

Modelos Generativos Profundos para Imágenes

Modelos generativos basados en *score*

Pablo Musé

pmuse@fing.edu.uy

Instituto de Ingeniería Eléctrica
Facultad de Ingeniería



UNIVERSIDAD
DE LA REPÚBLICA
URUGUAY

Agenda

① Función de *score*

② Ajuste de la función de *score*

Divergencia de Fisher

Sliced score matching

Denoising score matching

③ Generación de muestras

Dinámica de Langevin

Hipótesis de la variedad

Annealed Langevin dynamics

① Función de *score*

② Ajuste de la función de *score*

Divergencia de Fisher

Sliced score matching

Denoising score matching

③ Generación de muestras

Dinámica de Langevin

Hipótesis de la variedad

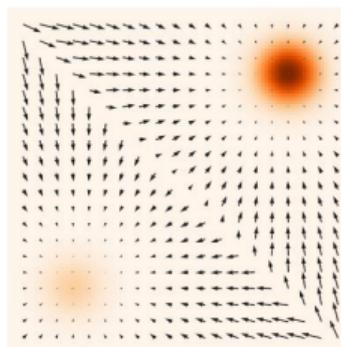
Annealed Langevin dynamics

Función de *score* (de Stein): definición

Dada una densidad de probabilidad $p_\theta(\mathbf{x})$, $\mathbf{x} \in \mathbb{R}^d$, la función de *score* (de Stein) se define como

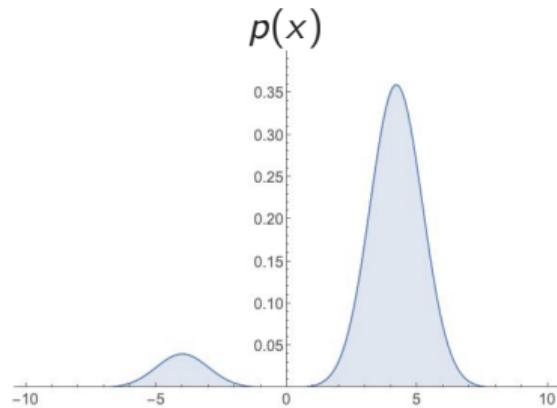
$$\nabla_{\mathbf{x}} \log p_\theta(\mathbf{x}).$$

- Es el gradiente de la densidad con respecto a la entrada.
- Esto es **distinto** al gradiente $\nabla_\theta \log p_\theta(\mathbf{x})$ de la log-densidad con respecto a los parámetros θ (que es lo que usamos en el descenso gradiente).
- La función de score apunta en la dirección del aumento de la densidad del modelo.

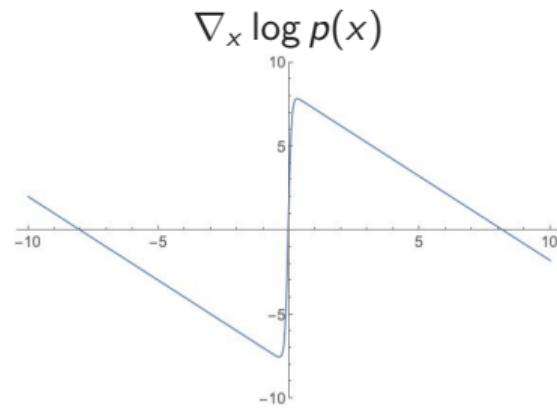


Función de *score* (de Stein): ejemplo 1D

Mezcla de gaussianas:



Función de *score* correspondiente:



Obs.: la función de *score* es negativa (\leftarrow) cerca del modo izquierdo y es positiva (\rightarrow) cerca del modo derecho.

Las funciones de *score* no incluyen constantes de normalización

Supongamos que tenemos un modelo basado en energía de la forma

$$p_\theta(\mathbf{x}) = \frac{1}{\int \exp(f_\theta(\mathbf{x})) d\mathbf{x}} \exp(f_\theta(\mathbf{x})) = \frac{1}{Z(\theta)} \exp(f_\theta(\mathbf{x}))$$

- El cálculo de una función de *score* del modelo no requiere $Z(\theta)$:

$$\nabla_{\mathbf{x}} \log p_\theta(\mathbf{x}) = \nabla_{\mathbf{x}} f_\theta(\mathbf{x}) - \nabla_{\mathbf{x}} \log Z(\theta) = \nabla_{\mathbf{x}} f_\theta(\mathbf{x})$$

- **Idea:** ¡En lugar de aprender $p_\theta(\mathbf{x})$, se aprende un modelo $\nabla_{\mathbf{x}} \log p_\theta(\mathbf{x})$ de la función de *score* de los datos $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$!

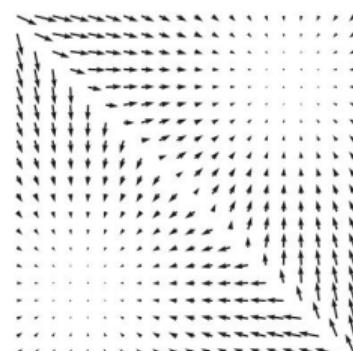
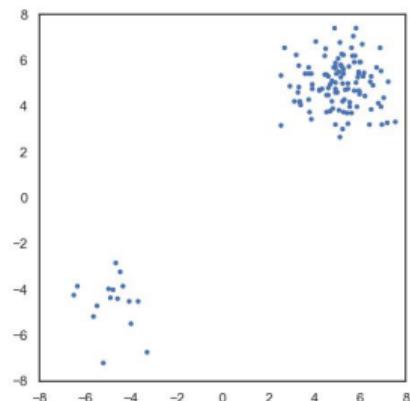
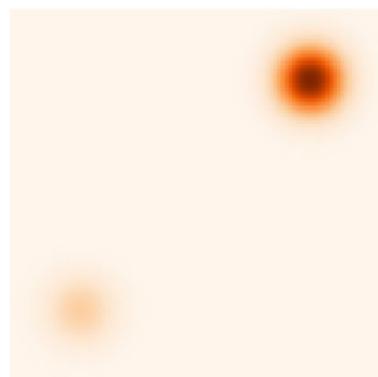
Estimación de la función de score

Idea: aprender un modelo $s_\theta(\mathbf{x}) : \mathbb{R}^d \rightarrow \mathbb{R}^d$ de la función de score a partir de $\mathcal{D} = \{\mathbf{x}_n\}_{n=1}^N$.

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

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

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



- **Objetivo:** encontrar θ para que $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) \approx \mathbf{s}_\theta(\mathbf{x})$

① Función de *score*

② Ajuste de la función de *score*

Divergencia de Fisher

Sliced score matching

Denoising score matching

③ Generación de muestras

Dinámica de Langevin

Hipótesis de la variedad

Annealed Langevin dynamics

① Función de *score*

② Ajuste de la función de *score*

Divergencia de Fisher

Sliced score matching

Denoising score matching

③ Generación de muestras

Dinámica de Langevin

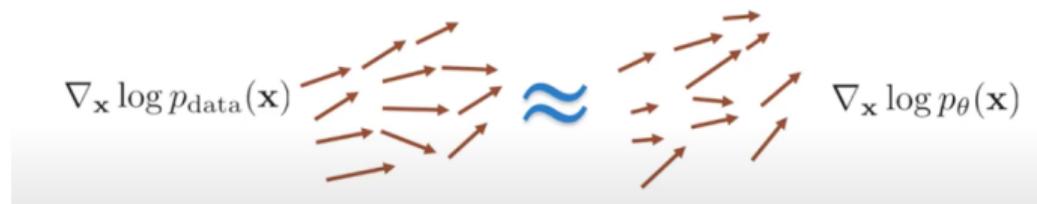
Hipótesis de la variedad

Annealed Langevin dynamics

Ajuste de score basado en la divergencia de Fisher

¿Cómo se ajustan las funciones de score?

- Los scores de los datos y del modelo son campos vectoriales. Queremos alinearlos:



- Formalmente, utilizamos la divergencia de Fisher:

$$\frac{1}{2} \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}} [\|\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - s_{\theta}(\mathbf{x})\|_2^2], \quad s_{\theta}(\mathbf{x}) = (s_{\theta,1}(\mathbf{x}), \dots, s_{\theta,d}(\mathbf{x})).$$

- Es la distancia euclídea media entre dos vectores sobre todas las \mathbf{x}
- La divergencia de Fisher es cero si y sólo si $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) = s_{\theta}(\mathbf{x})$.

Una función objetivo práctica para el ajuste por score

La divergencia de Fisher no nos da un objetivo que podamos optimizar fácilmente.

$$\frac{1}{2} \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}} [\|\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - s_{\theta}(\mathbf{x})\|_2^2].$$

- No conocemos $\log p_{\text{data}}(\mathbf{x})$, y mucho menos su gradiente.
- Se puede demostrar que si $p_{\text{data}}(\mathbf{x})s_{\theta}(\mathbf{x}) \xrightarrow{\|\mathbf{x}\| \rightarrow +\infty} 0$, entonces se obtiene el costo equivalente*

$$J(\theta) = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}} \left[\frac{1}{2} \|s_{\theta}(\mathbf{x})\|_2^2 + \text{tr}(\nabla_{\mathbf{x}} s_{\theta}(\mathbf{x})) \right].$$

- Aproximando este objetivo de *score matching* por Monte Carlo con las $\mathbf{x}_n \sim p_{\text{data}}$:

$$J(\theta) \approx \frac{1}{N} \sum_{n=1}^N \left[\frac{1}{2} \|s_{\theta}(\mathbf{x}_n)\|_2^2 + \text{tr}(\nabla_{\mathbf{x}} s_{\theta}(\mathbf{x}_n)) \right].$$

* Hyvärinen et al. *Estimation of Non-Normalized Statistical Models*, JMLR 2005.

Función objetivo de *score matching*: prueba (caso 1D)

Se aplica integración por partes y se impone la condición $\lim_{\|\mathbf{x}\| \rightarrow +\infty} p_{\text{data}}(\mathbf{x})s_\theta(\mathbf{x}) = 0$. La divergencia de Fisher se escribe

$$\frac{1}{2} \int_{-\infty}^{\infty} p(\mathbf{x})(\nabla_{\mathbf{x}} \log p(\mathbf{x}))^2 d\mathbf{x} + \frac{1}{2} \int_{-\infty}^{\infty} p(\mathbf{x})s_\theta(\mathbf{x}^2) d\mathbf{x} - \int_{-\infty}^{\infty} p(\mathbf{x})\nabla_{\mathbf{x}} \log p(\mathbf{x})s_\theta(\mathbf{x})^2 d\mathbf{x}.$$

- El primer término es independiente de θ .
- Se aplica integración por partes al tercer término:

$$\begin{aligned} \int_{-\infty}^{+\infty} p(\mathbf{x})\nabla_{\mathbf{x}} \log p(\mathbf{x})s_\theta(\mathbf{x})^2 d\mathbf{x} &= \int_{-\infty}^{+\infty} p(\mathbf{x}) \frac{\nabla_{\mathbf{x}} p(\mathbf{x})}{p(\mathbf{x})} s_\theta(\mathbf{x})^2 d\mathbf{x} = \int_{-\infty}^{+\infty} \nabla_{\mathbf{x}} p(\mathbf{x}) s_\theta(\mathbf{x})^2 d\mathbf{x} \\ &= \underbrace{p(\mathbf{x})s_\theta(\mathbf{x})|_{-\infty}^{+\infty}}_{=0} - \int_{-\infty}^{+\infty} p(\mathbf{x})\nabla_{\mathbf{x}} s_\theta(\mathbf{x}) d\mathbf{x} \end{aligned}$$

Luego, dejando de lado término constante en θ , se obtiene $J(\theta)$.

Score matching no escala bien con la dimensión

Ahora podemos intentar ajustar un modelo de *score matching*:

- Parametrizamos $s_\theta(\mathbf{x})$ mediante una red neuronal y optimizamos $J(\theta)$ mediante descenso por el gradiente:

$$J(\theta) \approx \frac{1}{N} \sum_{n=1}^N \left[\frac{1}{2} \|s_\theta(\mathbf{x}_n)\|_2^2 + \text{tr}(\nabla_{\mathbf{x}} s_\theta(\mathbf{x}_n)) \right].$$

- Esto no escala bien:** la optimización por descenso de gradiente requiere que se calcule un *backprop* anidado en cada elemento diagonal de $\nabla_{\mathbf{x}} s_\theta(\mathbf{x})$:

$$\nabla_{\mathbf{x}} s_\theta(\mathbf{x}) = \begin{pmatrix} \frac{\partial s_{\theta,1}(\mathbf{x})}{\partial x_1} & \frac{\partial s_{\theta,1}(\mathbf{x})}{\partial x_2} & \frac{\partial s_{\theta,1}(\mathbf{x})}{\partial x_3} \\ \frac{\partial s_{\theta,2}(\mathbf{x})}{\partial x_1} & \frac{\partial s_{\theta,2}(\mathbf{x})}{\partial x_2} & \frac{\partial s_{\theta,2}(\mathbf{x})}{\partial x_3} \\ \frac{\partial s_{\theta,3}(\mathbf{x})}{\partial x_1} & \frac{\partial s_{\theta,3}(\mathbf{x})}{\partial x_2} & \frac{\partial s_{\theta,3}(\mathbf{x})}{\partial x_3} \end{pmatrix}$$

⇒ Son d^2 derivadas.

① Función de *score*

② Ajuste de la función de *score*

Divergencia de Fisher

Sliced score matching

Denoising score matching

③ Generación de muestras

Dinámica de Langevin

Hipótesis de la variedad

Annealed Langevin dynamics

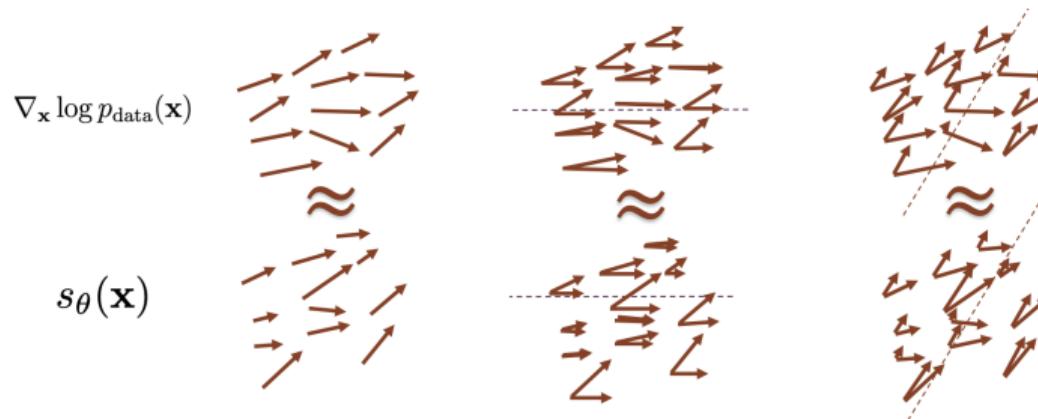
Sliced score matching

Tenemos que encontrar una aproximación más eficiente al objetivo de *score matching*

$$J(\theta) = \frac{1}{N} \sum_{n=1}^N \left[\frac{1}{2} \|s_\theta(\mathbf{x}_n)\|_2^2 + \text{tr}(\nabla_{\mathbf{x}} s_\theta(\mathbf{x}_n)) \right].$$

- **Obs.:** en dimensiones bajas, el problema es manejable.

⇒ **Idea:** optimizar las proyecciones 1D de la función de score



Sliced Fisher divergence

La comparación de score optimiza la divergencia de Fisher:

$$\frac{1}{2} \mathbb{E}_{\mathbf{v} \sim q(\mathbf{v})} \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x})} \left[\left(\mathbf{v}^T \nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x}) - \mathbf{v}^T s_{\theta}(\mathbf{x}) \right)^2 \right]$$

- $q(\mathbf{v})$ es una distribución sobre los vectores de proyección \mathbf{v} que elegimos.
- Como antes, esto no es práctico porque no conocemos p_{data} . Con la misma técnica de integración por partes:

$$\frac{1}{2} \mathbb{E}_{\mathbf{v} \sim q(\mathbf{v})} \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x})} \left[\mathbf{v}^T \nabla_{\mathbf{x}} s_{\theta}(\mathbf{x}) \mathbf{v} + \frac{1}{2} (\mathbf{v}^T s_{\theta}(\mathbf{x}))^2 \right].$$

- Cada término del objetivo anterior puede calcularse eficientemente.
- La distribución de la proyección es típicamente gaussiana o Rademacher.

① Función de *score*

② Ajuste de la función de *score*

Divergencia de Fisher

Sliced score matching

Denoising score matching

③ Generación de muestras

Dinámica de Langevin

Hipótesis de la variedad

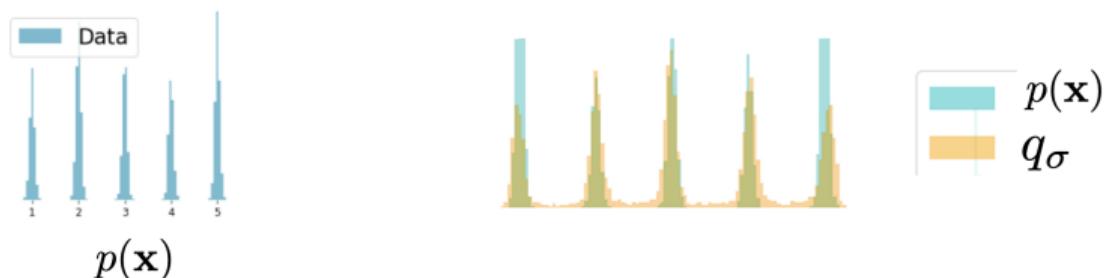
Annealed Langevin dynamics

Denoising score matching

Supongamos que tenemos acceso a una versión ruidosa de los datos limpios $\mathbf{x} \sim p(\mathbf{x})$. Por ejemplo, podemos añadir ruido gaussiano a \mathbf{x} :

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

$$q_\sigma(\tilde{\mathbf{x}}|\mathbf{x}) = \mathcal{N}(\tilde{\mathbf{x}}; \mathbf{x}, \sigma^2 I), \quad q_\sigma(\tilde{\mathbf{x}}) = \int q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})p(\mathbf{x})d\mathbf{x}$$



- La estimación del *score* para $\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}})$ es más sencilla.
- Para nivel de ruido pequeño, $q_\sigma(\tilde{\mathbf{x}}|\mathbf{x}) \approx p(\tilde{\mathbf{x}})$.

Denoising score matching

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

$$\begin{aligned} \frac{1}{2} \mathbb{E}_{\tilde{\mathbf{x}} \sim q_\sigma(\tilde{\mathbf{x}})} [\|\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}}) - s_\theta(\tilde{\mathbf{x}})\|_2^2] &= \dots \text{cuentas (ejercicio)...} = \\ &= \frac{1}{2} \mathbb{E}_{\mathbf{x} \sim p, \tilde{\mathbf{x}} \sim q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})} [\|\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}}|\mathbf{x}) - s_\theta(\tilde{\mathbf{x}})\|_2^2]. \end{aligned}$$

- $\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})$ es fácil de calcular. E.g., para el caso anterior con ruido gaussiano,

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

- Más rápido que *sliced score matching*, pero sólo puede aprender distribuciones con ruido.

*P. Vincent. A Connection Between Score Matching and Denoising Autoencoders. Neural Computation, 2011.

Algoritmo para el aprendizaje de modelos generativos basados en *score*

- Dado un conjunto inicial de parametros $\theta^{(0)}$, para $t = 1, 2, \dots$:
 - ➊ Muestrear un *batch* de muestras de entrenamiento $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N$
 - ➋ Calcular el gradiente $\nabla_{\theta} J(\theta)$ del objetivo, donde $J(\theta)$ es:

$$J_{\text{sliced}}(\theta) = \frac{1}{N} \sum_{n=1}^N \left[\mathbf{v}_n^T \nabla_{\mathbf{x}}^2 s_{\theta}(\mathbf{x}_n) \mathbf{v}_n + \frac{1}{2} \left(\mathbf{v}_n^T \nabla_{\mathbf{x}} s_{\theta}(\mathbf{x}_n) \right)^2 \right]$$

para proyecciones aleatorias de vectores $\{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n\}$ o

$$J_{\text{denoised}}(\theta) = \frac{1}{N} \sum_{n=1}^N \left[\|\nabla_{\tilde{\mathbf{x}}} \log q_{\sigma}(\tilde{\mathbf{x}} | \mathbf{x}_n) - s_{\theta}(\tilde{\mathbf{x}}_n)\|_2^2 \right]$$

para alguna q_{σ} .

- ➌ Tomar un paso de gradiente $\theta^{(t+1)} = \theta^{(t)} - \eta \nabla_{\theta} J(\theta^{(t)})$.

① Función de *score*

② Ajuste de la función de *score*

Divergencia de Fisher

Sliced score matching

Denoising score matching

③ Generación de muestras

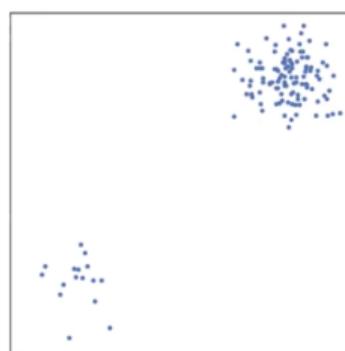
Dinámica de Langevin

Hipótesis de la variedad

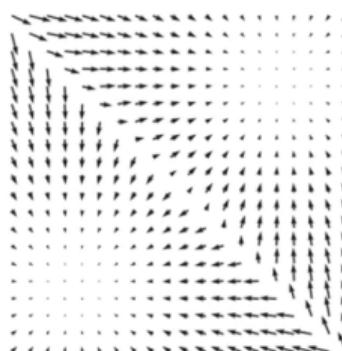
Annealed Langevin dynamics

Generación de muestras: intuición

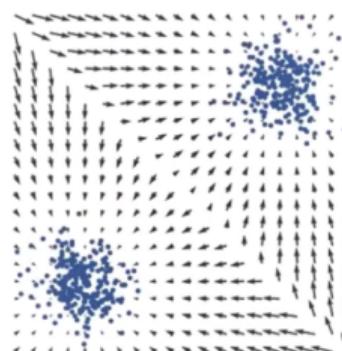
- Habiendo estimado la función de *score* de p_{data} ahora queremos generar muestras \mathbf{x} a partir de la función de *score* $\nabla_{\mathbf{x}} \log p_{\theta}(\mathbf{x})$.
- Lo haremos siguiendo el gradiente de la distribución de los datos



Samples



Score Estimation



Langevin dynamics

- Intuitivamente, empezamos con ruido \mathbf{z} y seguimos el gradiente de la distribución de datos hasta generar \mathbf{x} precisos.

① Función de *score*

② Ajuste de la función de *score*

Divergencia de Fisher

Sliced score matching

Denoising score matching

③ Generación de muestras

Dinámica de Langevin

Hipótesis de la variedad

Annealed Langevin dynamics

Dinámica de Langevin

Proceso de muestreo MCMC que nos permite muestrear a partir de $p_\theta(\mathbf{x})$ utilizando únicamente su *score* $\nabla_{\mathbf{x}} \log p_\theta(\mathbf{x})$:

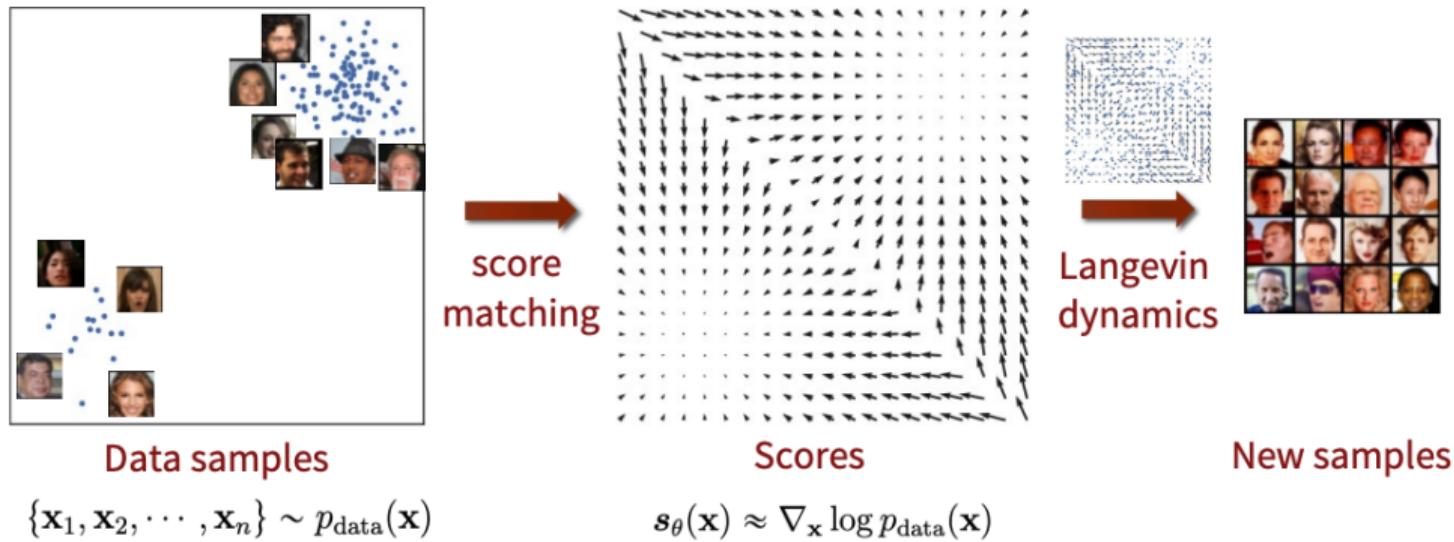
- Inicializar x_0 a partir de una distribución de ruido
- Para los pasos $t = 1, 2, \dots, T$:
 - Muestrear una variable de ruido $\varepsilon_t \sim \mathcal{N}(0, I)$
 - $\mathbf{x}_t = \mathbf{x}_{t-1} + \frac{1}{2}\alpha_t \nabla_{\mathbf{x}} \log p_\theta(\mathbf{x}) + \sqrt{\alpha_t} \varepsilon_t$

Observaciones:

- Este proceso puede verse como un gradiente ascendente en $\log p_\theta(\mathbf{x})$;
- Se garantiza la generación de muestras de $p_\theta(\mathbf{x})$ cuando el tamaño del paso $\alpha_t \rightarrow 0$ cuando $T \rightarrow \infty$, con la tasa de enfriamiento adecuada.
- Podemos utilizar este proceso para el muestreo de una función de *score* $s_\theta(\mathbf{x}) \approx \nabla_{\mathbf{x}} \log p_\theta(\mathbf{x})$ introduciendo s_θ en el algoritmo anterior.

Modelado de funciones de score

En suma, a alto nivel la estrategia consiste en aprender $s_\theta(\mathbf{x}) \approx \nabla_{\mathbf{x}} \log p_\theta(\mathbf{x})$ a partir de un conjunto de muestras, y luego generar nuevas \mathbf{x} mediante la dinámica de Langevin.



① Función de *score*

② Ajuste de la función de *score*

Divergencia de Fisher

Sliced score matching

Denoising score matching

③ Generación de muestras

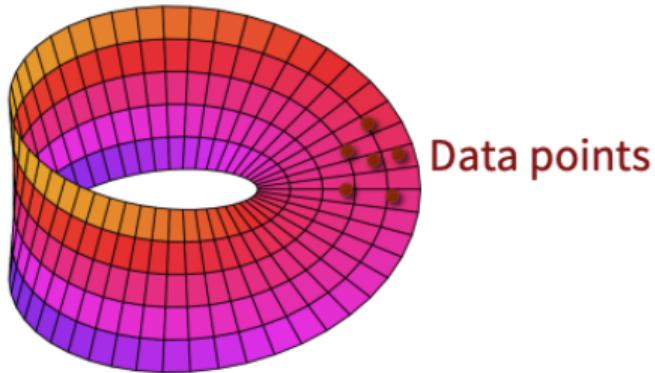
Dinámica de Langevin

Hipótesis de la variedad

Annealed Langevin dynamics

Hipótesis de la variedad

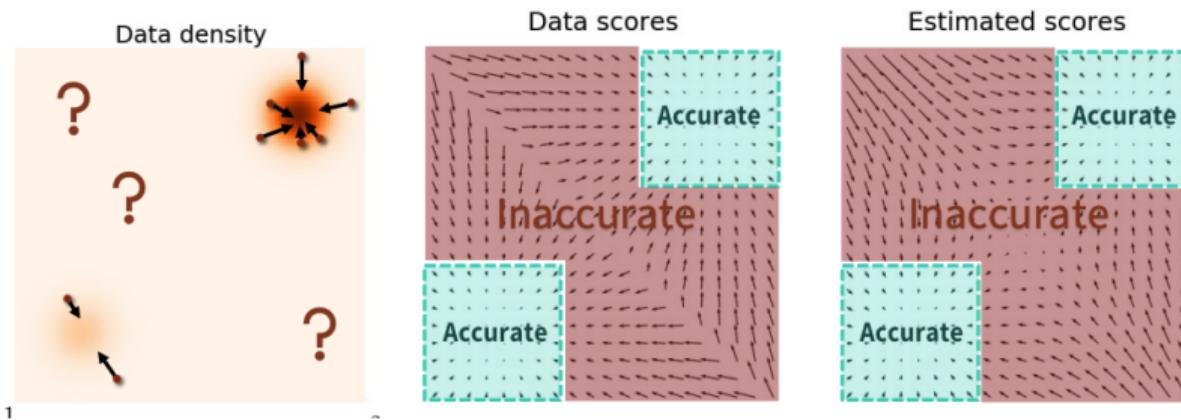
- El enfoque anterior enfrenta desafíos derivados de la **hipótesis de la variedad**, que sostiene que **los datos viven en un subespacio de baja dimensión** dentro del espacio ambiente de alta dimensión.



- Esto es un problema ya que $\nabla_{\mathbf{x}} \log p_{\text{data}}(\mathbf{x})$ no está definido donde $p_{\text{data}}(\mathbf{x}) = 0$

Aprendizaje y muestreo de funciones de *score* en regiones de baja densidad

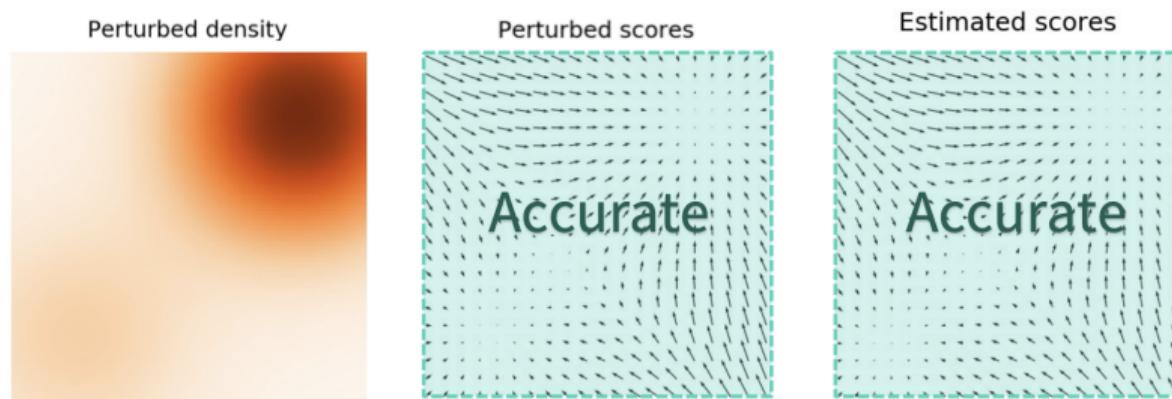
Por lo tanto, el aprendizaje de funciones de *score* y la generación de muestras a partir de ellas resulta difícil en regiones de baja densidad.



- No tenemos suficientes muestras para aprender s_θ en regiones de baja densidad
- Incluso, aún si conociéramos $\nabla_x \log p_\theta(x)$, puede ser demasiado débil para generar muestras.

Aprendizaje y muestreo de funciones de *score* en regiones de baja densidad

Una solución es añadir ruido gaussiano a los datos (es decir, suavizar la densidad):

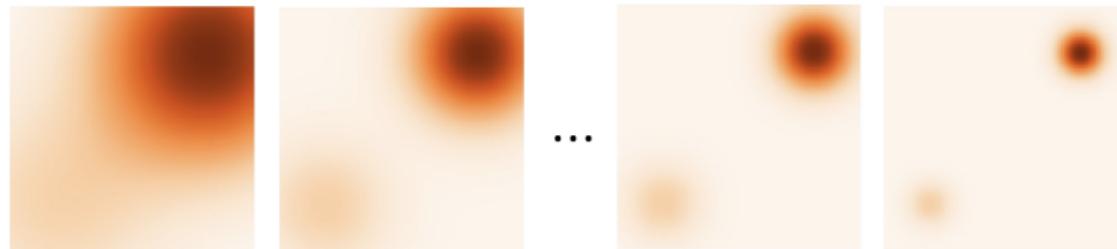


- La función de *score* está ahora bien definida en todas partes.
- Sin embargo, está definida en una distribución diferente.
- Hay un *tradeoff*: suavizar más y obtener mejor *score* frente a suavizar menos y aproximarse mejor a la distribución de los datos.

Ruido multiescala

Para balancear entre la precisión de la aproximación y la estabilidad de la función de *score*, queremos añadir ruido a múltiples escalas:

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

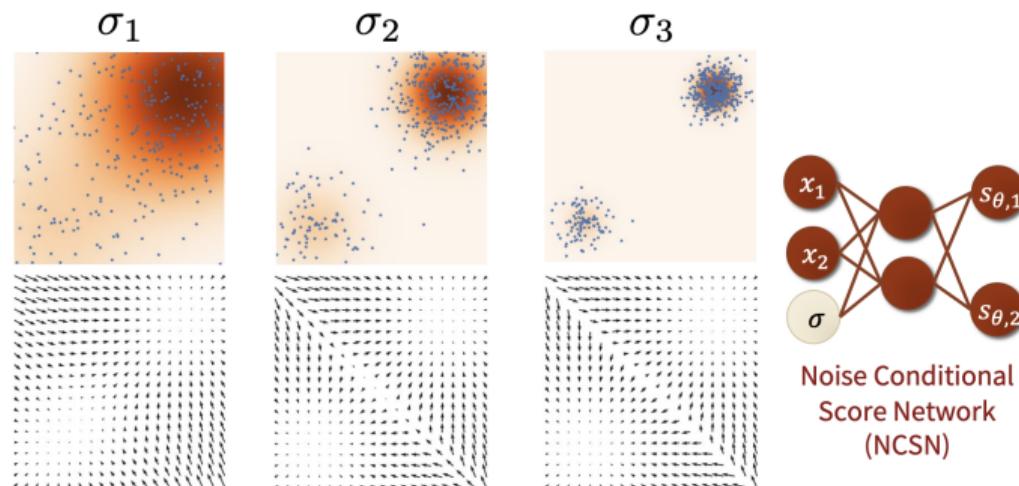


...

- Entrenamos utilizando ruido tanto de potencia alta como baja.
- En las fases iniciales del muestreo, utilizamos un ruido mayor.
- A lo largo del muestreo, reducimos gradualmente el ruido.

Redes de score condicionales al ruido

Implementamos esta estrategia entrenando un modelo de función de score $s_\theta(\mathbf{x}, \sigma)$ que está condicionado al ruido σ .



- Se entrena el mismo modelo para todos los σ . Los parámetros se comparten para todos los niveles de ruido. Por ejemplo, el objetivo pasa a ser:

$$\mathbb{E}_{\sigma \sim \{\sigma_1, \sigma_2, \dots, \sigma_L\}} \left[\frac{1}{2} \mathbb{E}_{\mathbf{x} \sim p, \tilde{\mathbf{x}} \sim q_\sigma(\tilde{\mathbf{x}}|\mathbf{x})} \left[\|\nabla_{\tilde{\mathbf{x}}} \log q_\sigma(\tilde{\mathbf{x}}|\mathbf{x}) - s_\theta(\tilde{\mathbf{x}}, \sigma)\|_2^2 \right] \right].$$

① Función de *score*

② Ajuste de la función de *score*

Divergencia de Fisher

Sliced score matching

Denoising score matching

③ Generación de muestras

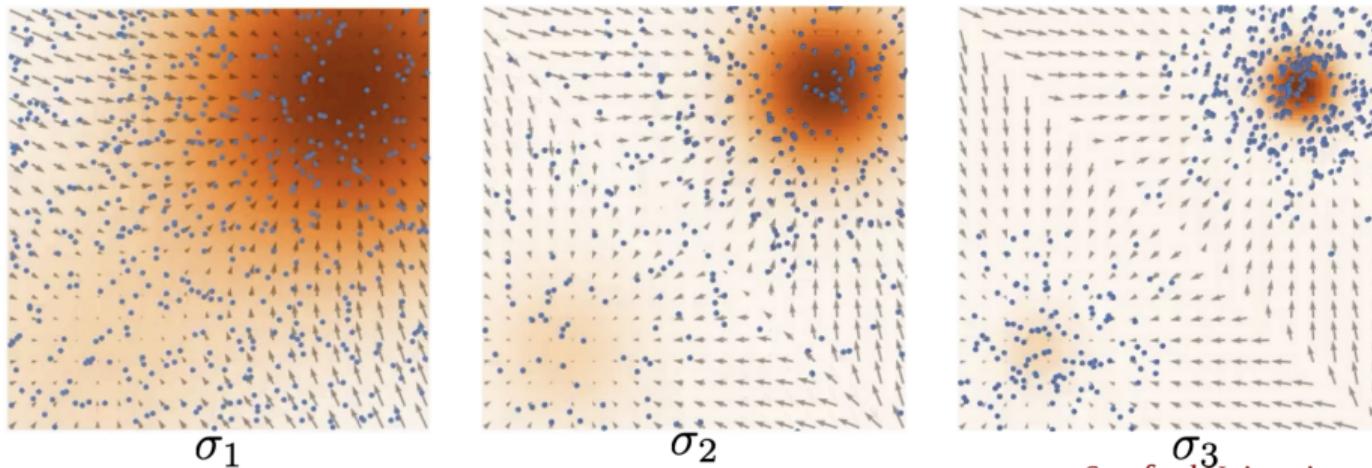
Dinámica de Langevin

Hipótesis de la variedad

Annealed Langevin dynamics

Annealed Langevin dynamics: intuición

Podemos ejecutar la dinámica de Langevin con niveles de ruido decrecientes. En cada nuevo nivel de ruido, empezamos donde lo dejamos con el nivel de ruido anterior.



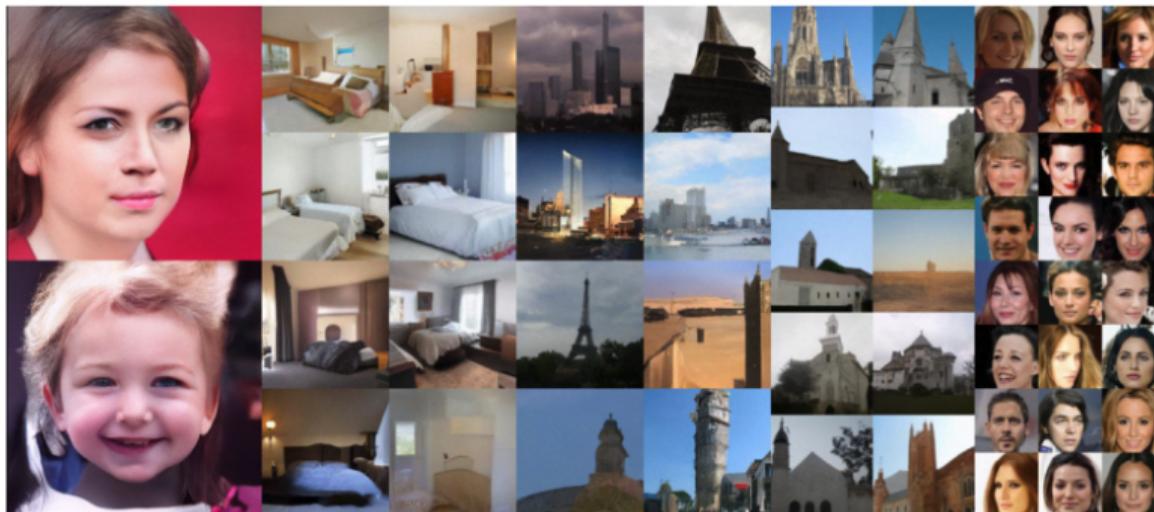
Annealed Langevin dynamics

La dinámica de Langevin atenuada es un proceso de muestreo MCMC que nos permite generar muestras a partir de un modelo de *score* condicional al ruido $s_\theta(\mathbf{x}, \sigma)$.

- Inicializar x_0 a partir de una distribución de ruido.
- Para niveles de ruido $\sigma_l \in \sigma_1, \sigma_2, \dots, \sigma_L$:
 - Setear el paso como $\alpha_t = \beta \cdot \frac{\sigma_l}{\sigma_L}$ para algún $\beta > 0$
 - Para $t = 1, 2, \dots, T$:
 - Muestrear una variable de ruido $\varepsilon_t \sim \mathcal{N}(0, I)$.
 - $\mathbf{x}_t = \mathbf{x}_{t-1} + \frac{1}{2}\alpha_t s_\theta(\mathbf{x}, \sigma_l) + \sqrt{\alpha_t} \varepsilon_t$
 - Setear $\mathbf{x}_0 = \mathbf{x}_T$: iniciar la siguiente ejecución al final de la última ejecución.

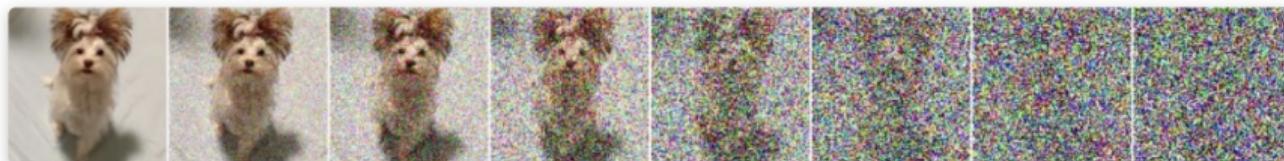
Modelos generativos basados en *score*: muestras

Los modelos generativos basados en *score* pueden generar muestras de alta resolución que en versiones recientes igualan en calidad a las GAN.



Modelos generativos basados en *score*: refinamiento iterativo

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



Perturbing an image with multiple scales of Gaussian noise.

Al muestrear mediante la dinámica Langevin atenuada, deshacemos este ruido: la imagen emerge gradualmente del ruido gaussiano.



Podemos referirnos a este tipo de *denoising* como *refinamiento iterativo*.

Muestreo condicional y generación controlable

Supongamos que conocemos un proceso $p(\mathbf{y}|\mathbf{x})$ que mapea los datos \mathbf{x} a variables auxiliares \mathbf{y} (por ejemplo, etiquetas, imágenes ruidosas). Intentamos modelizar $p(\mathbf{x}|\mathbf{y})$ (una “especie de” inversa).

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

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

- Podemos aprender $\nabla_{\mathbf{x}} \log p(\mathbf{x})$ de forma no condicional.
- El $\nabla_{\mathbf{x}} \log p(\mathbf{y}|\mathbf{x})$ ya es conocido (por ejemplo, un clasificador).
- Así, podemos muestrear de $\nabla_{\mathbf{x}} \log p(\mathbf{x}|\mathbf{y})$ utilizando la dinámica de Langevin.

Modelos generativos basados en score: muestras condicionales

Muestras generadas a partir de CIFAR-10 condicionadas por clase:



Otros ejemplos: condicionamiento de imágenes en color sobre imágenes monocromas, condicionamiento de imágenes ruidosas en imágenes limpias, etc.

Modelos generativos basados en *score*: resumen

Ventajas:

- Pueden entrenar modelos basados en energía sin preocuparse de $Z(\theta)$;
- Producen muestras de alta calidad (como las GAN) con un entrenamiento estable.

Desventajas:

- El muestreo es mucho más lento que en las GAN (es necesario ejecutar el MCMC de Langevin durante miles de pasos);
- No realizan estimaciones de densidad ni proporcionan verosimilitudes;
- Falta de variables latentes para el aprendizaje de representaciones.

¿Podemos abordar algunos de los contras? Sí (modelos difusión).

Referencias

-  C. M. Bishop, *Pattern Recognition and Machine Learning*.
Springer, 2006.
-  Stanford, “CS236 Deep Generative Models.” <https://deepgenerativemodels.github.ioLecture>, 2024.
-  J. M. Tomczak, *Deep Generative Modeling*.
Springer Cham, 2024.