

Modelos Generativos Profundos para Imágenes

Modelos de difusión

Pablo Musé

pmuse@fing.edu.uy

Instituto de Ingeniería Eléctrica
Facultad de Ingeniería



Agenda

① Repaso: modelado basado en función de *score*

② Modelos y procesos de difusión

Forward noising process

Backward diffusion (denoising) process

③ Aprendizaje de modelos de difusión

Inferencia variacional

Parametrización del ruido

④ Aplicaciones

Generación de muestras

Estimación de densidades

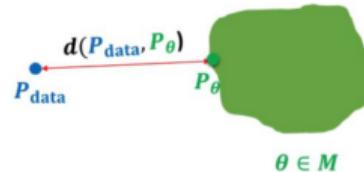
Inferencia de variables latentes

Difusión en espacio latente

Resumen



$\mathbf{x}_n \sim p_{\text{data}}, n=1,2,\dots,N$



Familia de modelos

- Modelos generativos basados en *score*:
 - Producen muestras de alta calidad y tienen un entrenamiento rápido y estable
 - Sin estimación de densidad ni aprendizaje de representaciones

¿Podemos conectar los modelos basados en *score* con las familias de modelos anteriores?

① Repaso: modelado basado en función de *score*

② Modelos y procesos de difusión

Forward noising process

Backward diffusion (denoising) process

③ Aprendizaje de modelos de difusión

Inferencia variacional

Parametrización del ruido

④ Aplicaciones

Generación de muestras

Estimación de densidades

Inferencia de variables latentes

Difusión en espacio latente

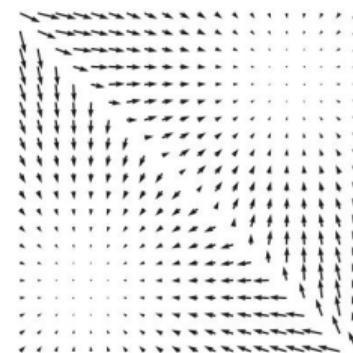
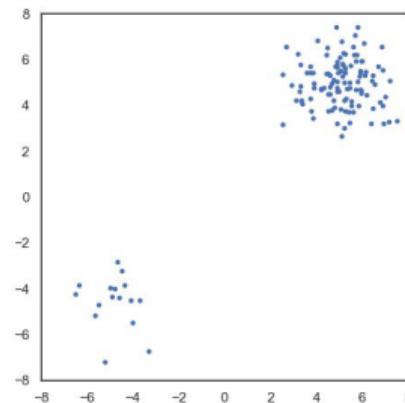
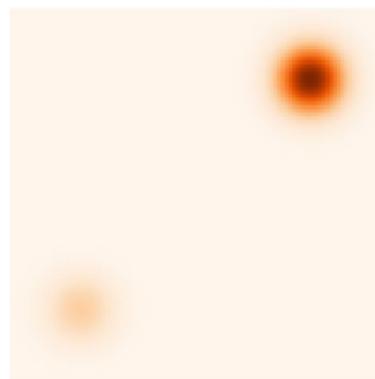
Estimación de la función de *score*: Definición

Idea: aprender un modelo de la función de *score* $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$ a partir de datos.

$$p_{\text{data}}(\mathbf{x})$$

$$\mathbf{x}_1, \dots, \mathbf{x}_N \text{ i.i.d.}$$

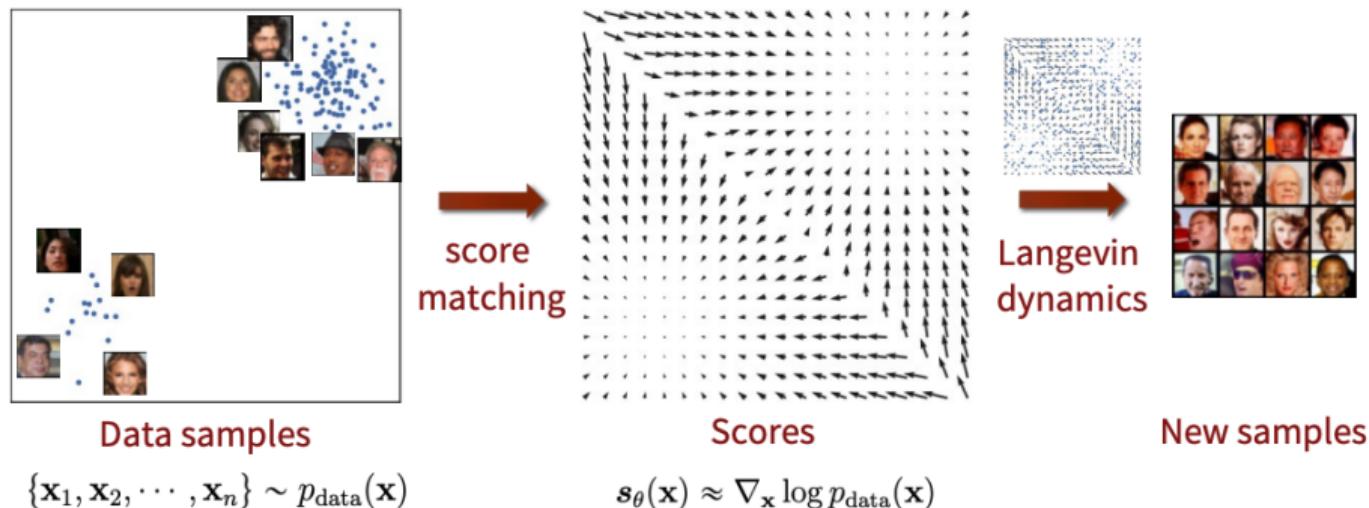
$$\mathbf{s}_{\theta}(\mathbf{x}) \approx \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$$



- Dado un conjunto de datos $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$, queremos estimar $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$
- Modelo: función paramétrica $s_{\theta}(\mathbf{x}) : \mathbb{R}^d \rightarrow \mathbb{R}^d$
- Minimizar la divergencia de Fisher garantiza que $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) \approx s_{\theta}(\mathbf{x})$
- La aproximamos mediante *sliced score matching* o *denoising score matching*

Muestreo usando funciones de *score*

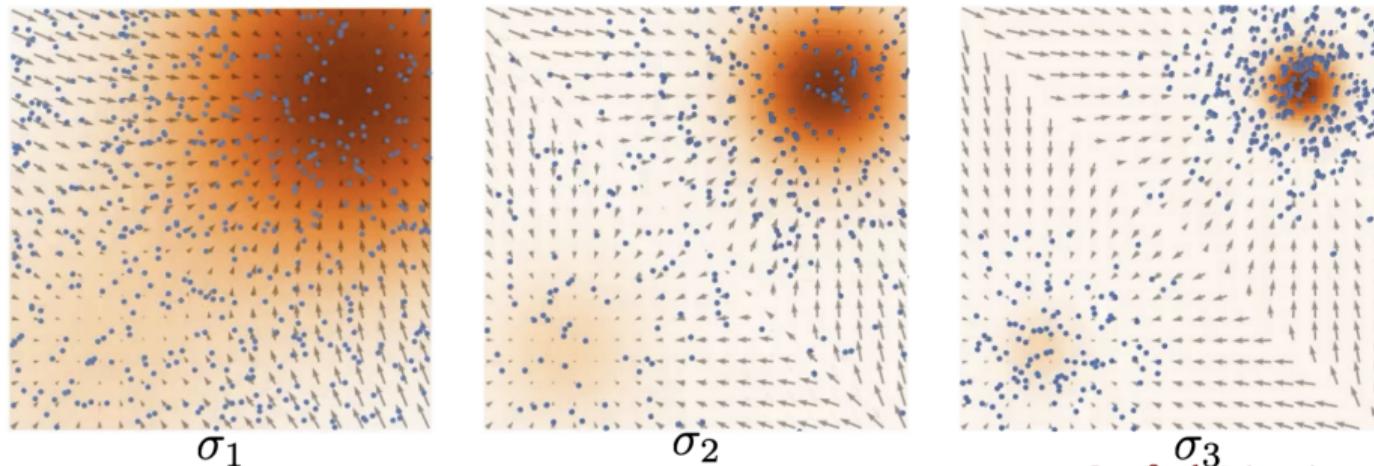
Una vez que entrenado $s_\theta(\mathbf{x}) \approx \nabla_{\mathbf{x}} \log p_\theta(\mathbf{x})$ sobre \mathcal{D} , se generan nuevas \mathbf{x} mediante la dinámica de Langevin (ascenso del gradiente en $\log p_\theta(\mathbf{x})$).



Dado que es difícil aprender $s_\theta(\mathbf{x}) \approx \nabla_{\mathbf{x}} \log p_\theta(\mathbf{x})$ en regiones de baja densidad, utilizamos dinámica de Langevin atenuada, con una red condicional al ruido $s_\theta(\mathbf{x}, \sigma)$.

Dinámica de Langevin atenuada: intuición

Podemos ejecutar la dinámica de Langevin con niveles de ruido decrecientes. En cada nuevo nivel de ruido, comenzamos desde donde terminamos con el nivel anterior.



① Repaso: modelado basado en función de *score*

② Modelos y procesos de difusión

Forward noising process

Backward diffusion (denoising) process

③ Aprendizaje de modelos de difusión

Inferencia variacional

Parametrización del ruido

④ Aplicaciones

Generación de muestras

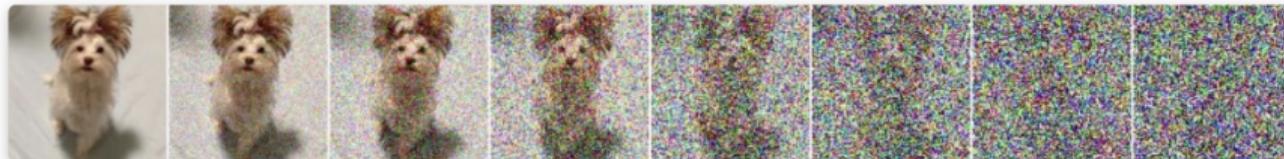
Estimación de densidades

Inferencia de variables latentes

Difusión en espacio latente

Modelos generativos basados en *score*: refinamiento iterativo

Los modelos generativos basados en la *score* aprenden a modelar distribuciones con niveles crecientes de ruido gaussiano:



Perturbing an image with multiple scales of Gaussian noise.

Al muestrear mediante annealed Langevin dynamics, vamos eliminando el ruido: la imagen aparece gradualmente del ruido gaussiano:

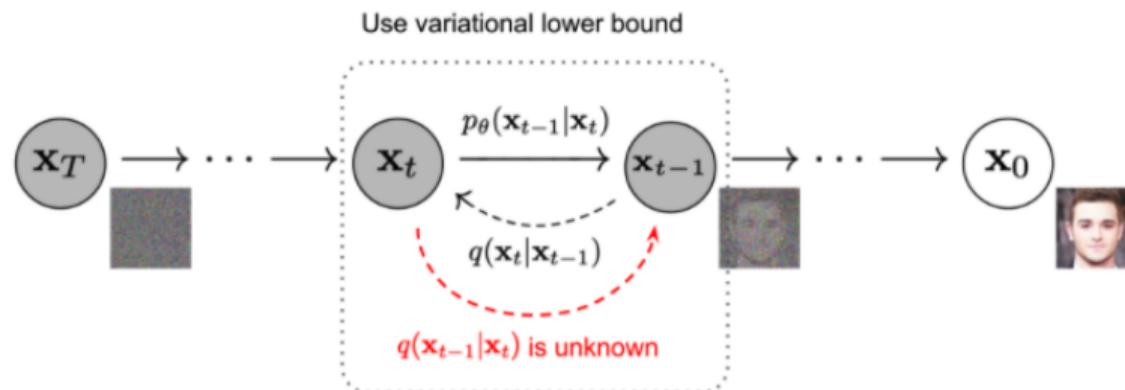


Los modelos de difusión codifican explícitamente este proceso utilizando los conceptos de difusión hacia adelante y hacia atrás.

Modelos de difusión: intuición

La idea detrás de los modelos de difusión es definir un proceso que degrada gradualmente los datos hasta convertirlos en ruido, y luego aprender la inversa de ese proceso.

- El proceso hacia adelante (*noising*) $q(\mathbf{x}_t | \mathbf{x}_{t-1})$ es definido por el usuario y es análogo al las distribuciones de datos ruidosos en los modelos basados en *score*.
- El proceso hacia atrás $p_{\theta}(\mathbf{x}_{t-1} | \mathbf{x}_t)$ (*denoising*) revierte el proceso de ruido y se aprende de los datos.



- Esto da lugar a una generación iterativa y *coarse-to-fine* (¡muy efectiva!)

① Repaso: modelado basado en función de *score*

② Modelos y procesos de difusión

Forward noising process

Backward diffusion (denoising) process

③ Aprendizaje de modelos de difusión

Inferencia variacional

Parametrización del ruido

④ Aplicaciones

Generación de muestras

Estimación de densidades

Inferencia de variables latentes

Difusión en espacio latente

Proceso de difusión hacia adelante (añadiendo ruido)

Comenzamos por definir el proceso de ruido o difusión hacia adelante.

- Partimos de muestras de datos $\mathbf{x}_0 \sim q(\mathbf{x}_0)$, normalizados t.q. $\text{Var}(\mathbf{x}_0) = 1$.
- Ejecutamos una cadena de Markov que añade ruido gradualmente a los datos, produciendo una secuencia de muestras cada vez más ruidosas: $\mathbf{x}_1, \dots, \mathbf{x}_T$.



Perturbing an image with multiple scales of Gaussian noise.

- En cada $t \in \{1, \dots, T\}$, muestreamos \mathbf{x}_t usando el siguiente operador de Markov:

$$q(\mathbf{x}_t | \mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{\alpha_t} \mathbf{x}_{t-1}, (1 - \alpha_t) \mathbf{I}), \quad \text{donde } \alpha_t \in (0, 1), \quad \alpha_t \xrightarrow{t \rightarrow +\infty} 0.$$

- Al final, para T grande, \mathbf{x}_T sigue una distribución gaussiana estándar $\mathcal{N}(0, \mathbf{I})$.
- Obs.: para todo t , la varianza de \mathbf{x}_t se preserva:

$$\text{Var}(\mathbf{x}_1) = \alpha_1 \text{Var}(\mathbf{x}_0) + (1 - \alpha_1) = 1, \dots, \text{Var}(\mathbf{x}_t) = \alpha_t \text{Var}(\mathbf{x}_{t-1}) + (1 - \alpha_t) = \alpha_t \cdot 1 + (1 - \alpha_t) = 1.$$

Cálculo de marginales y conexiones con el *annealing*

- La distribución $q(\mathbf{x}_1, \dots, \mathbf{x}_T)$ tiene marginales que pueden calcularse analíticamente.
- Para calcularlas, obsérvese que para $\varepsilon_k \sim \mathcal{N}(0, \mathbf{I})$, $k = 1, \dots, t$, i.i.d:

$$\mathbf{x}_1 = \sqrt{\alpha_1} \mathbf{x}_0 + \sqrt{1 - \alpha_1} \varepsilon_1$$

$$\mathbf{x}_2 = \sqrt{\alpha_1 \alpha_2} \mathbf{x}_1 + \sqrt{\alpha_2} \sqrt{1 - \alpha_1} \varepsilon_1 + \sqrt{1 - \alpha_2} \varepsilon_2 = \sqrt{\alpha_1 \alpha_2} \mathbf{x}_1 + \sqrt{1 - \alpha_1 \alpha_2} \bar{\varepsilon}_2, \quad \bar{\varepsilon}_2 \sim \mathcal{N}(0, \mathbf{I})$$

⋮

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \bar{\varepsilon}_t, \quad \bar{\varepsilon}_t \sim \mathcal{N}(0, \mathbf{I}), \quad \text{con } \bar{\alpha}_t = \prod_{k=0}^t \alpha_t.$$

⇒ Las distribuciones marginales $q(\mathbf{x}_t | \mathbf{x}_0)$ se escriben: $q(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t} \mathbf{x}_0, (1 - \bar{\alpha}_t) \mathbf{I})$.

- i.e., la difusión produce distribuciones ruidosas, como en los modelos basados en *score*:

$$\sigma_1 > \sigma_2 > \dots > \sigma_{L-1} > \sigma_L$$



① Repaso: modelado basado en función de *score*

② Modelos y procesos de difusión

Forward noising process

Backward diffusion (denoising) process

③ Aprendizaje de modelos de difusión

Inferencia variacional

Parametrización del ruido

④ Aplicaciones

Generación de muestras

Estimación de densidades

Inferencia de variables latentes

Difusión en espacio latente

Proceso de difusión hacia atrás (*denoising*)

Queremos aprender un modelo que revierta el proceso de añadido de ruido.

- Proceso de *denoising* ideal: inversa de la cadena de Markov anterior. En t , quisiéramos muestrear de $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$, pero no conocemos este proceso \Rightarrow lo aprendemos de los datos.
- Definimos un modelo $p_\theta(\mathbf{x}_{0:T}) = p(\mathbf{x}_T) \prod_{t=1}^T p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ donde $\mathbf{x}_1, \dots, \mathbf{x}_T$ son V.A.s latentes, y

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)).$$

- Para generar a partir de este modelo, muestreamos ruido \mathbf{x}_T , y luego muestreamos sucesivamente de $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ hasta obtener una imagen \mathbf{x}_0 .



A continuación, veremos cómo aprender $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ a partir de $q(\mathbf{x}_t|\mathbf{x}_{t-1})$.

① Repaso: modelado basado en función de *score*

② Modelos y procesos de difusión

Forward noising process

Backward diffusion (denoising) process

③ Aprendizaje de modelos de difusión

Inferencia variacional

Parametrización del ruido

④ Aplicaciones

Generación de muestras

Estimación de densidades

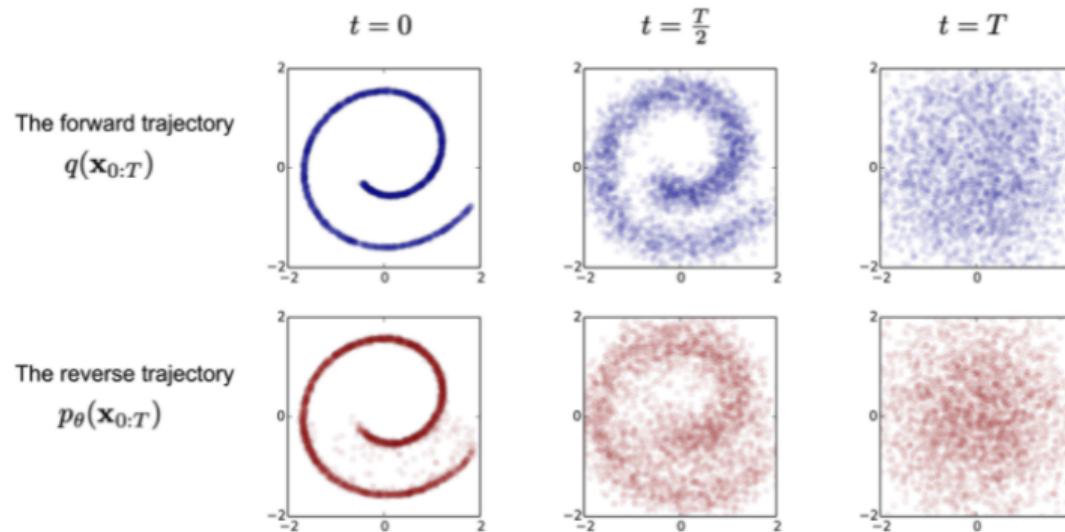
Inferencia de variables latentes

Difusión en espacio latente

Aprendizaje del proceso *backward* a partir del proceso *forward*

- **Objetivo:** aprender $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ a partir de $q(\mathbf{x}_t|\mathbf{x}_{t-1})$. Minimizaremos la divergencia KL:

$$D_{KL}(q(\mathbf{x}_{1:T}|\mathbf{x}_0) \| p_\theta(\mathbf{x}_{1:T}|\mathbf{x}_0)).$$



- En este ejemplo, queremos ajustar las distribuciones p_θ a las distribuciones q .

① Repaso: modelado basado en función de *score*

② Modelos y procesos de difusión

Forward noising process

Backward diffusion (denoising) process

③ Aprendizaje de modelos de difusión

Inferencia variacional

Parametrización del ruido

④ Aplicaciones

Generación de muestras

Estimación de densidades

Inferencia de variables latentes

Difusión en espacio latente

Aprendizaje de modelos de difusión mediante inferencia variacional

Aplicamos inferencia variacional y optimizamos la ELBO sobre la $\log p_\theta(\mathbf{x}_0)$ de un dato \mathbf{x}_0 :

$$\begin{aligned}\log p_\theta(\mathbf{x}_0) &= \log \int q(\mathbf{x}_{1:T}|\mathbf{x}_0) \frac{p_\theta(\mathbf{x}_0, \mathbf{x}_{1:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} d\mathbf{x}_{1:T} \\ &\geq \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \left[\log \frac{p_\theta(\mathbf{x}_0, \mathbf{x}_{1:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \right] = \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \left[\log \frac{p_\theta(\mathbf{x}_{1:T}|\mathbf{x}_0)p_\theta(\mathbf{x}_0)}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \right] \\ &= -\underbrace{\mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \left[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta(\mathbf{x}_{1:T}|\mathbf{x}_0)p_\theta(\mathbf{x}_0)} \right]}_{\mathcal{L}(\mathbf{x}_0; \theta)} = \log p_\theta(\mathbf{x}_0) - D_{KL}(q(\mathbf{x}_{1:T}|\mathbf{x}_0) \| p_\theta(\mathbf{x}_{1:T}|\mathbf{x}_0)).\end{aligned}$$

⇒ Esto se parece mucho a la función objetivo de un VAE:

- Al optimizar esta cota, maximizamos $p(\mathbf{x}_0)$ y minimizamos $D_{KL}(q(\mathbf{x}_{1:T}|\mathbf{x}_0) \| p_\theta(\mathbf{x}_{1:T}|\mathbf{x}_0))$.
- A diferencia de un VAE, el encoder q está diseñado manualmente y no se entrena: solo optimizamos los decoders $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$.

Transformando la ELBO

Ambos p y q tienen estructuras especiales (Markov) que podemos usar para transformar la ELBO en una suma de términos más simple:

$$\begin{aligned}\mathcal{L}(\mathbf{x}_0; \theta) &= \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \left[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T})} \right] = \mathbb{E}_q \left[\log \frac{q(\mathbf{x}_1|\mathbf{x}_0) \prod_{t=1}^T q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)}{p_\theta(\mathbf{x}_T)p_\theta(\mathbf{x}_0|\mathbf{x}_1) \prod_{t=2}^T p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} \right] \\ &= \mathbb{E}_q \left[-\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^T \log \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} + \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1)} \right] \\ (\text{Bayes}) \quad &= \mathbb{E}_q \left[-\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^T \log \left(\frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} \cdot \frac{q(\mathbf{x}_t|\mathbf{x}_0)}{q(\mathbf{x}_{t-1}|\mathbf{x}_0)} \right) + \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1)} \right] \\ (\text{Telescopica}) \quad &= \mathbb{E}_q \left[-\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^T \log \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} + \log \frac{q(\mathbf{x}_T|\mathbf{x}_0)}{q(\mathbf{x}_1|\mathbf{x}_0)} + \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1)} \right] \\ &= \mathbb{E}_q \left[\log \frac{q(\mathbf{x}_T|\mathbf{x}_0)}{p_\theta(\mathbf{x}_T)} + \sum_{t=2}^T \log \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} - \log p_\theta(\mathbf{x}_0|\mathbf{x}_1) \right]\end{aligned}$$

Una ELBO más conveniente

Tenemos entonces:

$$L_{VLB} = \mathbb{E}_q \left[\underbrace{D_{KL}(q(\mathbf{x}_T|\mathbf{x}_0)\|p_\theta(\mathbf{x}_T))}_{L_T} + \sum_{t=2}^T \underbrace{D_{KL}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)\|p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}} - \underbrace{\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0} \right]$$

- ① **Término de prior:** $L_T = D_{KL}(q(\mathbf{x}_T|\mathbf{x}_0)\|p_\theta(\mathbf{x}_T))$ compara el final \mathbf{x}_T . Es cero por construcción.
- ② **Término de reconstrucción:** $L_0 = -\log p_\theta(\mathbf{x}_0|\mathbf{x}_1)$. Es la densidad de \mathbf{x}_0 dada la “estimación” \mathbf{x}_1 .
- ③ **Términos de difusión:** $L_{t-1} = D_{KL}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)\|p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))$, $t \in \{2, \dots, T\}$.
⇒ Miden si el proceso *backward* aprendido $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ se parece al verdadero $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$.

Parametrizando el modelo

Nos enfocamos en los términos de costo de difusión:

$$L_{t-1} = D_{KL}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \| p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)), \quad t \in \{1, 2, \dots, T-1\}.$$

- Debemos calcular primero $q(\mathbf{x}_t|\mathbf{x}_{t+1}, \mathbf{x}_0)$:

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \frac{q(\mathbf{x}_{t-1}, \mathbf{x}_t, \mathbf{x}_0)}{q(\mathbf{x}_t, \mathbf{x}_0)} = \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0)q(\mathbf{x}_{t-1}|\mathbf{x}_0)q(\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)q(\mathbf{x}_0)}.$$

Tenemos los términos siguientes:

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0) = q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{\alpha_t}\mathbf{x}_{t-1}, (1 - \alpha_t)\mathbf{I})$$

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1 - \bar{\alpha}_t)\mathbf{I}), \quad \text{con } \bar{\alpha}_t = \prod_{i=1}^T \alpha_t$$

$$q(\mathbf{x}_{t-1}|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \sqrt{\bar{\alpha}_{t-1}}\mathbf{x}_0, (1 - \bar{\alpha}_{t-1})\mathbf{I}).$$

$\Rightarrow q(\mathbf{x}_t|\mathbf{x}_{t+1}, \mathbf{x}_0)$ gaussiana, con media y matriz de covarianza a calcular (slide siguiente).

Parametrizando el modelo

- Operando con las tres gaussianas $q(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{x}_0)$, $q(\mathbf{x}_t | \mathbf{x}_0)$ y $q(\mathbf{x}_{t-1} | \mathbf{x}_0)$ tenemos

$$\frac{q(\mathbf{x}_{t-1}, \mathbf{x}_t, \mathbf{x}_0)}{q(\mathbf{x}_t, \mathbf{x}_0)} = \frac{q(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{x}_0)q(\mathbf{x}_{t-1} | \mathbf{x}_0)}{q(\mathbf{x}_t | \mathbf{x}_0)} = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\mu}(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t \mathbf{I}),$$

con

$$\begin{aligned}\tilde{\mu}(\mathbf{x}_t, \mathbf{x}_0) &= \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1 - \bar{\alpha}_t} \mathbf{x}_0 + \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t} \mathbf{x}_t, \quad \beta_t = 1 - \alpha_t \\ \tilde{\beta}_t &= \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t}.\end{aligned}$$

- A continuación, elegimos un $p(\mathbf{x}_{t-1} | \mathbf{x}_t)$ de la forma $p(\mathbf{x}_{t-1} | \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t), \tilde{\beta}_t \mathbf{I})$, con $\mu_\theta(\mathbf{x}_t)$ parametrizada por una red neuronal.
⇒ Minimizar L_{t-1} significa ajustar $\mu_\theta(\mathbf{x}_t)$ a $\tilde{\mu}(\mathbf{x}_t, \mathbf{x}_0)$, e.g. utilizando SGD sobre θ .

① Repaso: modelado basado en función de *score*

② Modelos y procesos de difusión

Forward noising process

Backward diffusion (denoising) process

③ Aprendizaje de modelos de difusión

Inferencia variacional

Parametrización del ruido

④ Aplicaciones

Generación de muestras

Estimación de densidades

Inferencia de variables latentes

Difusión en espacio latente

Parametrización del ruido

- Lo anterior se puede mejorar buscando predecir, en lugar de la media $\tilde{\mu}(\mathbf{x}_t, \mathbf{x}_0)$, el ruido agregado a \mathbf{x}_0 en el tiempo t , ε_t .
- Para eso, escribamos primero \mathbf{x}_0 en función de \mathbf{x}_t y ε_t , i.e.:

$$\mathbf{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}} (\mathbf{x}_t - \sqrt{1 - \bar{\alpha}_t} \varepsilon_t).$$

- De esto, resulta que $\tilde{\mu}_t := \tilde{\mu}(\mathbf{x}_t, \mathbf{x}_0)$ vale

$$\tilde{\mu}_t = \frac{1}{\sqrt{\bar{\alpha}_t}} \left(\mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \varepsilon_t \right), \text{ con } \varepsilon_t \text{ el ruido añadido a } \mathbf{x}_0 \text{ para obtener } \mathbf{x}_t.$$

- Podemos definir nuestro modelo como

$$\mu_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{\bar{\alpha}_t}} \left(\mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \varepsilon_\theta(\mathbf{x}_t, t) \right).$$

donde $\varepsilon_\theta(\mathbf{x}_t, t)$ es un modelo del ruido ε_t parametrizado por una NN.

⇒ Minimizar L_{t-1} equivale a predecir y eliminar el ruido ε_t de una muestra \mathbf{x}_t corrompida.

Aprendiendo el ruido

Usando la parametrización anterior y que la divergencia KL entre dos gaussianas de misma matriz de covarianza es proporcional a la norma L^2 al cuadrado de la diferencia entre las medias, L_{t-1} se reduce a:

$$\begin{aligned} L_{t-1} &= \mathbb{E}_{\mathbf{x}_0, \varepsilon} \left[C_1 \cdot \| \tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) - \mu_\theta(\mathbf{x}_t, t) \|^2 \right] \\ &= \mathbb{E}_{\mathbf{x}_0, \varepsilon} \left[C_2 \cdot \| \varepsilon_t - \varepsilon_\theta(\mathbf{x}_t, t) \|^2 \right] \\ &= \mathbb{E}_{\mathbf{x}_0, \varepsilon} \left[C_2 \cdot \| \varepsilon_t - \varepsilon_\theta(\sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \varepsilon_t, t) \|^2 \right], \quad C_1, C_2 > 0 \text{ constantes.} \end{aligned}$$

Optimizamos una aproximación MC de $\sum_{t=2}^T L_{t-1}$ repitiendo hasta convergencia:

- ① Muestrear un dato $\mathbf{x}_0 \in \mathcal{D}$
- ② Muestrear un tiempo $t \sim \mathcal{U}\{1, 2, \dots, T\}$
- ③ Muestrear ruido $\varepsilon \sim \mathcal{N}(0, I)$. Generar la muestra ruidosa $\mathbf{x}_t = \sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \varepsilon_t$
- ④ Dar un paso de gradiente sobre $\| \varepsilon_t - \varepsilon_\theta(\mathbf{x}_t, t) \|^2$, y volver a (1).

Denoising score matching vs. modelos de difusión

Sea $q_\sigma(\tilde{\mathbf{x}})$ una versión ruidosa de la distribución de datos, por ejemplo, ruido gaussiano:

$$\tilde{\mathbf{x}} = \mathbf{x} + \sigma \cdot \varepsilon \quad \text{con } \mathbf{x} \sim p(\mathbf{x}), \varepsilon \sim \mathcal{N}(0, I).$$

Denoising score matching aproxima la divergencia de Fisher entre s_θ y $q_\sigma(\tilde{\mathbf{x}})$ como:

$$\frac{1}{2} \mathbb{E}_{\tilde{\mathbf{x}} \sim q_\sigma(\tilde{\mathbf{x}})} \left[\|\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}}) - s_\theta(\tilde{\mathbf{x}})\|^2 \right] = \frac{1}{2} \mathbb{E}_{\mathbf{x} \sim p, \tilde{\mathbf{x}} \sim q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})} \left[\|\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}}|\mathbf{x}) - s_\theta(\tilde{\mathbf{x}})\|^2 \right].$$

- Cuando usamos ruido gaussiano,

$$\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}}|\mathbf{x}) = \frac{\mathbf{x} - \tilde{\mathbf{x}}}{\sigma^2} = \frac{\varepsilon}{\sigma}.$$

⇒ El modelo de *score* $s_\theta(\tilde{\mathbf{x}})$ y el *denoiser* $\varepsilon(\tilde{\mathbf{x}})$ están relacionados:

$$s_\theta(\tilde{\mathbf{x}}) \approx \nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}}|\mathbf{x}) = \frac{\varepsilon}{\sigma} \approx \frac{\varepsilon(\tilde{\mathbf{x}})}{\sqrt{1 - \bar{\alpha}_t}}.$$

① Repaso: modelado basado en función de *score*

② Modelos y procesos de difusión

Forward noising process

Backward diffusion (denoising) process

③ Aprendizaje de modelos de difusión

Inferencia variacional

Parametrización del ruido

④ Aplicaciones

Generación de muestras

Estimación de densidades

Inferencia de variables latentes

Difusión en espacio latente

① Repaso: modelado basado en función de *score*

② Modelos y procesos de difusión

Forward noising process

Backward diffusion (denoising) process

③ Aprendizaje de modelos de difusión

Inferencia variacional

Parametrización del ruido

④ Aplicaciones

Generación de muestras

Estimación de densidades

Inferencia de variables latentes

Difusión en espacio latente

Ancestral sampling

Dado un modelo entrenado $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$, podemos generar \mathbf{x}_0 realizando *ancestral sampling*:

$$\mathbf{x}_{t-1} \sim p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) \quad \text{para } t = T, T-1, \dots, 1$$

Recordemos que

$$\mathbf{x}_{t-1} = \mu_\theta(\mathbf{x}_t) + \sigma_t \cdot \mathbf{z}, \quad \text{con } \mathbf{z} \sim \mathcal{N}(0, I),$$

y

$$\mu_\theta(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \varepsilon_\theta(\mathbf{x}_t, t) \right).$$

Así, tenemos:

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \varepsilon_\theta(\mathbf{x}_t, t) \right) + \sigma_t \cdot \mathbf{z},$$

para $t \in \{T, T-1, \dots, 1\}$.

Muestreo Ancestral vs. Dinámica de Langevin

Recordemos que en la dinámica de Langevin realizamos repetidamente la actualización:

$$\mathbf{x}_{\text{new}} = \mathbf{x} + \frac{\alpha_t}{2} \nabla_{\mathbf{x}} \log p_{\theta}(\mathbf{x}) + \sqrt{\alpha_t} \varepsilon_t.$$

Recordemos también que el modelo de score $s_{\theta}(\tilde{\mathbf{x}})$ y el denoiser $\varepsilon(\tilde{\mathbf{x}})$ están relacionados:

$$s_{\theta}(\tilde{\mathbf{x}}) \approx \nabla_{\tilde{\mathbf{x}}} \log q_{\sigma}(\tilde{\mathbf{x}}|\mathbf{x}) = \frac{\varepsilon}{\sigma} \approx \frac{\varepsilon(\tilde{\mathbf{x}})}{\sqrt{1 - \bar{\alpha}_t}}.$$

Dado que el proceso de muestreo para un modelo de difusión es:

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \varepsilon_{\theta}(\mathbf{x}_t, t) \right) + \sigma_t \cdot \mathbf{z},$$

puede verse como una forma de dinámica de Langevin (atenuada).

Muestras de Modelos de Difusión

Las muestras son al menos tan buenas como las de las GAN, pero los modelos de difusión son mucho más fáciles de entrenar.

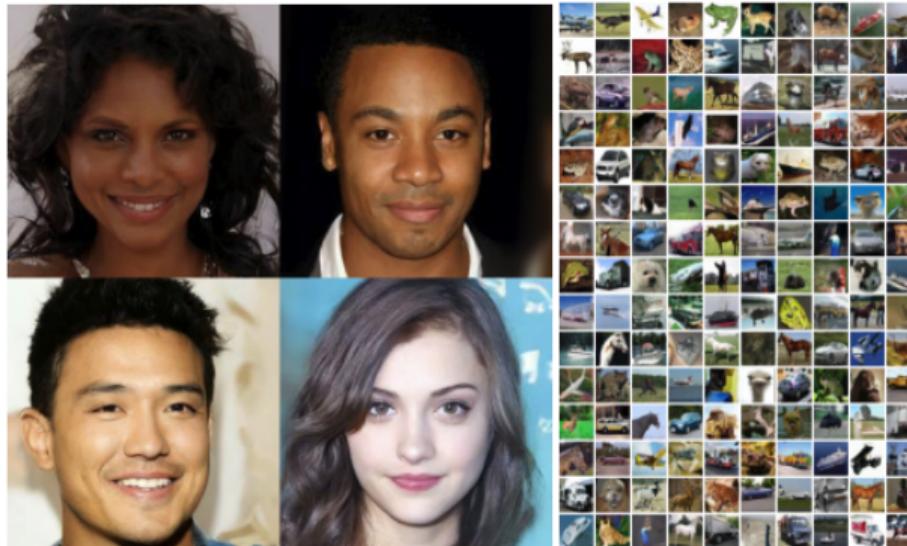


Figure 1: Generated samples on CelebA-HQ 256 × 256 (left) and unconditional CIFAR10 (right)

Desventaja con respecto a las GAN: generación de muestras mucho más lenta (esto está mejorando mucho).

Muestras Condicionales de Modelos de Difusión

También se puede condicionar el generador $p(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{y})$ a datos externos \mathbf{y} en el momento del entrenamiento (por ejemplo, un prompt con embedding de texto).



Los modelos de difusión se utilizan ampliamente como decodificadores en sistemas de texto a imagen como DALLE2, Stable Diffusion y Midjourney.

Muestreo condicional y generación controlable

Supongamos que conocemos un proceso $p(\mathbf{y}|\mathbf{x})$ que mapea datos \mathbf{x} a variables auxiliares \mathbf{y} (por ejemplo, etiquetas o imágenes con ruido).

Queremos modelar $p(\mathbf{x}|\mathbf{y})$ para resolver problemas inversos o hacer generación condicionada.

$$\nabla_{\mathbf{x}} \log p(\mathbf{x}|\mathbf{y}) = \nabla_{\mathbf{x}} \log p(\mathbf{x}) + \nabla_{\mathbf{x}} \log p(\mathbf{y}|\mathbf{x}).$$

Los modelos basados en score son naturalmente adecuados para esta tarea:

- Podemos aprender $\nabla_{\mathbf{x}} \log p(\mathbf{x})$ de manera incondicional.
- $\nabla_{\mathbf{x}} \log p(\mathbf{y}|\mathbf{x})$ ya se conoce (por ejemplo, es un clasificador).
- Por lo tanto, podemos muestrear a partir de $\nabla_{\mathbf{x}} \log p(\mathbf{x}|\mathbf{y})$ usando dinámica de Langevin.

Classifier-Free Guidance entrena $\nabla_{\mathbf{x}} \log p(\mathbf{x}|\mathbf{y})$ y $\nabla_{\mathbf{x}} \log p(\mathbf{x})$ directamente cuando (\mathbf{x}, \mathbf{y}) está disponible durante el entrenamiento (mismo modelo, con *dropout* en \mathbf{y}). En la inferencia, se combinan ambos con un peso. Véase Ho et al., 2022.

Modelos basados en *score* y modelos de difusión: muestras condicionales

Este es el resultado de generar muestras condicionales por clase:



Otros ejemplos incluyen: condicionamiento de imágenes en color sobre imágenes monocromáticas (colorización), condicinamiento de imágenes sobre versiones con oclusiones (inpainting), etc. (problemas inversos mal condicionados)

① Repaso: modelado basado en función de *score*

② Modelos y procesos de difusión

Forward noising process

Backward diffusion (denoising) process

③ Aprendizaje de modelos de difusión

Inferencia variacional

Parametrización del ruido

④ Aplicaciones

Generación de muestras

Estimación de densidades

Inferencia de variables latentes

Difusión en espacio latente

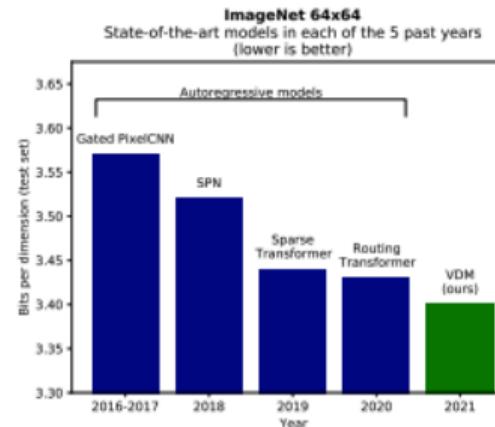
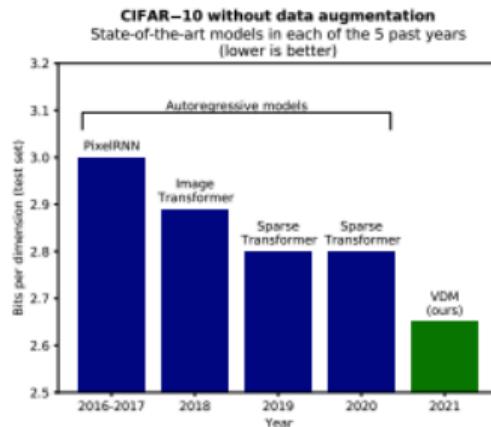
Maximización de la verosimilitud con modelos de difusión

Los modelos de difusión optimizan una cota inferior de la log-verosimilitud:

$$\log p_{\theta}(\mathbf{x}_0) \geq -\mathbb{E}_q \left[\log \frac{q(\mathbf{x}_{1:T} | \mathbf{x}_0)}{p_{\theta}(\mathbf{x}_{0:T})} \right] = -L_{\text{LVB}}$$

Los modelos de difusión variacional (Kingma et al., 2021) introducen:

- Una discretización más fina ($T \rightarrow \infty$) y un *scheduling* aprendido para α_t .
- Características de Fourier para \mathbf{x}_t en $\varepsilon(\mathbf{x}_t, t)$ para mejorar la predicción de \mathbf{x}_{t-1} .



① Repaso: modelado basado en función de *score*

② Modelos y procesos de difusión

Forward noising process

Backward diffusion (denoising) process

③ Aprendizaje de modelos de difusión

Inferencia variacional

Parametrización del ruido

④ Aplicaciones

Generación de muestras

Estimación de densidades

Inferencia de variables latentes

Difusión en espacio latente

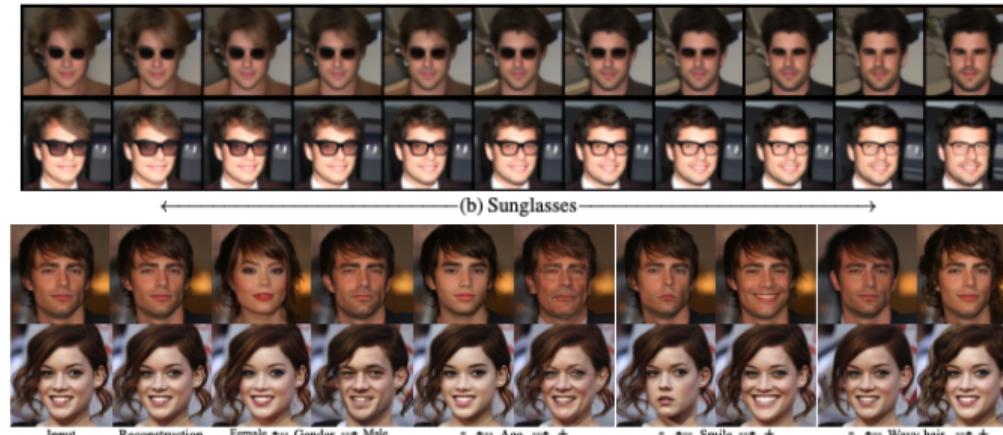
Inferencia de variables latentes con modelos de difusión

Los modelos de difusión pueden ampliarse con variables latentes de baja dimensión \mathbf{z} :

$$\log p_{\theta}(\mathbf{x}_0) \geq \mathbb{E}_q \left[\log \frac{p_{\theta}(\mathbf{x}_0, \mathbf{z})}{q(\mathbf{z}|\mathbf{x}_0)} \right] \geq \mathbb{E}_q \left[\log \frac{p_{\theta}(\mathbf{x}_{0:T}, \mathbf{z})}{q(\mathbf{x}_{1:T}, \mathbf{z}|\mathbf{x}_0)} \right]$$

donde aplicamos el ELBO dos veces. Muestramos de $p(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{z})$.

El modelo resultante usa \mathbf{z} para codificar información semántica y \mathbf{x}_T para codificar detalles (por ejemplo, texturas) (Preechakul et al., 2022; Wang et al., 2023).



① Repaso: modelado basado en función de *score*

② Modelos y procesos de difusión

Forward noising process

Backward diffusion (denoising) process

③ Aprendizaje de modelos de difusión

Inferencia variacional

Parametrización del ruido

④ Aplicaciones

Generación de muestras

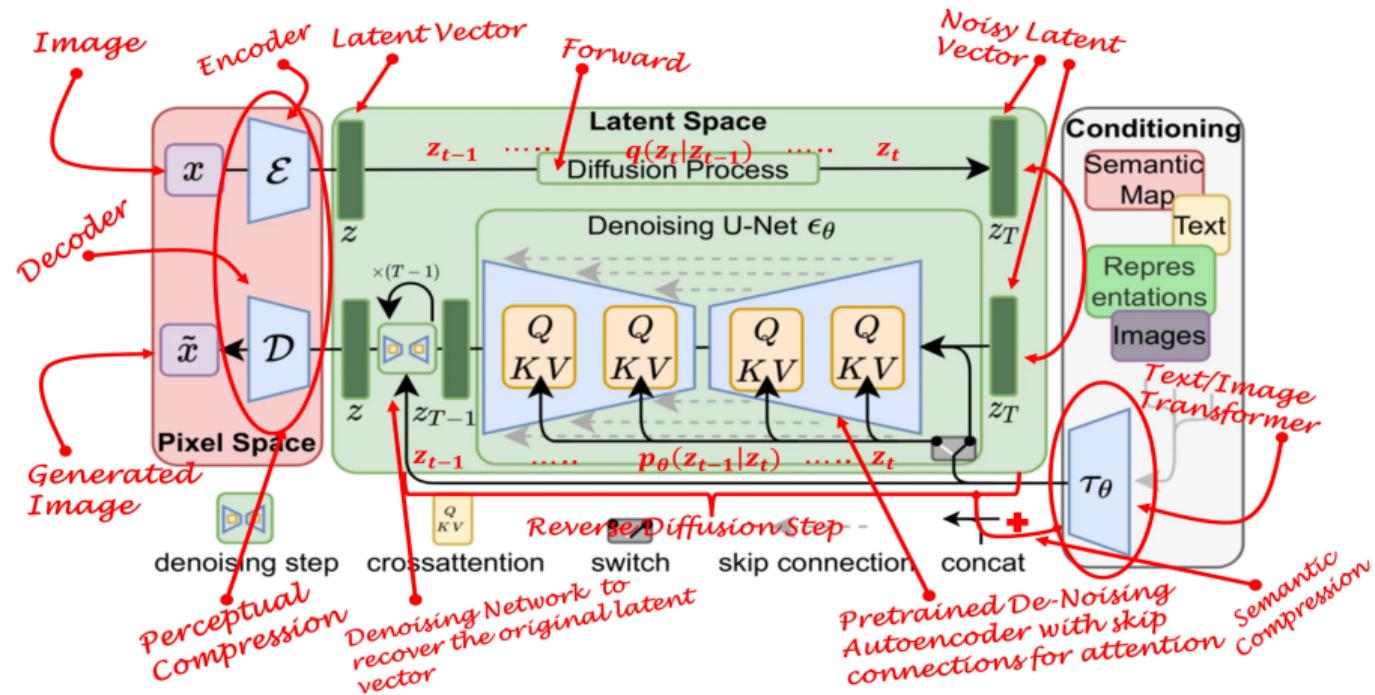
Estimación de densidades

Inferencia de variables latentes

Difusión en espacio latente

Difusión en espacio latente (*latent diffusion models*)

Stable diffusion



Modelos de Difusión: Ventajas y Contras

Ventajas:

- ① Muestras de muy alta calidad (comparables con GANs) con un objetivo estable.
- ② Generación controlable efectiva (con y sin variables latentes).
- ③ También son estado del arte en estimación de densidad.

Contras:

- ① El muestreo sigue siendo bastante lento (es necesario ejecutar una cadena MCMC). Hay samplers mucho más rápidos (e.g., DDIM) y nuevas estrategias de generación instantánea (consistency models).
- ② Funciona peor con datos discretos (área de investigación activa).