Variational Autoencoders

Métodos Generativos, curso 2025-2026

Guillermo Iglesias, guillermo.iglesias@upm.es
Jorge Dueñas Lerín, jorge.duenas.lerin@upm.es
Edgar Talavera Muñoz, e.talavera@upm.es

5 de noviembre de 2025

Escuela Técnica Superior de Ingeniería de Sistemas Informáticos | UPM

Contenidos

Contenidos

Auto-encoders Variacionales (VAEs)

Motivación

Los autoencoders tienen un gran problema: no son buenos **generadores** de datos.

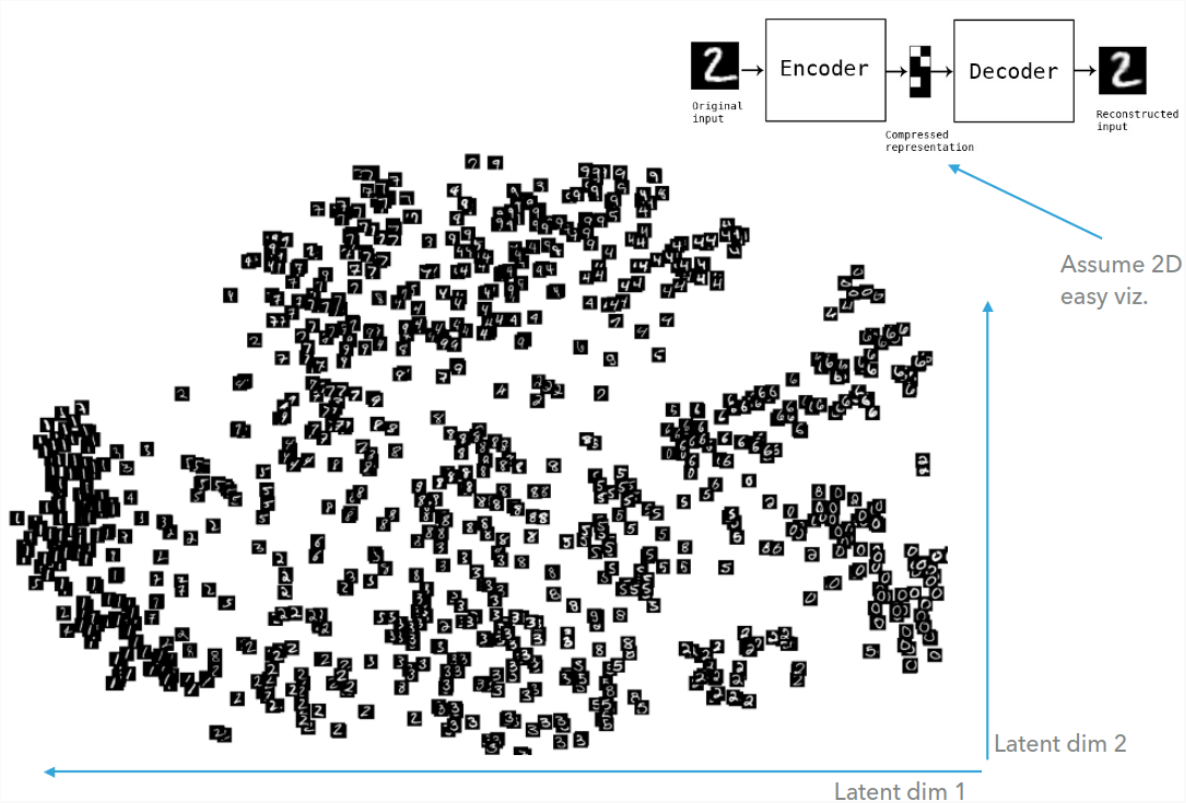
¿Por qué?

Motivación

Pensemos en un ejemplo sencillo: la reconstrucción de imágenes del dataset MNIST.

¿Cómo pensáis que será el espacio latente (representación en el *bottleneck*)?

Motivación

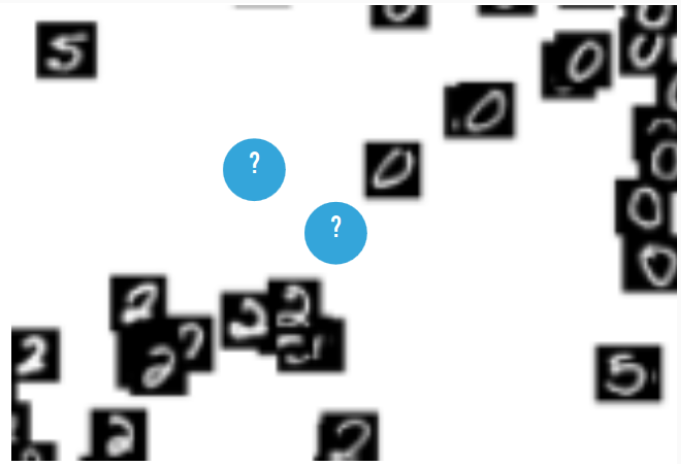


Ejemplo de espacio latente con el dataset MNIST (fuente).

Motivación

Esta representación presenta determinados problemas: al no ser continua, tendremos problemas cuando la entrada sea ligeramente distinta a los datos con los que se entrenó el autoencoder:

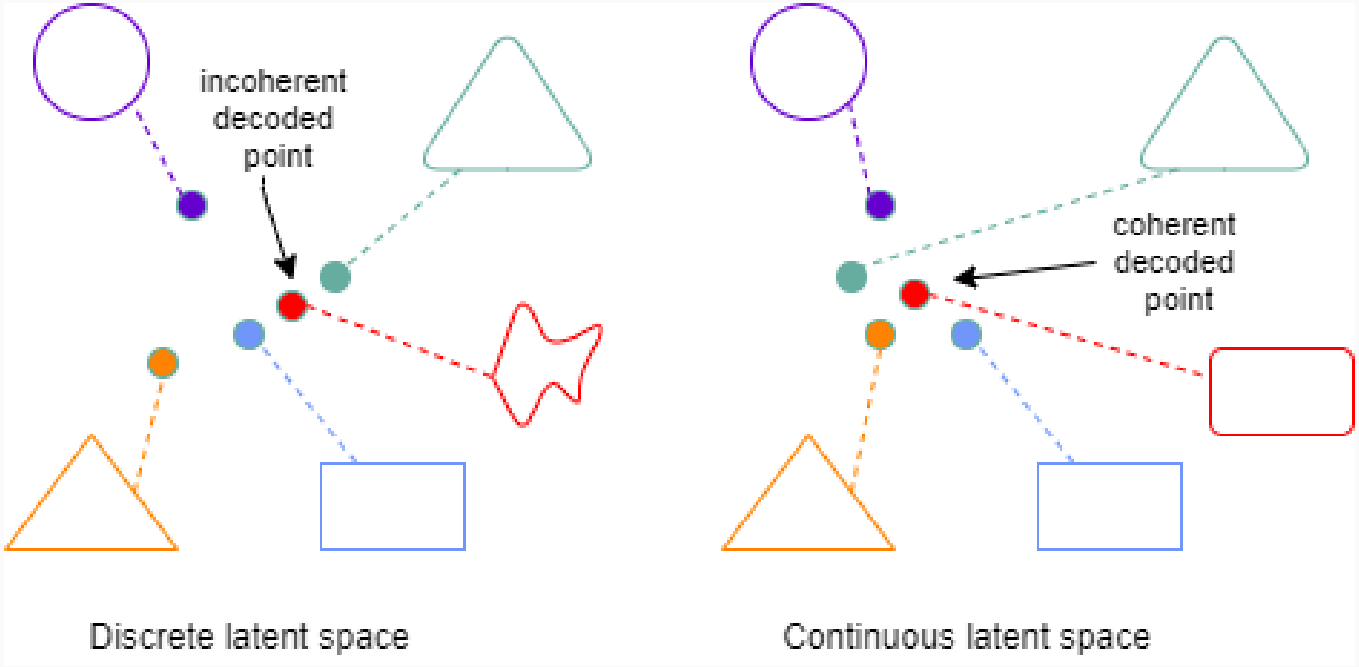
¿Qué ocurrirá cuando la entrada sean imágenes que generen códigos latentes entre medio de las muestras de entrenamiento?



Ejemplo de problemas al generar nuevas muestras (fuente).

Motivación

Esta imagen lo muestra de forma intuitiva:

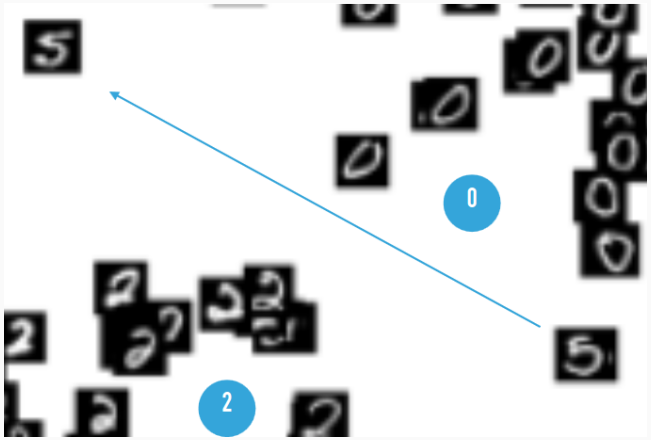


Ejemplo de problemas al generar *muestrear* de un espacio latente no continuo (fuente).

Motivación

¿Qué es lo deseable?

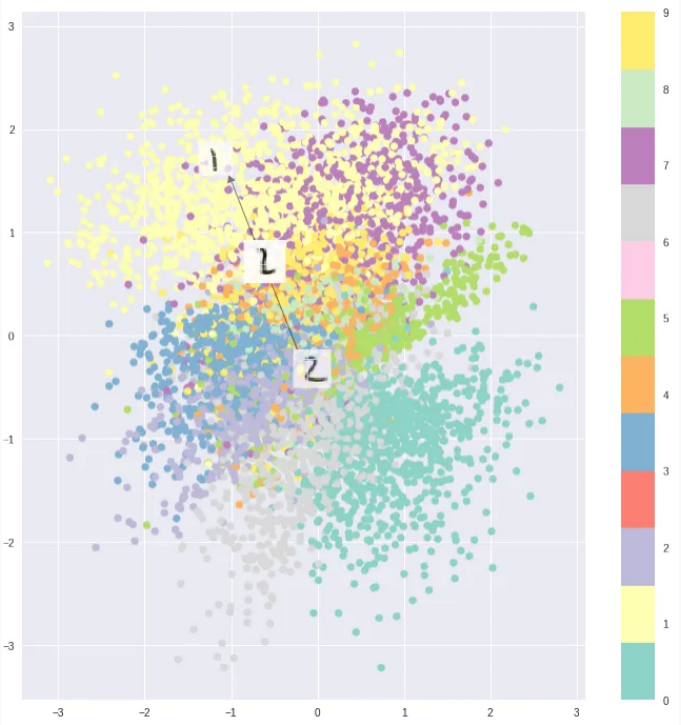
- Un espacio latente **continuo** y **ordenado**
- en el que poder obtener muestras parecidas a los datos de la entrada aunque no coincidan exactamente con alguno de los de entrenamiento
- y en el que poder **interpolar** entre distintos espacios latentes para obtener **nuevas muestras**



Ejemplo de problemas al generar nuevas muestras (fuente).

Motivación

Ejemplo de interpolación con un espacio latente **contínuo** y **ordenado**:

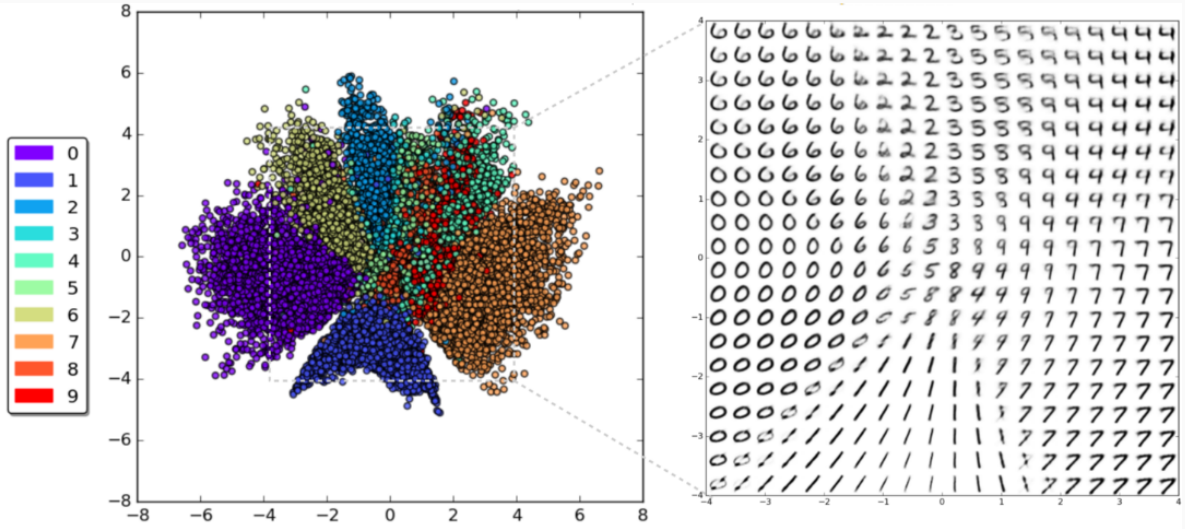


Ejemplo de interpolación (fuente).

Motivación

¿Cómo lo logramos?

Modificando ligeramente la arquitectura del auto-encoder para conseguir un espacio latente **contínuo** y **ordenado**.



Ejemplo de espacio latente **contínuo** y **ordenado** de las muestras generadas a partir de su muestreo. (fuente).

Variational Autoencoders (VAEs)

¿Qué son los Auto-Encoders Variacionales (VAEs)? Son una variante de los autoencoders [?] que permiten la generación de datos sintéticos.

- Combinan redes neuronales con distribuciones de probabilidad.
- Permiten que los datos generados sigan el mismo patrón que los datos de entrada.

Así, la red aprende los parámetros de una distribución de probabilidad.

- Construyen explícitamente un **espacio latente continuo** y **ordenado**.
- No una función arbitraria como en las redes neuronales convencionales.

¿Cómo funcionan?

En los Variational Autoencoders (VAEs), el espacio latente está definido por **dos vectores de tamaño** n:

- $\vec{\mu} = (\mu_1, \ldots, \mu_n)$: Vector de **medias**.
- $\vec{\sigma} = (\sigma_1, \ldots, \sigma_n)$: Vector de **desviaciones estándar**.

Forman un vector de distribuciones normales:
$(N(\mu_1, \sigma_1), \ldots, N(\mu_n, \sigma_n))$.

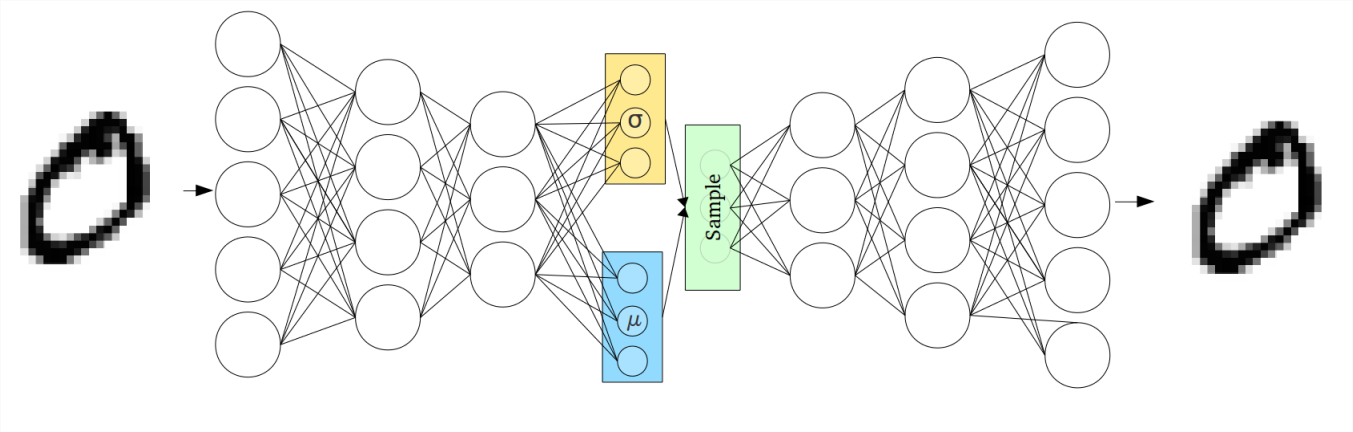
- Cada μ_i controlará el centro aproximado donde codificar los datos de entrada.
- Cada σ_i controlará cuánto pueden desviarse en cualquiera de sus muestras.

Con este modelo, el decodificador asocia áreas completas (no solo puntos individuales) a variantes ligeras de la misma salida.

- Esto resulta en un espacio interpolado mucho más suave.
- Es capaz de producir nuevas salidas que comparten propiedades.

Estructura

El espacio latente está definido por **dos vectores de tamaño** *n*:

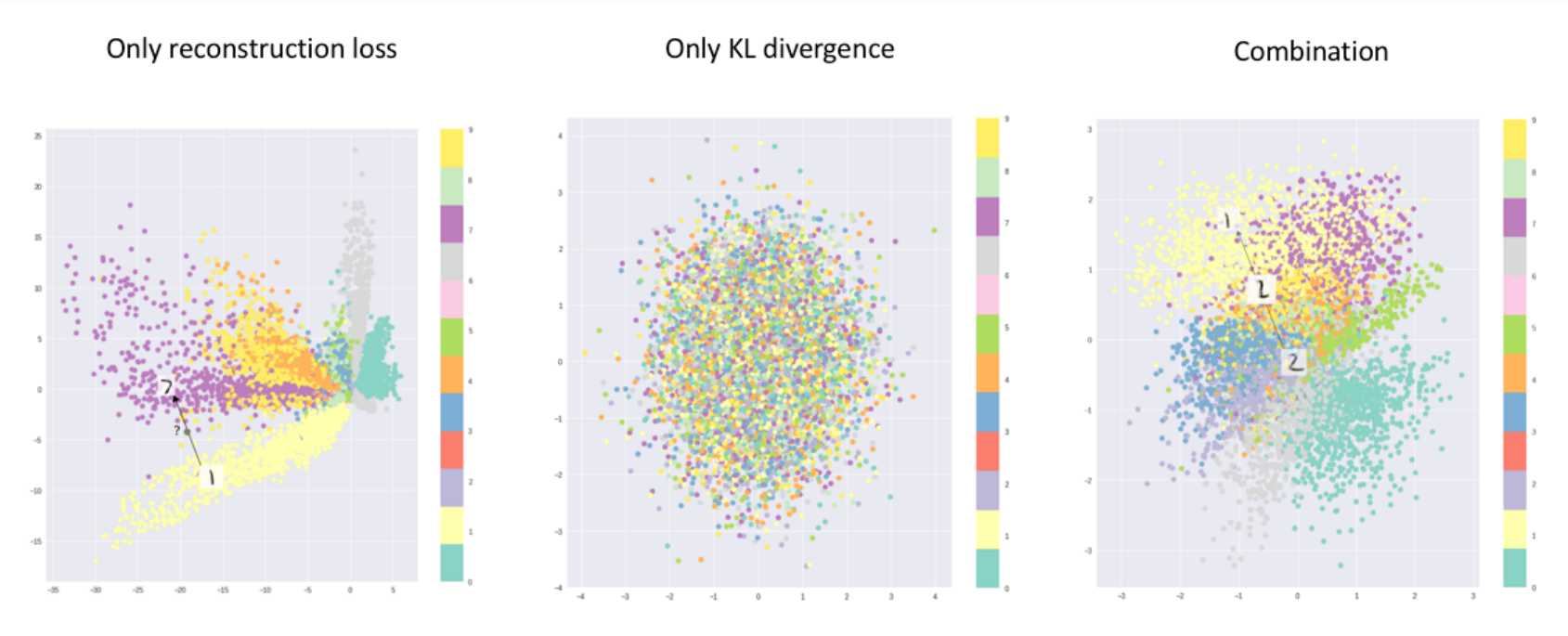


Luego debemos ajustar las funciones de pérdida individualmente de tal manera que:

- Una **función de pérdida tradicional** que calcula la **diferencia** con el objeto generado.
- La **divergencia KL (Kullback-Leibler)** entre la distribución latente aprendida y la distribución anterior (*prior distribution*), que actúa como término de regularización.
- Se suele usar $\mathcal{N}(0, 1)$

Estructura

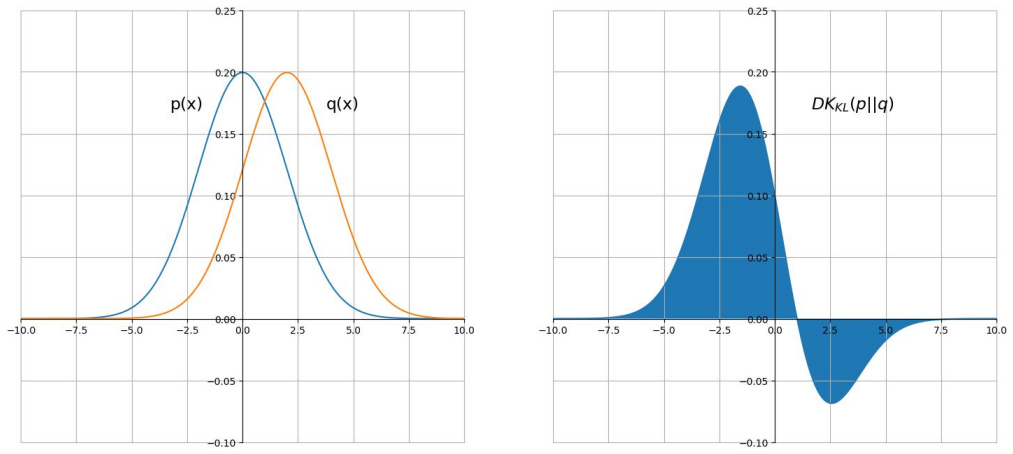
¿Por qué necesitamos las pérdidas de reconstrucción y la divergencia KL?



Ejemplo con diferentes términos de la función de pérdidas (fuente).

Divergencia KL (I)

Mide la diferencia entre dos distribuciones de probabilidad.



Por ejemplo, en las distribuciones de la figura tenemos dos distribuciones:

- Una distribución normal y conocida $p(x)$.
- Una distribución normal y desconocida $q(x)$.

Es una **divergencia, no una distancia**, ya que no es simétrica.

Divergencia KL (II)

Forzando una distribución normal estándar ($\mu = 0$ y $\sigma = 1$) para nuestra distribución de datos, tenemos que la divergencia KL se puede calcular como:

$$KL = \sum_{i=1}^{n} \sigma_i^2 + \mu_i^2 - log(\sigma_i) - 1$$

Nuestra función de pérdida consistirá en dos términos:

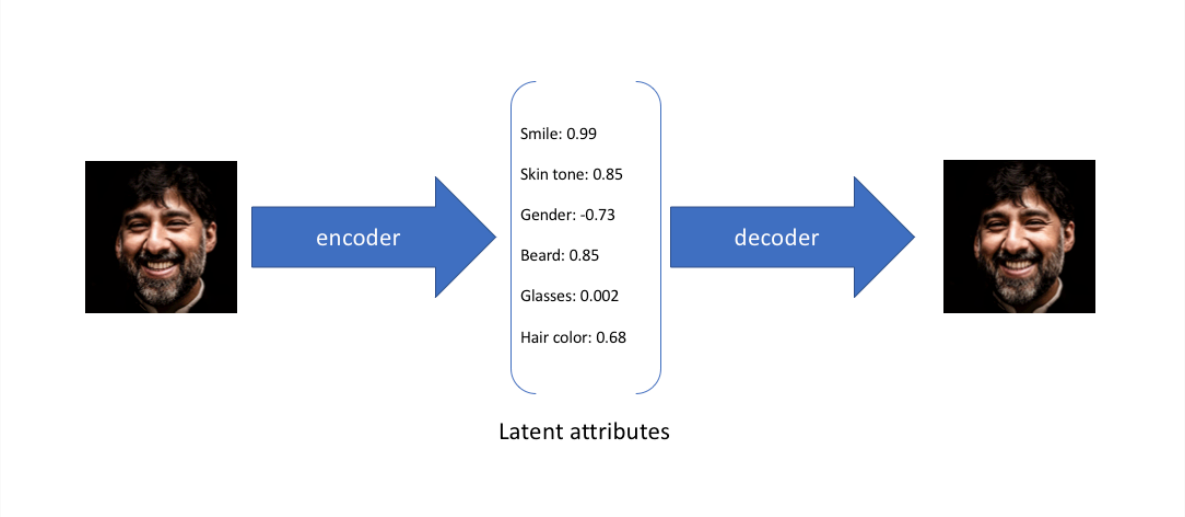
- Función de pérdida tradicional $\mathcal{L}_r$: Ajustará los datos de salida.
- Función de divergencia *KL*: Ajustará el espacio latente a la distribución estándar.

Por lo tanto, la expresión de la función de pérdida será:

$$\mathcal{L}(y, \hat{y}) = \mathcal{L}_r(y, \hat{y}) + \mathcal{L}_{\text{KL}}(y, \hat{y})$$

Aplicaciones

Una aplicación de los VAEs es la de poder generar muestras teniendo cierto control sobre lo que generamos.

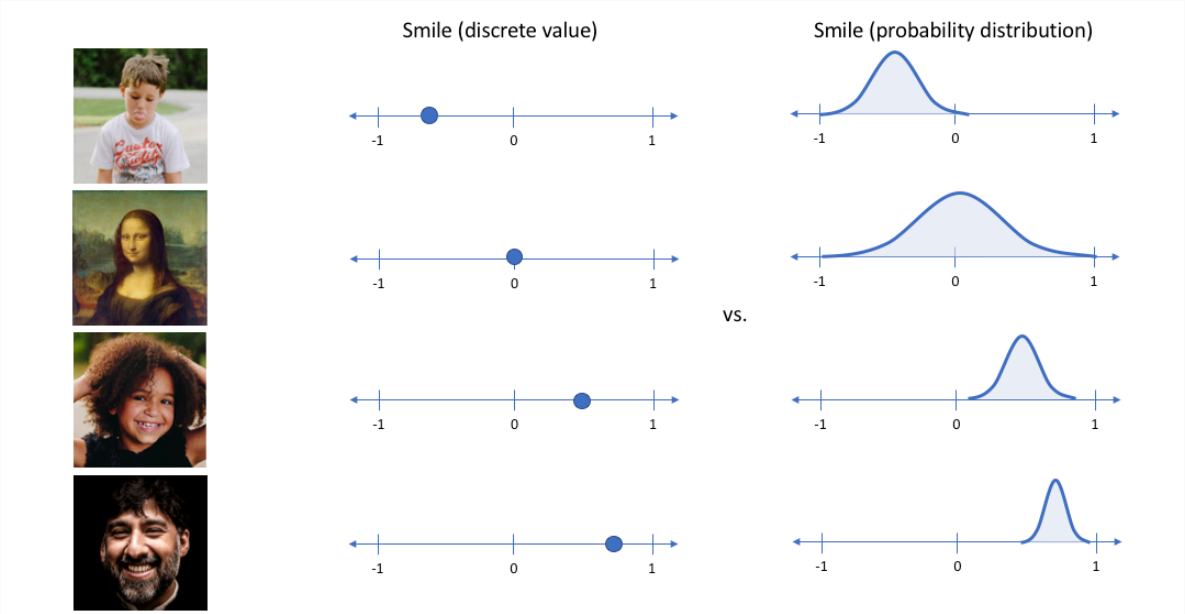


Ejemplo de generación de caras controlando los atributos (fuente).

En los VAEs el espacio latente está entrelazado. Conditional VAE. Se verá en CGANs.

Aplicaciones

¿Cómo se comportarían un AE y un VAE a la hora de caracterizar la sonrisa de una imagen?

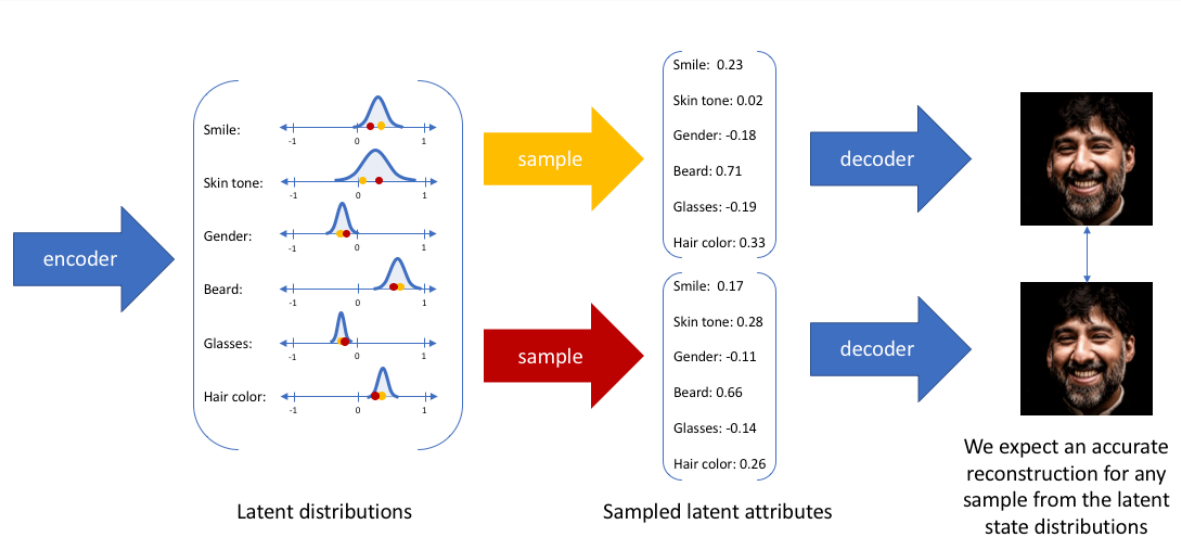


Ejemplo de espacio latente en AEs vs VAEs (fuente).

Aplicaciones

Al contrario que con los AEs, con los VAEs podemos obtener muestras nuevas diferentes a las de entrenamiento.

Por ejemplo, si muestreamos dos veces con valores similares, deberemos obtener muestras similares:



Ejemplo de generación de dos caras similares controlando los atributos (fuente).

Bonus: ¿Aprendizaje supervisado o no supervisado?

Tradicionalmente, se han clasificado como **aprendizaje no supervisado**.

- Después de todo, no trabajan con datos etiquetados.
- ¡Pero no puedes optimizar autoencoders sin la retroalimentación de la propia reconstrucción!

En el **aprendizaje supervisado**, se aprende con retroalimentación de los datos.

- Se espera que, al proporcionar algunos ejemplos, el algoritmo descubra la función que mapea las entradas a las salidas deseadas con el menor error.

Yann LeCun inventó el término **aprendizaje auto-supervisado** para hablar sobre estos modelos.

I now call it “self-supervised learning”, because “unsupervised” is both a loaded and confusing term. [...] Self-supervised learning uses way more supervisory signals than supervised learning, and enormously more than reinforcement learning. That’s why calling it “unsupervised” is totally misleading.

Yann LeCun - Recent Advances in Deep Learning (2019)

Generación de imágenes con Variational Autoencoders

Recursos

- Diapositivas de Moodle
- Google Collaboratory
- Deep Learning Book (https://www.deeplearningbook.org/)
- https://www.pyimagesearch.com/blog
- https://machinelearningmastery.com/blog

Referencias i