Traitement et Analyse d'Images

Modèles génératifs
Modèles de diffusion

olivier.bernard@insa-lyon.fr

Quel est l'intérêt des modèles de diffusion ?

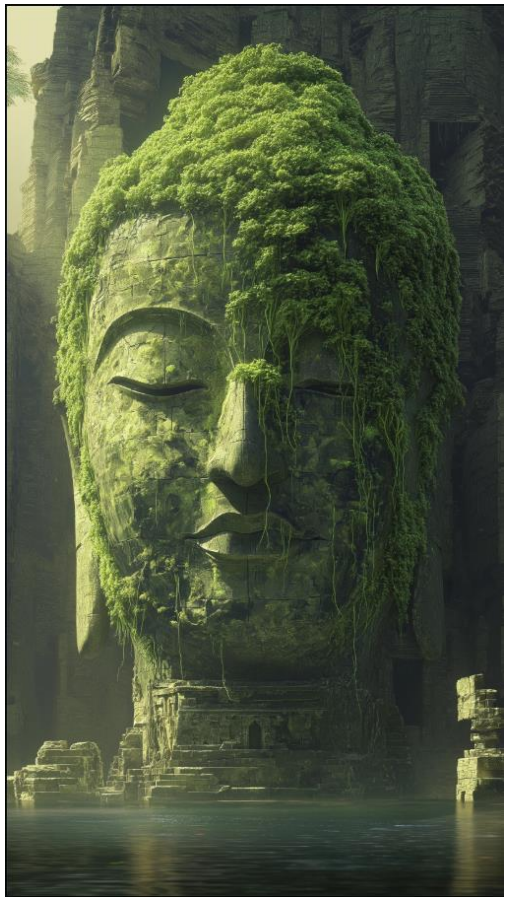
- Meilleurs méthodes actuelles pour la génération d'images synthétiques
- Permets de générer des images sous formes conditionnées
- Beaucoup de solutions logiciels, comme Midjourney, dall-E

An Asian girl in ancient coarse linen clothes rides a giant panda and carries a wooden cage. A chubby little girl with two buns walks on the snow. High-precision clothing texture, real tactile skin, foggy white tone, low saturation, retro film texture, tranquil atmosphere, minimalism, long-range view, telephoto lens

Quel est l'intérêt des modèles de diffusion ?

- Meilleurs méthodes actuelles pour la génération d'images synthétiques
- Permets de générer des images sous formes conditionnées
- Beaucoup de solutions logiciels, comme Midjourney, dall-E

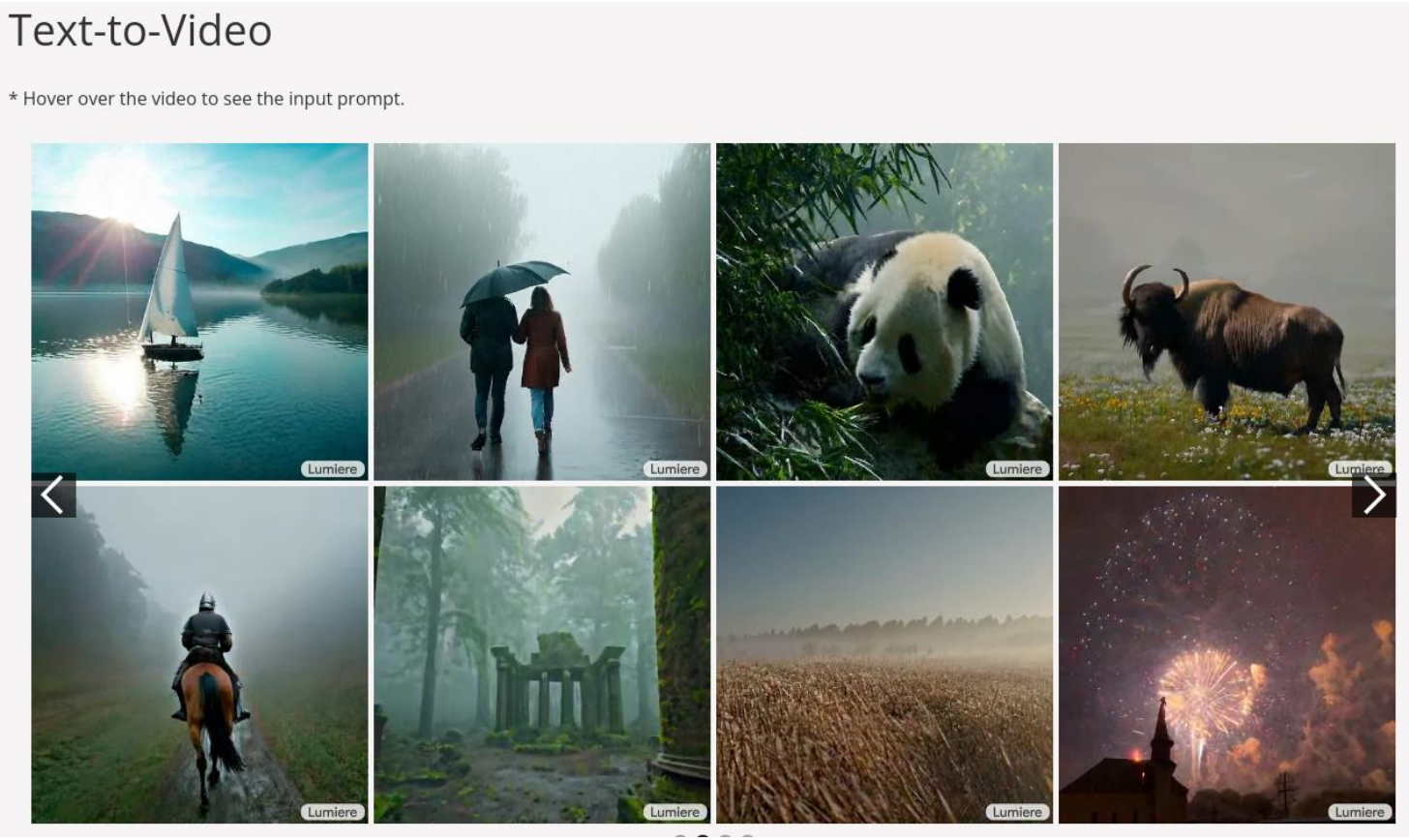
A digital artwork depicting the Buddha's head, intricately designed with green trees growing from it and vines surrounding its face. The background is an enchanted forest filled with ancient ruins, creating a mystical atmosphere. In front of the Buddha's head lies a tranquil river that reflects his serene expression. This scene embodies peace amidst chaos in nature



Quel est l'intérêt des modèles de diffusion ?

- Extensions récentes pour la synthèse de vidéo

https://lumiere-video.github.io/#section_image_to_video



Quel est l'intérêt des modèles de diffusion ?

- Extensions récentes pour la synthèse de vidéo

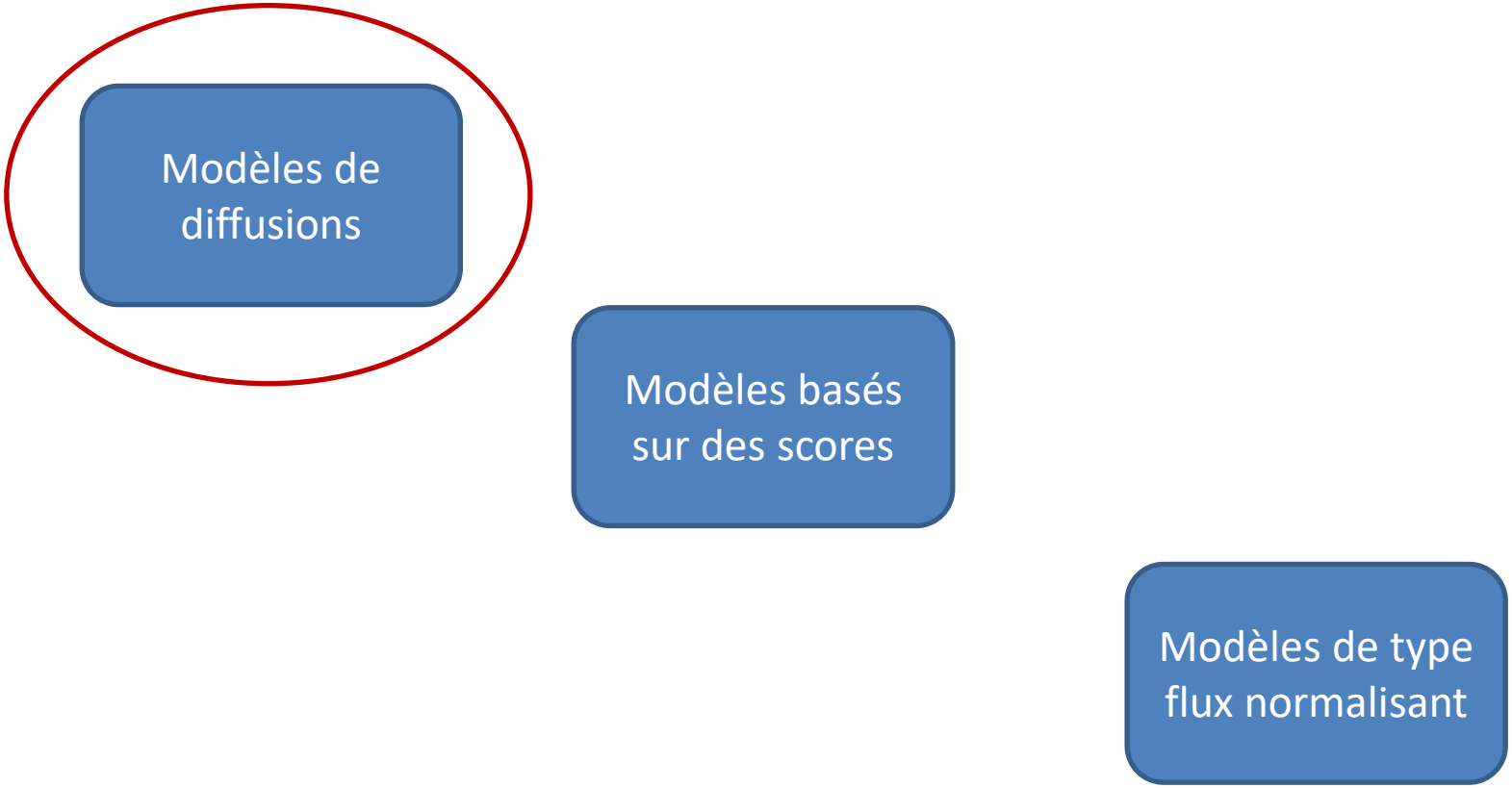
https://lumiere-video.github.io/#section_image_to_video

Image-to-Video

* Hover over the video to see the input image and prompt.

Quel est l'intérêt des modèles de diffusion ?

- Famille des réseaux de diffusions



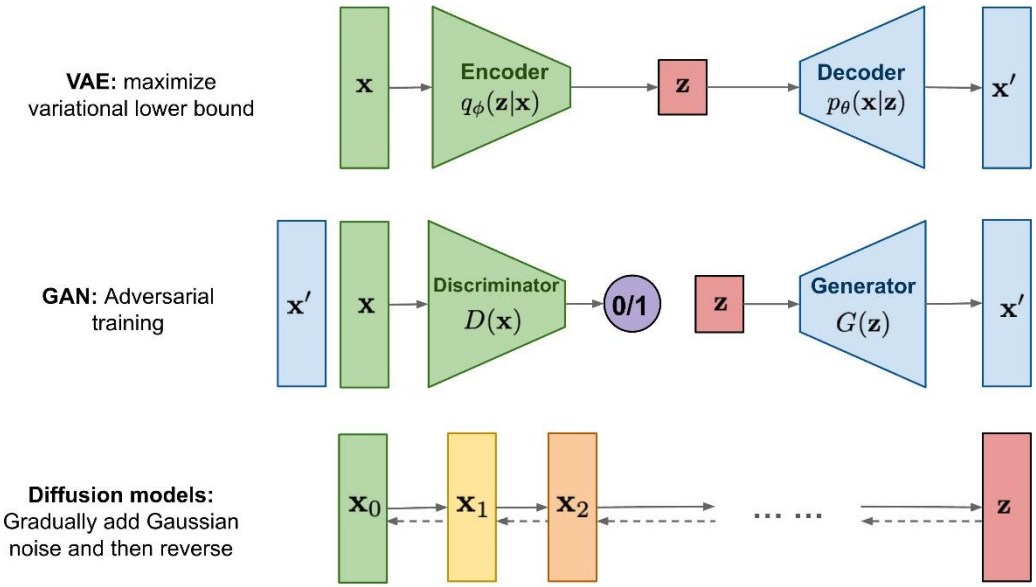
Modèles de diffusion probabiliste avec débruitage

DDPM

L'ensemble des mathématiques sont décrites dans le blog suivant

https://creatis-myriad.github.io/tutorials/2023-11-30-tutorial-ddpm.html

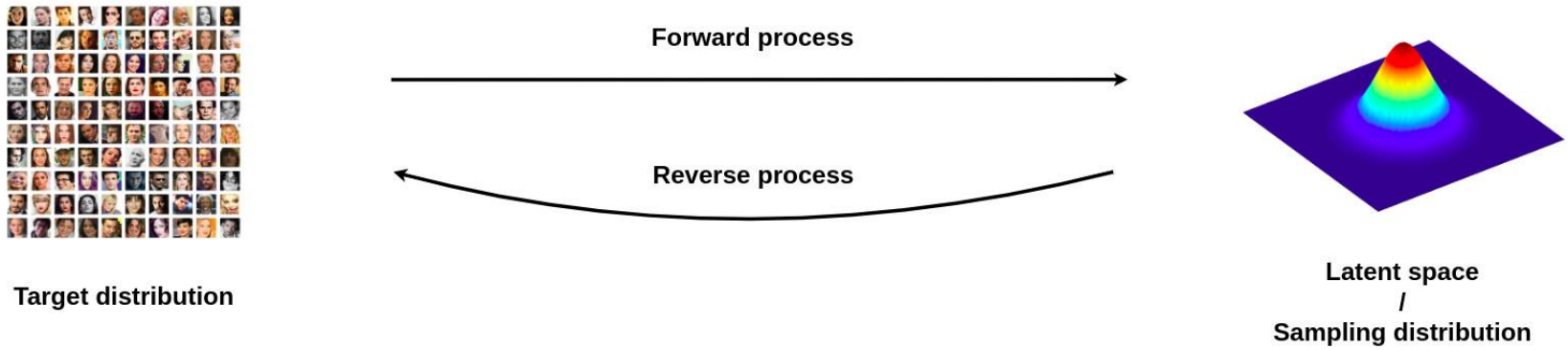
- Principales caractéristiques
  - Appartient à la famille des modèles génératifs (comme les VAE)

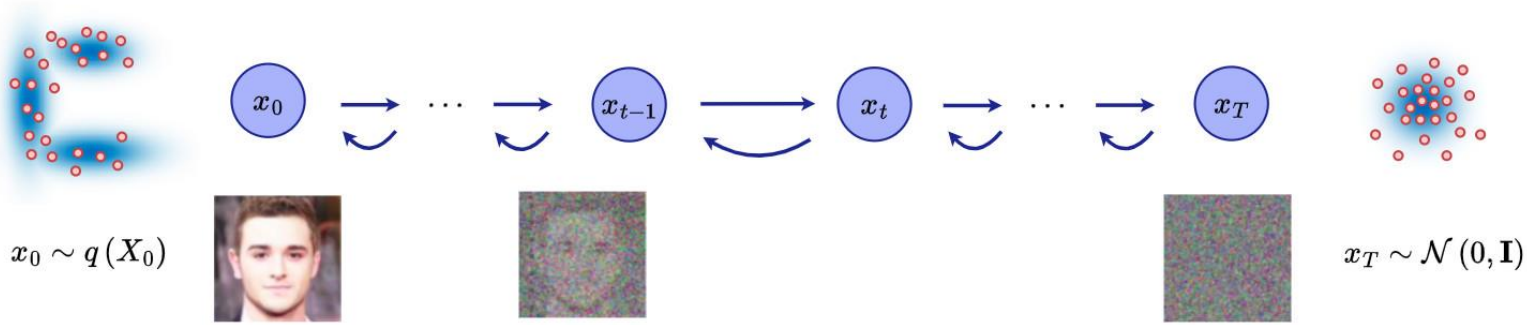


  - Basé sur la notion de chaîne de Markov

Modèle mathématique utilisé pour décrire un système qui évolue de manière aléatoire entre différents états, selon certaines règles de probabilité

- Principales caractéristiques
  - ➔ Définis une chaîne de Markov d'étapes de diffusion pour ajouter lentement un bruit aléatoire aux données
  - ➔ Le modèle apprend ensuite à inverser le processus de diffusion pour construire des échantillons de données à partir du bruit.





Outils mathématiques utilisés

- Théorème de Bayes

$$q(x_t \mid x_{t-1}) = \frac{q(x_{t-1} \mid x_t)\, q(x_t)}{q(x_{t-1})}$$

$$q(x_{t-1} \mid x_t) = \frac{q(x_t \mid x_{t-1})\, q(x_{t-1})}{q(x_t)}$$

- Théorème marginal

$$q(x_0, x_1, \cdots, x_T) = q(x_{0:T})$$

$$q(x_0) = \int q(x_0, x_1, \cdots, x_T)\, dx_1 \cdots dx_T$$

$$q(x_0) = \int q(x_{0:T})\, dx_{1:T}$$

- Théorème des probabilités conditionnelles

$$q(x_{t-1}, x_t) = q(x_t \mid x_{t-1}) q(x_{t-1})$$

$$q(x_{1:T} \mid x_0) = q(x_T \mid x_{0:T-1}) q(x_{T-1} \mid x_{0:T-2}) \ldots q(x_1 \mid x_0)$$

Propriété des chaînes de Markov

La probabilité de chaque événement ne dépend que de l'état atteint lors de l'événement précédent

- Théorème des probabilités conditionnelles

$$q(x_T \mid x_{0:T-1}) = q(x_T \mid x_{T-1})$$

$$q(x_{1:T} \mid x_0) = \prod_{t=1}^{T} q(x_t \mid x_{t-1})$$

- Théorème de Bayes

$$q(x_t \mid x_{t-1}) = q(x_t \mid x_{t-1}, x_0) = \frac{q(x_{t-1} \mid x_t, x_0)\, q(x_t \mid x_0)}{q(x_{t-1} \mid x_0)}$$

- Distribution conjointe

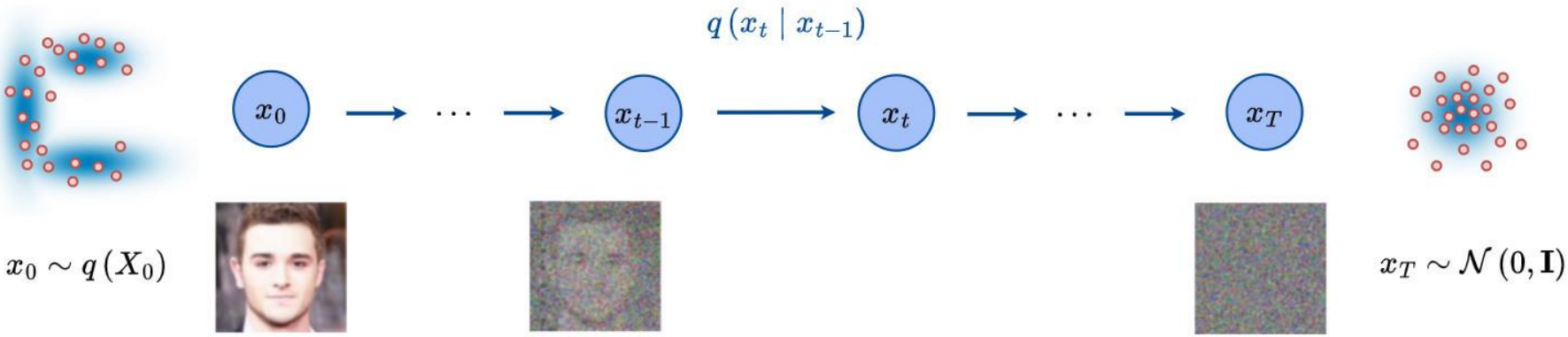
$$p_\theta(x_{0:T}) = p_\theta(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1} \mid x_t)$$

DDPM

Processus de diffusion vers l’avant

Processus de diffusion vers l'avant (forward diffusion process)

Procédure dans laquelle une petite quantité de bruit gaussien est ajoutée à l'échantillon ponctuel x_0, produisant une séquence d'échantillons bruités $x_1, \cdots, x_T$



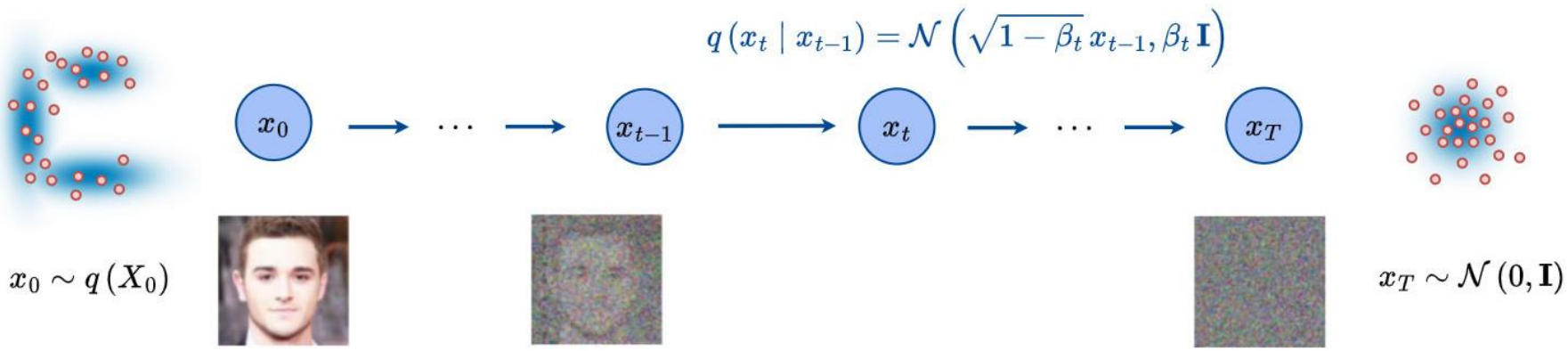
- x_0 est un échantillon issu d'une distribution de données réelles $x_0 \sim q(X_0)$
- $q(x_t|x_{t-1})$ modélise la probabilité d'avoir l'état x_t sachant l'état x_{t-1}

Processus de diffusion vers l'avant (forward diffusion process)

Le processus de propagation avant d'un DDPM est une chaîne de Markov

- La prédiction à l'étape t ne dépend que de l'état à l'étape $t-1$, qui ajoute progressivement un bruit gaussien aux données x_0
- Le processus complet est modélisé par: $q(x_{1:T} \mid x_0) = \prod_{t=1}^{T} q(x_t \mid x_{t-1})$
- La probabilité conditionnelle peut être modélisée efficacement par

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



Processus de diffusion vers l'avant (forward diffusion process)

- Comment définir la variance β_t ?

  - ➔ $\{\beta_t \in (0,1)\}_{t=1}^{T}$ séquence de constantes linéairement croissantes
  - ➔ $\beta_t = clip\left(1 - \frac{\bar{\alpha}_t}{\bar{\alpha}_{t-1}}, 0.999\right)$ séquence de constantes de type cosinus

avec $\bar{\alpha}_t = \frac{f(t)}{f(0)}$ et $f(t) = cos\left(\frac{\frac{t}{T} + s}{1+s} \cdot \frac{\pi}{2}\right)^2$

  - ➔ Dans ce cas

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

$$\begin{array}{llll} \text{Si} & \beta_t = 0, & \text{alors} & q(x_t \mid x_{t-1}) = x_{t-1} \\ \text{Si} & \beta_t = 1, & \text{alors} & q(x_t \mid x_{t-1}) = \mathcal{N}(0, \mathbf{I}) \end{array}$$

Processus de diffusion vers l'avant (forward diffusion process)

- Probabilité conditionnelle: relation importante

  - En utilisant l'astuce de reparamétrisation

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

$$x_t = \sqrt{1-\beta_t}\, x_{t-1} + \sqrt{\beta_t}\, \epsilon_{t-1} \qquad \text{avec} \qquad \epsilon_{t-1} = \mathcal{N}(0, \mathbf{I})$$

  - On peut démontrer la relation suivante

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \epsilon_t$$

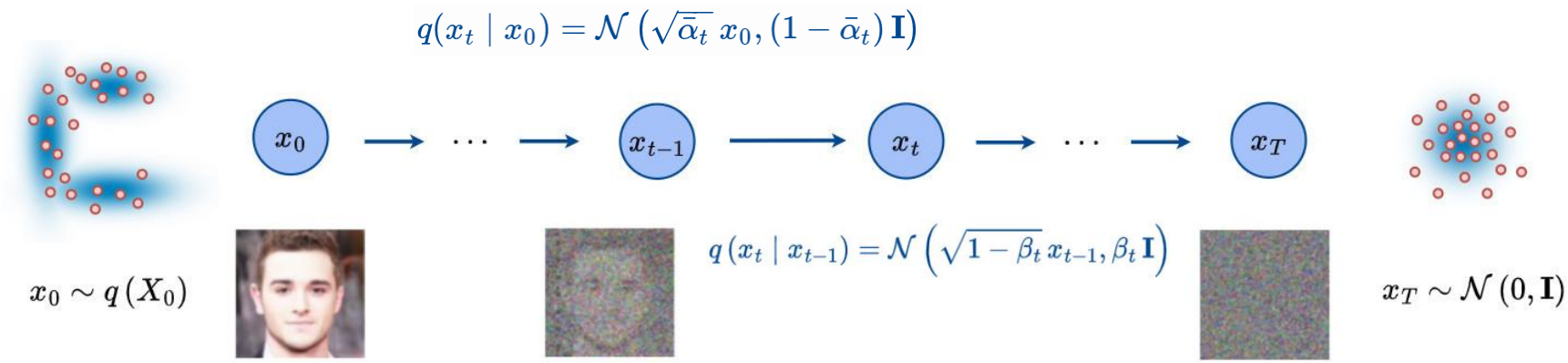
$$\boxed{q(x_t \mid x_0) = \mathcal{N}\left(\sqrt{\bar{\alpha}_t}\, x_0, (1-\bar{\alpha}_t)\, \mathbf{I}\right)}$$

avec $\alpha_t = 1 - \beta_t$

$$\bar{\alpha}_t = \prod_{k=1}^{t} \alpha_k$$

Processus de diffusion vers l'avant (forward diffusion process)

- Pour résumer



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

$$\begin{array}{llll} \text{Si} & \beta_t = 0, & \text{alors} & q(x_t \mid x_{t-1}) = x_{t-1} \\ \text{Si} & \beta_t = 1, & \text{alors} & q(x_t \mid x_{t-1}) = \mathcal{N}(0, \mathbf{I}) \end{array}$$

$$q(x_t \mid x_0) = \mathcal{N}\left(\sqrt{\bar{\alpha}_t}\, x_0, (1-\bar{\alpha}_t)\,\mathbf{I}\right)$$

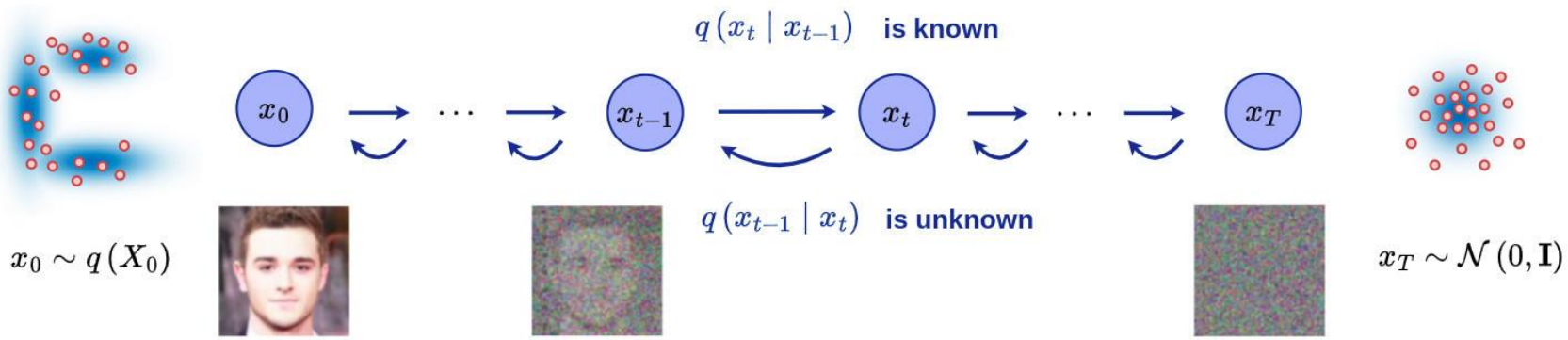
avec $\alpha_t = 1 - \beta_t$ et $\bar{\alpha}_t = \prod_{k=1}^{t} \alpha_k$

DDPM

Processus inverse

Processus inverse

Si on est capable d'inverser le processus de diffusion à partir de $q(x_{t-1}|x_t)$, alors on pourra générer un échantillon x_0 à partir d'un bruit gaussien $x_T \sim \mathrm{N}(0, \boldsymbol{I})$



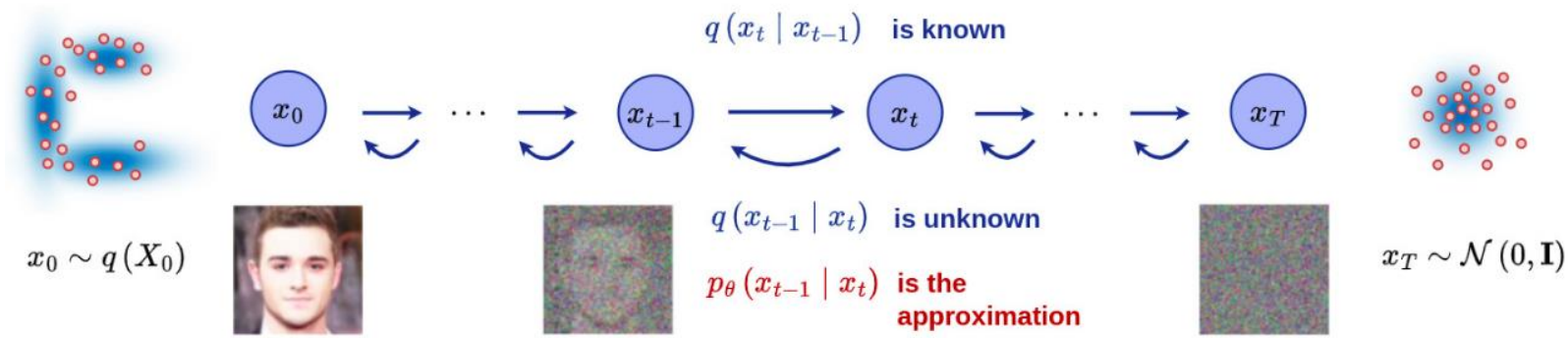
- Grâce au théorème de Bayes

$$q(x_{t-1} \mid x_t) = \frac{q(x_t \mid x_{t-1})\, q(x_{t-1})}{q(x_t)}$$

- Comme $q(x_t)$ n'est non connu, $q(x_{t-1}|x_t)$ est insoluble

Processus inverse

On va apprendre un modèle $p_\theta(x_{t-1}|x_t)$ pour approximer $q(x_{t-1}|x_t)$ afin d'exécuter le processus de diffusion inverse

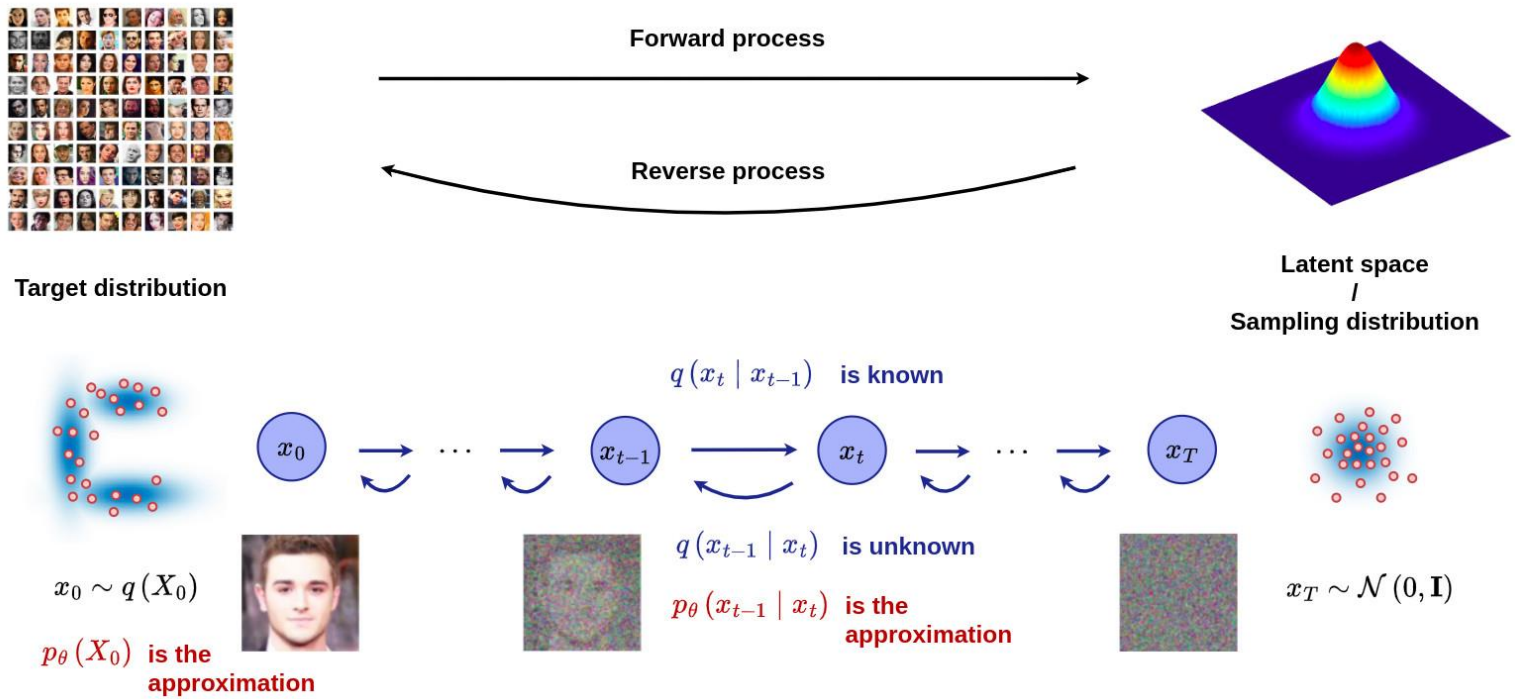


- Hypothèse gaussienne $p_\theta(x_{t-1} \mid x_t) = \mathcal{N}(\mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$
- Modélisation de l'ensemble du processus inverse

$$p_\theta(x_{0:T}) = p_\theta(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1} \mid x_t)$$

Processus inverse

- Pour résumer



➔ Modèle à apprendre

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

➔ Ensemble du processus inverse

$$p_\theta(x_{0:T}) = p_\theta(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1} \mid x_t)$$

DDPM

Stratégie d’apprentissage

Stratégie d'apprentissage

- Minimiser l'entropie croisée entre $q(x_0)$ et $p_\theta(x_0)$ aboutit à ce que les deux distributions soient aussi proches que possible

$$H(q, p_\theta) = -\int q(x_0) \cdot \log(p_\theta(x_0))\, dx_0 = -\mathbb{E}_{x_0 \sim q}\left[\log(p_\theta(x_0))\right]$$

- Réécriture de cette expression via le théorème marginal

$$\begin{aligned} H(q, p_\theta) &= -\mathbb{E}_{x_0 \sim q}\left[\log\left(\int p_\theta(x_{0:T})\, d_{x_{1:T}}\right)\right] \\ &= -\mathbb{E}_{x_0 \sim q}\left[\log\left(\int q(x_{1:T} \mid x_0) \frac{p_\theta(x_{0:T})}{q(x_{1:T} \mid x_0)}\, d_{x_{1:T}}\right)\right] \\ &= -\mathbb{E}_{x_0 \sim q}\left[\log\left(\mathbb{E}_{x_{1:T} \sim q(x_{1:T} \mid x_0)}\left[\frac{p_\theta(x_{0:T})}{q(x_{1:T} \mid x_0)}\right)\right]\right] \end{aligned}$$

Stratégie d'apprentissage

- Inégalité de Jensen

$$\phi(\mathbb{E}[X]) \leq \mathbb{E}[\phi(X)]$$

$$\begin{aligned}
\Rightarrow \quad H(q, p_\theta) &\leq -\mathbb{E}_{x_0 \sim q}\, \mathbb{E}_{x_{1:T} \sim q(x_{1:T}|x_0)} \left[\log \left(\frac{p_\theta(x_{0:T})}{q(x_{1:T} \mid x_0)} \right) \right] \\
&\leq -\mathbb{E}_{x_{0:T} \sim q(x_{0:T})} \left[\log \left(\frac{p_\theta(x_{0:T})}{q(x_{1:T} \mid x_0)} \right) \right] \\
&\leq \mathbb{E}_{x_{0:T} \sim q(x_{0:T})} \left[\log \left(\frac{q(x_{1:T} \mid x_0)}{p_\theta(x_{0:T})} \right) \right] \\
&\leq \mathcal{L}_{VUB}
\end{aligned}$$

- Limite supérieure variationnelle

$$\boxed{\mathcal{L}_{VUB} = \mathbb{E}_{x_{0:T} \sim q(x_{0:T})} \left[\log \left(\frac{q(x_{1:T} \mid x_0)}{p_\theta(x_{0:T})} \right) \right]}$$

Comme $H(q, p_\theta)$ est positif, minimiser $\mathcal{L}_{VUB}$ revient à minimiser $H(q, p_\theta)$

- Minimisation de $\mathcal{L}_{VUB}$

$$\begin{aligned}\mathcal{L}_{VUB} &= \mathbb{E}_{x_{0:T}\sim q}\left[\log\left(\frac{q(x_T \mid x_0)}{p_\theta(x_T)}\right) + \sum_{t=2}^{T}\log\left(\frac{q(x_{t-1} \mid x_t, x_0)}{p_\theta(x_{t-1} \mid x_t)}\right) - \log(p_\theta(x_0 \mid x_1))\right] \\ &= \underbrace{D_{KL}\left(q(x_T \mid x_0) \parallel p_\theta(x_T)\right)}_{\mathcal{L}_T} + \sum_{t=2}^{T}\underbrace{D_{KL}\left(q(x_{t-1} \mid x_t, x_0) \parallel p_\theta(x_{t-1} \mid x_t)\right)}_{\mathcal{L}_{t-1}} - \underbrace{\log(p_\theta(x_0 \mid x_1))}_{\mathcal{L}_0}\end{aligned}$$

L'obtention de cette expression est décrit dans le blog suivant

https://creatis-myriad.github.io/tutorials/2023-11-30-tutorial-ddpm.html

- Minimisation de $\mathcal{L}_{VUB}$
  - <u>Remarque n°1:</u> étant donné que la séquence $\{\beta_t\}_{t\in[1,T]}$ est choisie en amont, $q(x_T|x_0)$ est déterministe et $\mathcal{L}_T$ est un terme constant dont on ne tiendra pas compte dans le processus de minimisation
  - <u>Remarque n°2:</u> L_0 peut être modélisé par un décodeur particulier, ou être omis dans un but de simplification
  - <u>Remarque n°3:</u> en utilisant l'astuce de reparamétrisation, $q(x_{t-1}|x_t, x_0)$ peut être reformulé comme

$$\boxed{q(x_{t-1} \mid x_t, x_0) = \mathcal{N}(\tilde{\mu}_t(x_t, x_0), \tilde{\beta}_t \cdot \mathbf{I})}$$

avec

$$\tilde{\mu}_t(x_t, x_0) = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_t)$$

$$\tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t} \cdot \beta_t \qquad \bar{\alpha}_t = \prod_{k=1}^{t} \alpha_k \qquad \alpha_t = 1 - \beta_t$$

- Minimisation de $\mathcal{L}_{VUB}$

  - La minimisation de $\mathcal{L}_{VUB}$ revient donc à minimiser $D_{KL}(q(x_{t-1}|x_t, x_0) \,||\, p_\theta(x_{t-1}|x_t))$ pour tout instant t

avec
$$\begin{cases} q(x_{t-1} \mid x_t, x_0) = \mathcal{N}(\tilde{\mu}_t(x_t, x_0), \tilde{\beta}_t \cdot \mathbf{I}) \\ p_\theta(x_{t-1} \mid x_t) = \mathcal{N}(\mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \end{cases}$$

  - On souhaite rendre les deux distributions gaussiennes $q(x_{t-1}|x_t, x_0)$ et $p_\theta(x_{t-1}|x_t)$ les plus proches possible

  - Dans un soucis de simplification, on choisit $\Sigma_\theta(x_t, t) = \sigma_t \mathbf{I} = \tilde{\beta}_t \mathbf{I}$

L'idée est donc de se focaliser sur les moyennes des deux distributions et d'entrainer un réseau de neurones μ_θ à prédire $\tilde{\mu}_t = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_t\right)$

Stratégie d'apprentissage

Le terme de perte $\mathcal{L}_{t-1}$ est revisité pour minimiser la différence entre μ_θ et $\tilde{\mu}$

$$\mathcal{L}_{t-1} = \mathbb{E}_{x_0 \sim q, \epsilon \sim \mathcal{N}} \left[\frac{(1-\alpha_t)^2}{2\alpha_t(1-\bar{\alpha}_t)\bar{\beta}_t^{\,2}} \|\epsilon_t - \epsilon_\theta(x_t, t)\|^2 \right]$$

➔ Cette expression peut être simplifiée en ignorant le terme de pondération, ce qui donne la fonction de perte à minimiser finale suivante:

$$\mathcal{L}_{t-1}^{simple} = \mathbb{E}_{x_0 \sim q, \epsilon \sim \mathcal{N}, t \sim [1,T]} \left[\|\epsilon_t - \epsilon_\theta(x_t, t)\|^2 \right]$$

DDPM

Architecture

- Points clés

  - Le but est d'estimer la probabilité conditionnelle $p_\theta(x_{t-1}|x_t)$

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

$$\begin{cases} \mu_\theta(x_t,t) = \dfrac{1}{\sqrt{\alpha_t}}\left(x_t - \dfrac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t,t)\right) \\ \Sigma_\theta(x_t,t) = \sigma_t \mathbf{I} = \tilde{\beta}_t \mathbf{I} \end{cases}$$

  - Même si la modélisation clé des modèles de diffusion est la chaîne de Markov, il est possible d'exprimer directement x_t en fonction de x_0

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \epsilon_t$$

$$\begin{cases} \alpha_t = 1 - \beta_t \\ \bar{\alpha}_t = \prod_{k=1}^{t} \alpha_k \end{cases} \quad \text{et} \quad \epsilon_t = \mathcal{N}(0, \mathbf{I})$$

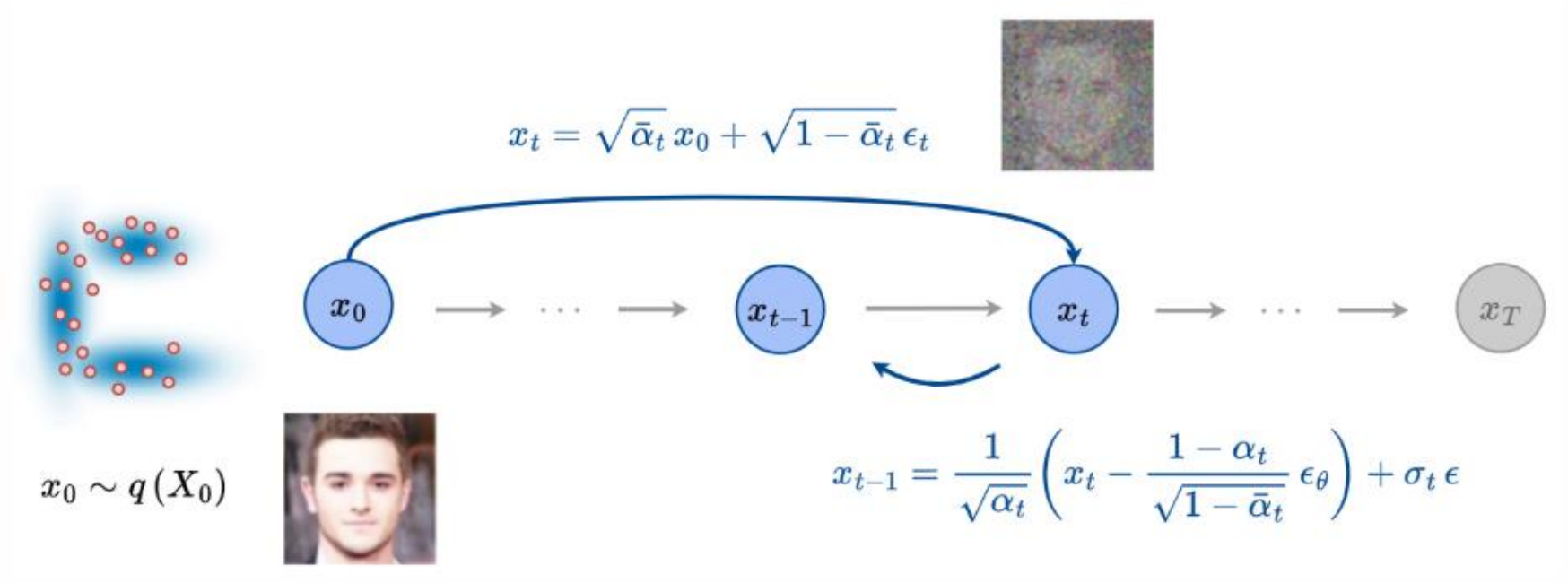
  - La seule inconnue est le bruit $\epsilon_\theta(x_t,t)$ que nous estimerons par un réseau de neurone via la minimisation de la fonction de perte suivante

$$\mathcal{L}_{t-1}^{simple} = \mathbb{E}_{x_0 \sim q, \epsilon \sim \mathcal{N}, t \sim [1,T]}\left[\|\epsilon_t - \epsilon_\theta(x_t,t)\|^2\right]$$

Architecture

Il est donc possible, à tout moment t, de générer une image bruitée x_t à partir de x_0 et ϵ_t, qui sont connus, et d'apprendre à estimer ϵ_t à partir de x_t

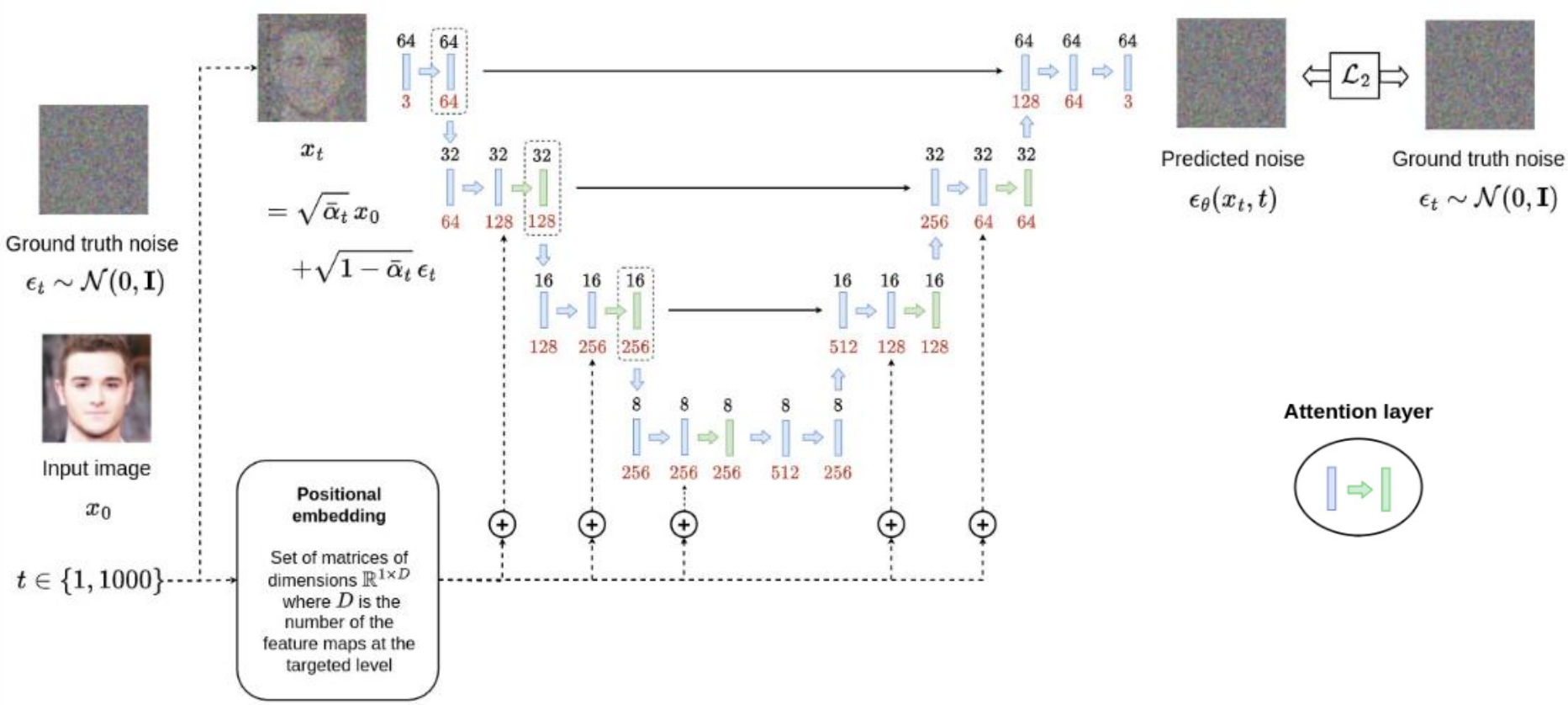
Le bruit estimé $\epsilon_\theta(x_t, t)$ peut alors être utilisé pour retrouver x_{t-1} à partir de x_t



Architecture

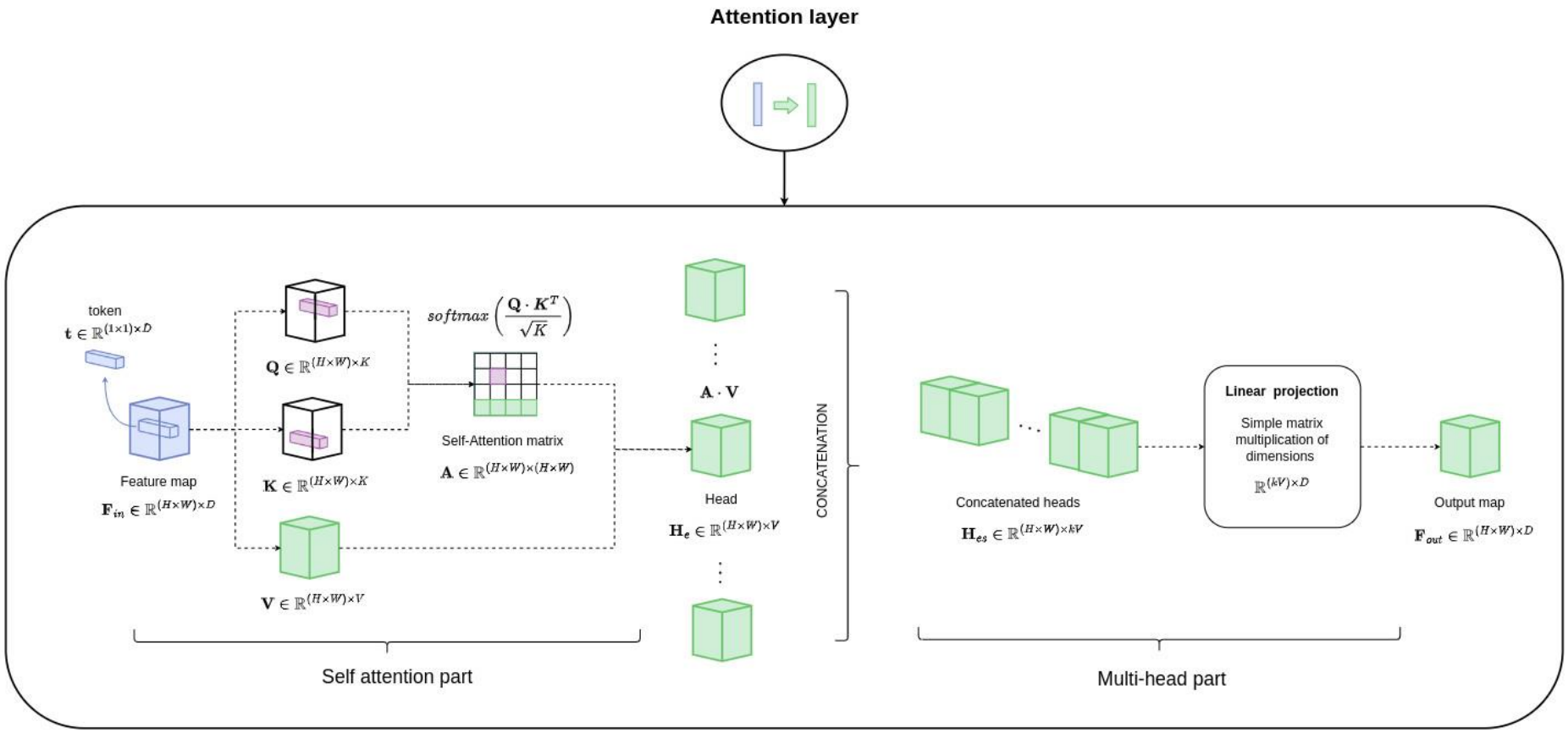
U-Net standard avec des couches d'attention et un encodage de position pour intégrer l'information temporelle

➔ Intégration de t est nécessaire car le bruit ajouté varie au cours du temps



Architecture

➔ Couches d'attention



- En résumer

  - En entrainement

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \mathrm{Uniform}(\{1, \ldots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$
6: **until** converged

  - En inférence / génération de nouvelle image synthétique

Algorithm 2 Sampling

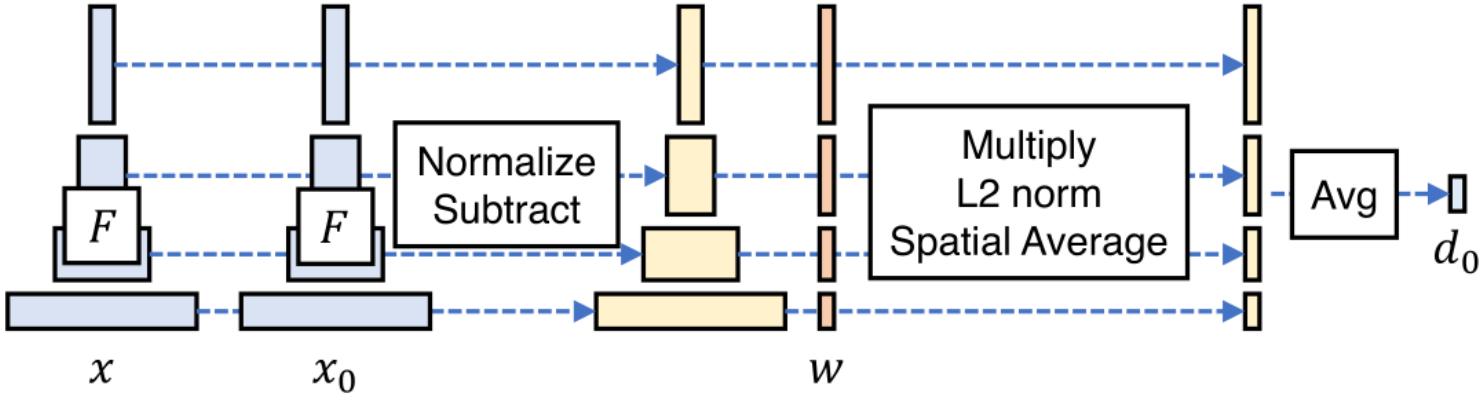
1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \ldots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t,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

Application pratique

Modèles de diffusion latente

Modèles de diffusion latente (LDM)

- Projection des images dans un espace dédié avant traitement
  - ➔ Utilisation d'un VAE en entrée/sortie du DDPM afin de réduire la complexité des images traitées et l'empreinte mémoire
  - ➔ Introduction d'une fonction de perte perceptuelle pour améliorer la qualité des images reconstruites

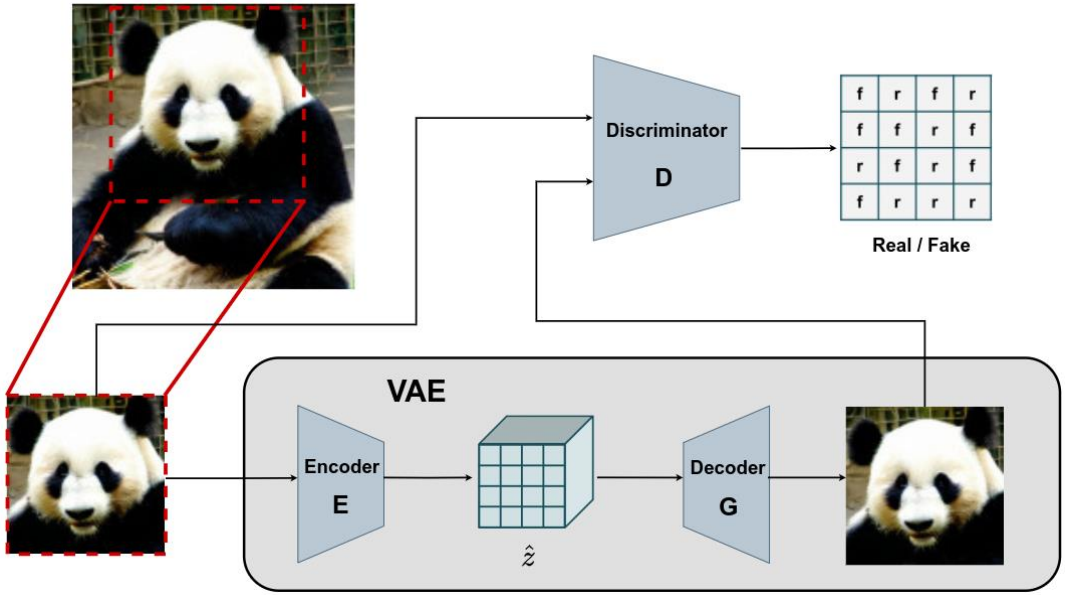


x et x_0 sont deux patchs « image » donnés en entrée

F est un réseau pré-entrainer, tel que VGG50

- Projection des images dans un espace dédié avant traitement

  - Implémentation d'une approche adversaire



  - Fonction de perte finale

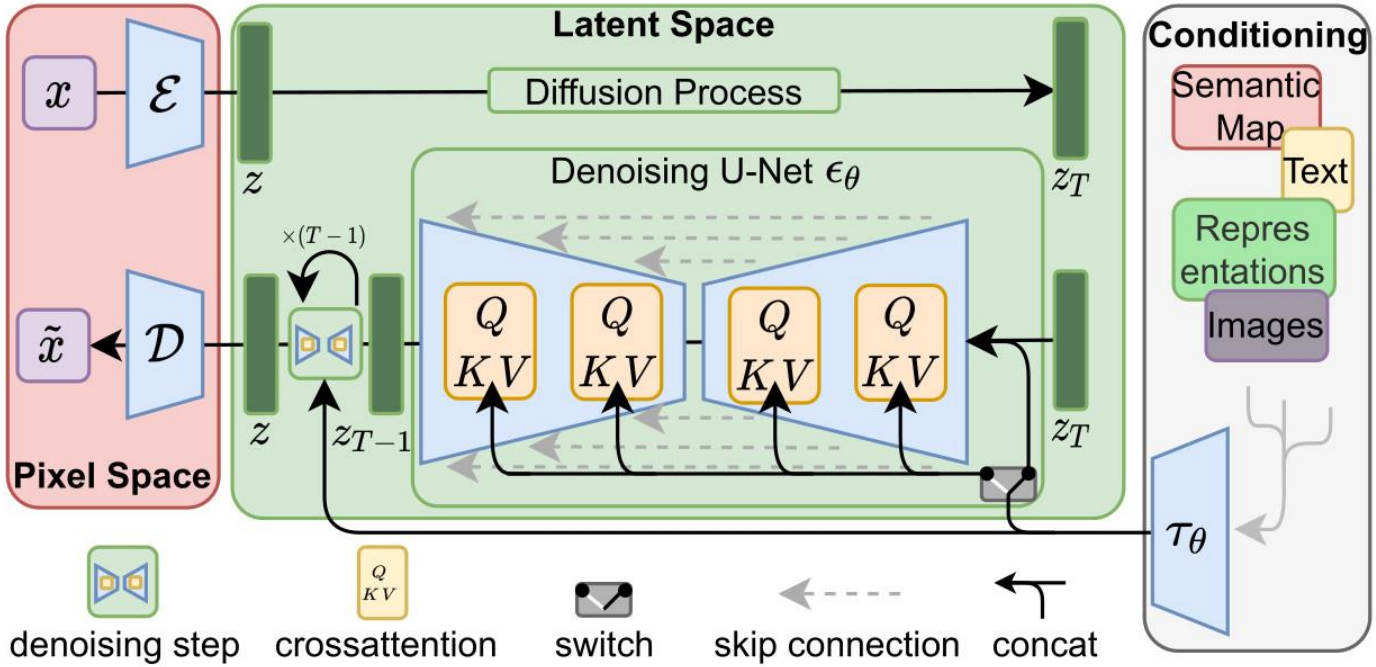
$$\mathcal{L} = \mathcal{L}_{recons} + \beta_1 \mathcal{L}_{KLD} + \beta_2 \mathcal{L}_{perceptual} + \beta_3 \mathcal{L}_{adversarial}$$

Modèles de diffusion latente (LDM)

- Le VAE est appris indépendamment du DDPM et son architecture est figée
- Minimisation de la fonction de perte suivante

$$\mathcal{L}_{LDM} = \mathbb{E}_{x_0 \sim q, \epsilon \sim \mathcal{N}, t \sim [1,T]} \left[\|\epsilon_t - \epsilon_\theta(x_t, t)\|^2 \right]$$

- Architecture LDM



- Caractéristiques

Parameters	LDM – 256×256
z-shape	$64 \times 64 \times 3$
Diffusion steps	1000
Noise scheduler	linear
Number of parameters	274 Million
Channels	224
Channel multiplier	$1, 2, 3, 4$
Attention resolutions	$32, 16, 8$
Number of head	1
Batch size	48
Iterations	410 k
Learning rate	$9.6\ e^{-5}$

Modèles de diffusion latente (LDM)

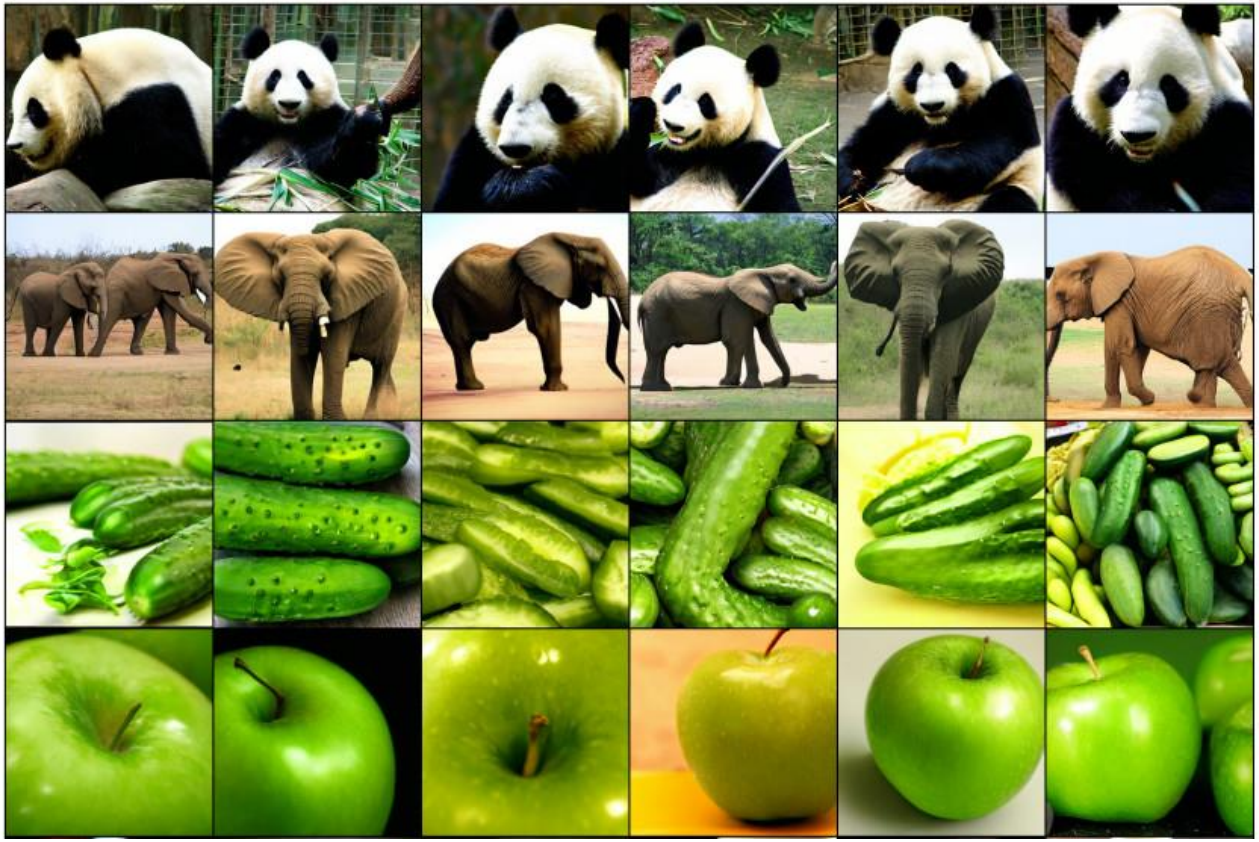
- Génération aléatoire d'images synthétiques *sans conditionnement* appris à partir de la base de données CelebA-HQ

Random samples on the CelebA-HQ dataset

Modèles de diffusion latente (LDM)

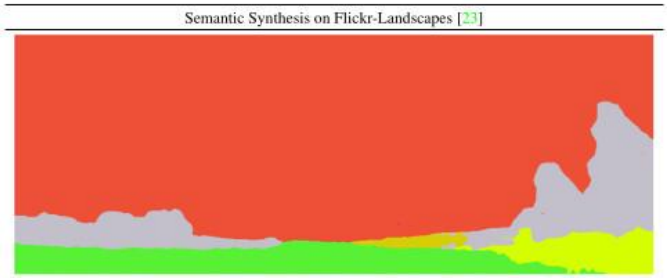
- Génération aléatoire d'images synthétiques *avec conditionnement sur la classe* appris à partir de la base de données ImageNet

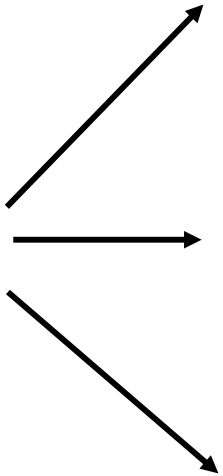
Random class conditional samples on the ImageNet dataset



Modèles de diffusion latente (LDM)

- Génération aléatoire d'images synthétiques *avec conditionnement sur des masques* appris à partir de la base de données Flickr-landscapes





Modèles de diffusion latente (LDM)

- Génération aléatoire d'images synthétiques *avec conditionnement sur le texte* appris à partir de la base de données LAION-400M
  - ➔ Utilisation du tokenizer BERT
  - ➔ Ce modèle possède plus de 1.45 Milliards de paramètres !

'A painting of the last supper by Picasso.'

That’s all folks