réalistes.
- Utilisation dans la création de contenu et les chatbots.

Traduction de Texte :

- Traduction automatique de texte d'une langue à une autre.
- Applications dans les services de traduction en ligne.

Génération de Poésie et de Prose :

- Création de poèmes et de prose littéraire.
- Utilisation dans les arts et la littérature.

Autoregressive Models

Principe des Modèles Autorégressifs :

- Modélise la distribution des données comme une séquence de décisions conditionnelles.
- Chaque élément de la séquence dépend des éléments précédents.

Exemple : Modèle de Langage

$$p(x_1, x_2, \ldots, x_n) = p(x_1)p(x_2|x_1) \cdots p(x_n|x_1, \ldots, x_{n-1})$$

- x_i : Élément de la séquence.
- $p(x_i|x_1, \ldots, x_{i-1})$: Probabilité conditionnelle.

Comparaison des premiers Modèles Génératifs

VAE vs GAN :

- **VAE** : Modélisent les distributions avec des hypothèses gaussiennes.
- **GAN** : Apprentissage par compétition (générateur vs discriminateur).

Modèles de Diffusion vs Autoregressive Models :

- **Diffusion** : Apprentissage séquentiel du bruitage/débruitage.
- **Autoregressive** : Modélisation séquentielle des dépendances conditionnelles.

Comparaison entre GAN et Modèles de Diffusion

Avantages des GAN :

- Capacité à générer des données très réalistes.
- Flexibilité dans la génération de différents types de données (images, audio, texte).

Inconvénients des GAN :

- Instabilité de l'entraînement et risque de collapse de mode.
- Nécessité de techniques avancées pour stabiliser l'entraînement.

Avantages des Modèles de Diffusion :

- Stabilité de l'entraînement et absence de collapse de mode.
- Capacité à générer des données de haute qualité avec une diversité élevée.

Conclusion intermédiaire sur les GAN

Résumé :

- Les GAN sont des outils puissants pour la génération de données réalistes.
- Ils présentent des défis en termes de stabilité de l'entraînement et de diversité des données générées.

Perspectives Futures :

- Amélioration des techniques d'entraînement pour stabiliser les GAN.
- Intégration des GAN avec d'autres modèles génératifs pour combiner leurs avantages.

Questions :

- Y a-t-il des questions ou des points à éclaircir sur les GAN ?

Introduction aux Processus de Diffusion

Concept Clé :

- Le processus de diffusion transforme progressivement des données en bruit aléatoire.
- Le modèle apprend l'inverse : comment transformer le bruit en données structurées.

Avantage : Une approche progressive pour modéliser des distributions complexes.

Denoising Diffusion Probabilistic Models (DDPM)

DDPM (Ho et al., 2020) :

- Modèle de diffusion pionnier pour la génération d'images.
- Bruitage progressif des données et débruitage à l'aide d'un réseau de neurones.

Points Clés du Modèle :

- Utilisation de la probabilité conditionnelle pour reconstruire x_{t-1} à partir de x_t.
- Fonction de perte : MSE entre données bruitées et reconstruction.
- Résultat : Génération d'images réalistes, avec qualité comparable aux GAN.

Référence : *Ho, J., Jain, A., Abbeel, P. (2020). Denoising Diffusion Probabilistic Models.*

Équations Différentielles Stochastiques (EDS)

Concepts clés :

- Une EDS décrit l'évolution d'une variable aléatoire au cours du temps.
- Formule de base :

$$dX_t = \mu(X_t, t)dt + \sigma(X_t, t)dW_t$$

- μ : Terme de dérive (tendance moyenne).
- σ : Terme de diffusion (incertitude ou bruit).
- dW_t : Bruit de Wiener (processus brownien).

Processus de Diffusion

Définition : Un processus de diffusion est un modèle stochastique continu décrivant la manière dont des particules ou des informations se dispersent.

- Exemples :
  - Diffusion de chaleur dans un matériau.
  - Modélisation de la distribution d'actifs financiers.
- Propriétés importantes :
  - Continuité des trajectoires.
  - Dépendance aux conditions initiales.

Modèles de Diffusion

Principe des Modèles de Diffusion :

- Transforme progressivement des données en bruit aléatoire.
- Apprend à inverser ce processus pour générer des données structurées.

Processus de Bruitage :

$$x_t = \sqrt{1 - \beta_t} x_{t-1} + \sqrt{\beta_t} \epsilon$$

- x_t : Données bruitées à l'étape t.
- β_t : Taux de bruit ajouté.
- ϵ : Bruit aléatoire gaussien.

Pourquoi les Modèles de Diffusion ?

Motivation :

- Les modèles génératifs doivent capturer des distributions complexes de données.
- Contrairement aux GAN, les modèles de diffusion sont stables à entraîner et n'ont pas de problèmes de collapse de mode.

Avantages :

- Fournissent un cadre robuste pour modéliser des distributions multimodales.
- Permettent un contrôle plus fin de la génération via le processus de débruitage.

Comparaison avec d'Autres Modèles Génératifs

Modèles de Diffusion vs GAN :

- **GAN** : Apprentissage par compétition (générateur vs discriminateur).
- **Modèles de Diffusion** : Apprentissage séquentiel du bruitage/débruitage.
- Pas de problème de convergence difficile pour les modèles de diffusion.

Modèles de Diffusion vs VAE :

- **VAE** : Modélisent les distributions avec des hypothèses gaussiennes.
- **Diffusion** : Plus de flexibilité dans la modélisation de la distribution.

Modélisation de la Distribution avec la Diffusion

Approche des Modèles de Diffusion :

- La distribution de données est modélisée comme une séquence d'états bruités.
- Chaque étape est paramétrée de manière à minimiser l'erreur entre les états bruités et les données originales.

Utilisation de la Probabilité :

- Les modèles de diffusion optimisent des densités de probabilité conditionnelles.

Processus de Bruitage et Débruitage

Processus de Bruitage :

- Ajout progressif de bruit gaussien aux données.
- Formule : $x_t = \sqrt{1 - \beta_t} x_{t-1} + \sqrt{\beta_t} \epsilon$

Processus de Débruitage :

- Reconstruction des données à partir du bruit, guidée par un modèle appris.

Fonction de Perte des Modèles de Diffusion

Objectif d'Apprentissage :

- Minimisation de l'écart entre le bruit ajouté et la prédiction du bruit par le modèle.

$$L = \mathbb{E}_{x_0,\epsilon,t}\left[\|\epsilon - \epsilon_\theta(x_t, t)\|^2\right]$$

Optimisation : Une approche basée sur la descente de gradient pour ajuster les paramètres du modèle.

Applications des Modèles de Diffusion en IA

Applications Pratiques :

- **Génération d'Images :** Modèles comme DALL-E et Stable Diffusion.
- **Synthèse Audio :** Amélioration de la qualité audio et de la clarté.
- **Remplissage de Données :** Inpainting pour compléter les images manquantes.

Potentiel Futur : Des avancées dans la génération de contenu multimodal.

Processus de Dégradation des Données

Principe de base :

- On ajoute progressivement du bruit à une donnée propre pour obtenir une version totalement bruitée.
- Objectif : étudier comment les données se détériorent en fonction du temps.

Équation de dégradation :

$$x_t = \sqrt{1 - \beta_t} x_{t-1} + \sqrt{\beta_t} \epsilon$$

- x_t : Données bruitées à l'étape t.
- β_t : Taux de bruit ajouté.
- ϵ : Bruit aléatoire gaussien.

Reconstruction des Données

Problème inverse :

- À partir des données totalement bruitées, on cherche à récupérer les données originales.
- Nécessite une approche d'estimation inverse pour débruiter progressivement.

Techniques utilisées :

- Algorithmes de débruitage progressif.
- Optimisation et apprentissage de la distribution des données.

Modélisation Simple : Distribution Univariée

Modèle de base : Considérons une distribution numérique simple.

- Exemple d'application :
  - Dégradation progressive d'une valeur numérique avec ajout de bruit.
  - Simulation de la diffusion et reconstruction.
- Visualisation :
  - Graphiques de la distribution initiale, bruitée et reconstruite.

Exemple : Processus de Diffusion en Machine Learning

Processus de Diffusion dans les Modèles Génératifs :

- Modèles de diffusion en apprentissage automatique : utiliser un processus de bruitage progressif et de reconstruction pour générer des données.
- Applications : génération d'images, synthèse audio, remplissage de données manquantes.

Principe Fondamental :

- Ajouter du bruit à des données jusqu'à obtenir une distribution proche d'un bruit aléatoire pur.
- Inverser le processus de diffusion pour générer des données nouvelles à partir de bruit.

Modèles de Diffusion et Bruitage Progressif

Étapes de Bruitage Progressif :

- On applique du bruit gaussien à chaque étape t selon l'équation :

$$x_t = \sqrt{1 - \beta_t} x_{t-1} + \sqrt{\beta_t} \epsilon$$

- x_t : Donnée bruitée au temps t, β_t : Taux de bruit, ϵ : Bruit gaussien.
- Plus t augmente, plus l'image se dégrade, convergeant vers une distribution de bruit aléatoire.

But : Apprendre une distribution inverse permettant de générer de nouvelles données en débruitant progressivement.

Exemple Pratique : Génération d'Images par Diffusion

Processus Inverse :

- À partir d'un bruit aléatoire x_T, appliquer des étapes de débruitage pour récupérer une image réaliste.
- Utilisation de réseaux de neurones pour apprendre cette "fonction de débruitage" progressive.

Architecture Typique :

- Modèles comme les U-Nets pour reconstruire les données à chaque étape.
- Utilisation de techniques de guidance (ex : guidance par classifier) pour améliorer la qualité des données générées.

Application : Génération d'images haute-résolution réalistes à partir de bruit.

Comparaison avec les GAN et autres Modèles Génératifs

Diffusion vs GAN :

- **GANs** : Utilisent un réseau de générateur et de discriminateur, souffrent souvent d'instabilité de l'entraînement.
- **Modèles de Diffusion** : Suivent un processus de débruitage bien défini et génèrent des données de haute qualité.
- Avantage : Pas de problème de collapse de mode, contrairement aux GAN.

Applications :

- Les modèles de diffusion montrent des performances compétitives pour la génération de données de haute fidélité.
- Utilisés pour des tâches sensibles à la diversité et au réalisme (ex : synthèse d'images médicales).

Équations Clés des Modèles de Diffusion

1. **Processus de Bruitage :**

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

2. **Processus de Débruitage (Reconstruction) :**

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

3. **Objectif d'Apprentissage :**

$$L = \mathbb{E}_{x_0,\epsilon,t}\left[\|\epsilon - \epsilon_\theta(x_t, t)\|^2\right]$$

où ϵ_θ est le réseau de prédiction de bruit.

Architecture Typique d'un Modèle de Diffusion

Composants Principaux :

- **Encodeur :** Transforme l'entrée bruitée en représentation latente.
- **U-Net :** Architecture principale pour la prédiction du bruit.
- **Décodeur :** Reconstruit l'image à partir de la représentation débruitée.

Particularités :

- Injection du pas de temps t dans le réseau.
- Utilisation de blocs résiduels et d'attention.

Applications Avancées des Modèles de Diffusion

1. **Inpainting d'Images :**
   - Reconstruction de parties manquantes d'une image.
   - Utilisation de masques pour guider la diffusion.
2. **Super-résolution :**
   - Amélioration de la résolution d'images basse qualité.
   - Processus de diffusion guidé par l'image basse résolution.
3. **Traduction Image-à-Image :**
   - Transformation d'un style d'image à un autre.
   - Exemple : Conversion photo en peinture.
4. **Génération de Texte-à-Image :**
   - Création d'images à partir de descriptions textuelles.
   - Modèles comme DALL-E 2 et Stable Diffusion.

Défis et Perspectives Futures

Défis Actuels :

- Temps de génération relativement long.
- Besoin de ressources computationnelles importantes.
- Difficulté à contrôler précisément la sortie.

Perspectives :

- Accélération des processus de génération (ex : DDIM).
- Amélioration du contrôle et de la guidance.
- Intégration avec d'autres techniques d'IA (RL, NLP).

Domaines Prometteurs :

- Création de contenu multimodal.
- Applications en sciences (chimie, biologie).
- Génération de données synthétiques pour l'entraînement d'IA.

Conclusion et Questions

Résumé :

- Les modèles de diffusion sont des outils puissants pour la modélisation des distributions de données.
- Les EDS et les processus de diffusion sont fondamentaux pour comprendre ces modèles.

Prochaines étapes :

- Approfondissement des techniques de reconstruction.
- Introduction aux modèles de diffusion appliqués aux images et aux données complexes.

Discussion : Questions ou points d'éclaircissement ?

Code: Modèle de Diffusion

Définition du Modèle de Diffusion :

```
import torch
import torch.nn as nn

class DiffusionModel(nn.Module):
    def __init__(self, input_dim, hidden_dim):
        super().__init__()
        self.fc1 = nn.Linear(input_dim, hidden_dim)
        self.fc2 = nn.Linear(hidden_dim, input_dim)

    def forward(self, x, t):
        # Ajout d'informations temporelles t
        x_t = torch.cat([x, t], dim=-1)
        h = torch.relu(self.fc1(x_t))
        return self.fc2(h)
```

Code: Modèle de Diffusion

Processus d'Entraînement :

```
diffusion_model = DiffusionModel(input_dim=100,
                                 hidden_dim=128)
optimizer = optim.Adam(diffusion_model.parameters(),
                       lr=1e-4)

def diffusion_loss(x, x_noisy, noise_pred):
    return nn.MSELoss()(noise_pred, x_noisy - x)

for epoch in range(epochs):
    for x in dataloader:
        t = torch.randint(0, T, (x.size(0),))  # Temps aléatoire
        noise = torch.randn_like(x)
        x_noisy = x + noise * torch.sqrt(t / T)

        optimizer.zero_grad()
        noise_pred = diffusion_model(x, t)
        loss = diffusion_loss(x, x_noisy, noise_pred)
        loss.backward()
        optimizer.step()
```

Défis de l'IA Générative

Stabilité de l'Entraînement :

- Les modèles génératifs peuvent être difficiles à entraîner et à converger.
- Nécessité de techniques avancées pour stabiliser l'entraînement.

Diversité des Données Générées :

- Risque de collapse de mode, où le modèle génère des échantillons avec peu de diversité.
- Nécessité de mécanismes pour encourager la diversité.

Qualité des Données Générées :

- Les données générées doivent être réalistes et de haute qualité.
- Nécessité de métriques pour évaluer la qualité des données générées.

Conclusion

Résumé :

- Les modèles génératifs, tels que les VAE, GAN et modèles de diffusion, sont essentiels pour la création de données réalistes.
- Chaque modèle présente des avantages et des défis spécifiques en termes de stabilité, diversité et qualité des données générées.
- Les modèles de diffusion offrent une approche robuste et flexible pour la génération de données de haute qualité.

Perspectives Futures :

- Amélioration des techniques d'entraînement et de contrôle pour les modèles génératifs.
- Intégration des modèles génératifs dans divers domaines d'application, y compris l'art, la médecine et la synthèse vocale.
- Exploration de nouvelles architectures et méthodes pour surmonter les défis actuels.

Questions :

- Y a-t-il des questions ou des points à éclaircir sur les modèles génératifs et les processus de diffusion ?