

A human brain is shown in profile, facing right. It is covered in vibrant, multi-colored paint splashes and splatters. The colors include bright yellow, orange, red, magenta, blue, green, and black. The paint appears to be dripping and splashing out from the brain, creating a dynamic and artistic representation of neural activity or creative thought.

Autocodificadores II

Clase 11

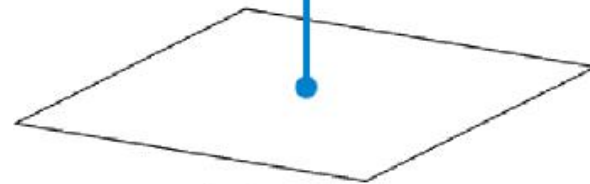
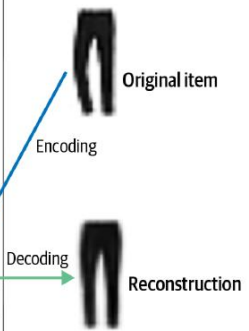
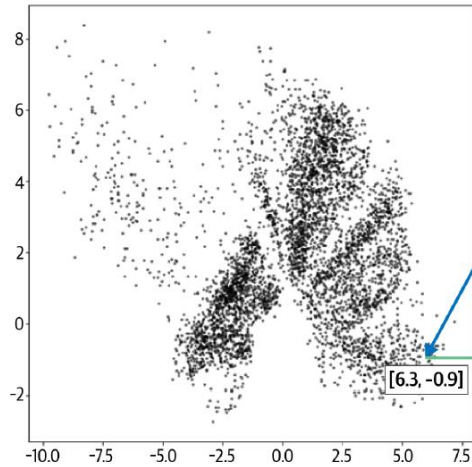
Dra. Wendy Aguilar

Modelos Generativos Profundos

UN ENFOQUE DESDE LA
CREATIVIDAD
COMPUTACIONAL

La clase pasada aprendimos ...

Autocodificadores Variacionales

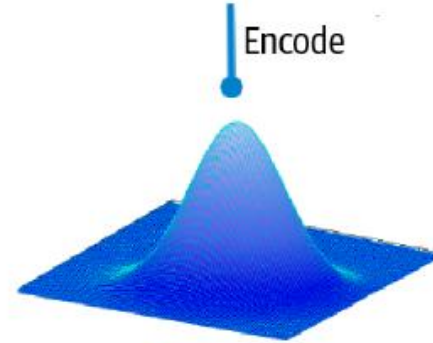


Autoencoder
1986

Aprender a mapear cada imagen directamente a un punto en el espacio latente. $x \rightarrow z$



Encode



Variational autoencoder
2013

Aprende a mapear cada imagen en una distribución de probabilidad normal multivariada alrededor de un punto en el espacio latente.



Se tiene que modificar el codificador y la función de pérdida

Autocodificador determinista vs variacional

Autocodificadores deterministas vs variacionales con Fashion-MNIST:

- [notebook aes vs vaes fashion mnist.zip](#)

Contiene:

```
, autoencoder_fashion_mnist_version1.ipynb  
  vae_fashion_mnist.ipynb  
  autoencoder_vs_vae_fashion_mnist.ipynb  
  decoder_det_fashion.keras  
  decoder_vae_fashion.keras  
  encoder_det_fashion.keras  
  encoder_vae_fashion.keras
```

Nota: Recomendando ejecutarlo en cpu, ya que no es necesario gastar recursos de gpu.



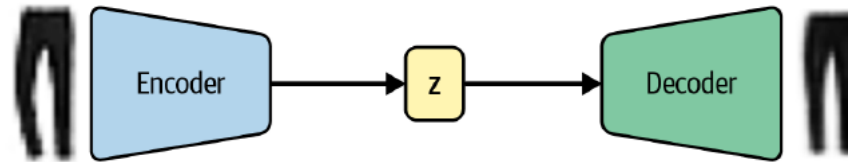
Profundizando un poco más en sus diferencias ...

Autocodificador determinista

- No es un modelo probabilístico generativo
- Su definición es puramente funcional:

$$\text{Encoder: } z = f_{\phi}(x)$$

$$\text{Decoder: } \hat{x} = g_{\theta}(z)$$



En los modelos generativos
queremos algo más que compresión:
queremos **modelar el proceso generativo** de los datos.

Profundizando un poco más en sus diferencias ...

Modelos generativos latentes



Imágenes, audio, texto, etc.

La representación interna que el modelo usa para generar.

En modelo generativo latente con datos x y variables latentes z se define como:

$$p(x, z) = p(x|z) p(z)$$

La probabilidad conjunta de ver un dato x y un latente z se obtiene:

$p(x, z)$: ¿Qué tan probable es que esa imagen en específico y ese código latente formen parte de la distribución que el modelo está definiendo?

La probabilidad del dato x dado el latente z .

Ejemplo
 x : Imagen de un 5
 z : código latente (concepto) de un 5

La probabilidad de que la imagen del 5 corresponda al código latente del 5.

La probabilidad de elegir un cierto código latente z antes de ver los datos.

El modelo tiene un espacio latente con distintos "conceptos": "5", "7", "3", "S", etc.

El **prior** $p(z)$ asigna una probabilidad a cada uno de esos códigos latentes.



Espacio latente

Profundizando un
poco más en sus
diferencias ...

Modelos generativos latentes

Modelos generativos latentes

$$p(\mathbf{x}, \mathbf{z}) = p(\mathbf{x}|\mathbf{z}) p(\mathbf{z})$$

Adicionalmente nos interesa saber es el posterior:

$$q_{\phi}(\mathbf{z}|\mathbf{x}) \approx p(\mathbf{z}|\mathbf{x})$$

- Porque nos permite, dado un dato real \mathbf{x} , inferir qué representaciones latentes \mathbf{z} lo explican.
- Sin este paso, no podríamos conectar el mundo observado con el espacio generativo del modelo.

Profundizando un poco más en sus diferencias ...

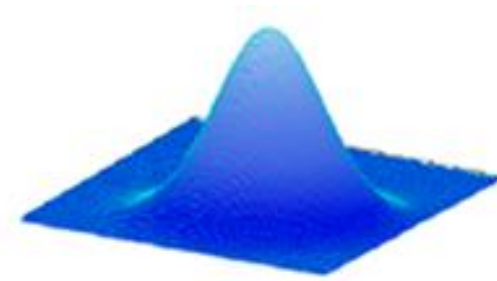
Modelos generativos latentes

$$p(\mathbf{x}, z) = p(\mathbf{x}|z) p(z)$$

VAEs

Prior: $p(z) = \mathcal{N}(0, I)$

Un prior (o distribución a priori) es la **suposición inicial** que hacemos sobre cómo se distribuyen las variables latentes z , antes de ver los datos.



Likelihood (decoder): $p_{\theta}(\mathbf{x}|z)$

La probabilidad de que la imagen del 5 corresponda al código latente del 5.

Parametrizada por una red neuronal que genera datos a partir de z .

θ son los **pesos de todas las capas que definen el decodificador** (densas, convolucionales, etc.),

Posterior aproximado (encoder):

$$q_{\phi}(z|\mathbf{x}) \approx p(z|\mathbf{x})$$

Ya que el posterior real es intratable.

- Introducimos una distribución alternativa q_{ϕ} , donde ϕ son los parámetros del codificador.
- Elegimos una familia simple de distribuciones (ej. Normal multivariada estándar).

$$p(z|\mathbf{x}) = \frac{p_{\theta}(\mathbf{x}|z)p(z)}{p(\mathbf{x})}$$

$$p(\mathbf{x}) = \int p_{\theta}(\mathbf{x}|z)p(z) dz$$

Es intratable porque esa integral sobre todos los posibles z no se puede resolver de manera exacta en espacios de alta dimensión.

Limitaciones de los VAEs

- El **espacio latente es continuo** y se regulariza con el prior gaussiano.
- Esto genera dos **problemas** prácticos:
 1. **Reconstrucciones borrosas** → porque el decodificador debe promediar sobre distribuciones continuas.



2. **Poco poder para modelar secuencias complejas** (audio, texto, video) → en estos dominios, las representaciones suelen ser naturales en forma discreta (fonemas, palabras, símbolos).

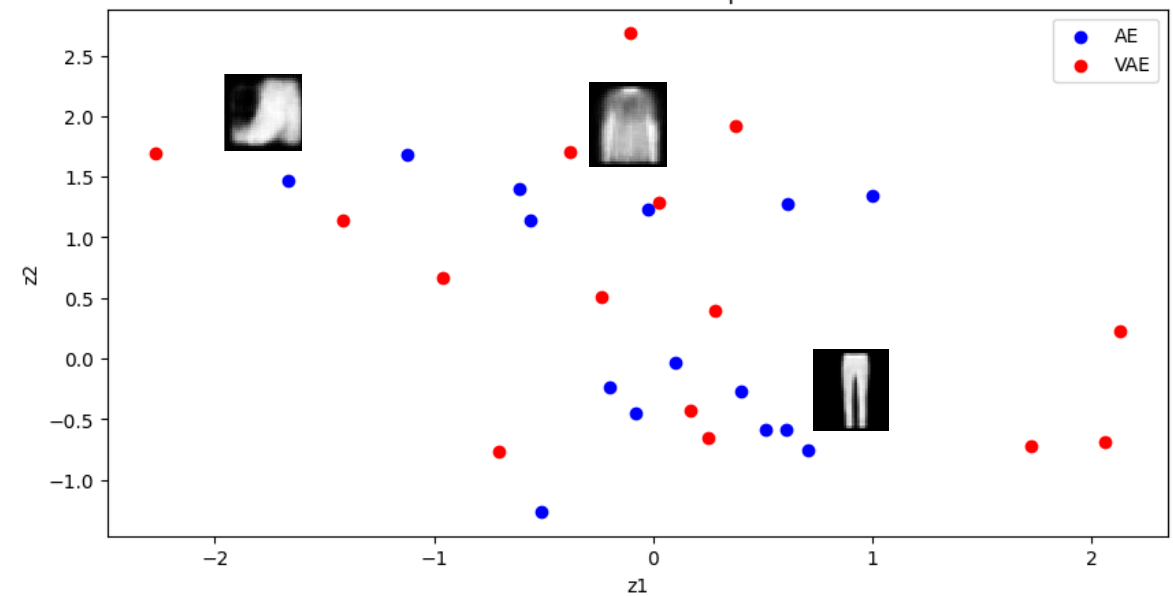
Estrategias de control generativo

Muestreo aleatorio

- Permite explorar libremente el espacio latente.

```
# ---- AE determinista ----  
z_random_ae = np.random.normal(size=(num_samples, latent_dim))  
reconstructions_ae = decoder_determinista.predict(z_random_ae)  
  
# ---- VAE ----  
z_random_vae = np.random.normal(size=(num_samples, latent_dim))  
reconstructions_vae = decoder_variacional.predict(z_random_vae)
```

- Cada vez que muestreemos obtenemos algo diferente.
- **Desventaja:** no tenemos control sobre el resultado → nunca sabemos qué clase de imagen aparecerá.



Estrategias de control generativo

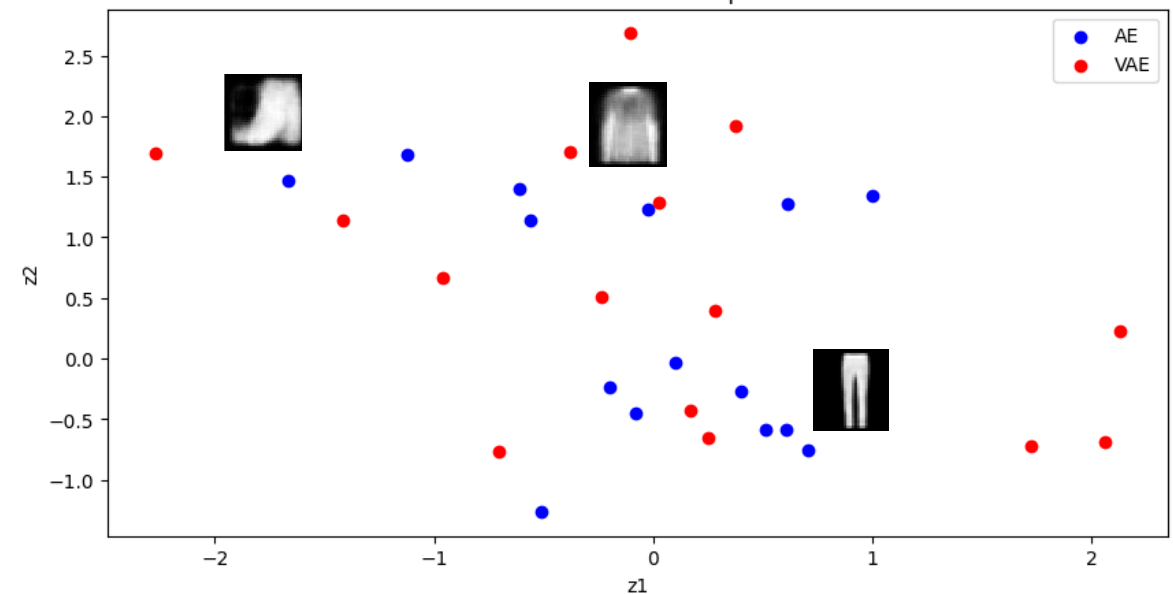
Muestreo aleatorio

- Permite explorar libremente el espacio latente.

```
# ---- AE determinista ----
z_random_ae = np.random.normal(size=(num_samples, latent_dim))
reconstructions_ae = decoder_determinista.predict(z_random_ae)

# ---- VAE ----
z_random_vae = np.random.normal(size=(num_samples, latent_dim))
reconstructions_vae = decoder_variacional.predict(z_random_vae)
```

- Cada vez que muestreemos obtenemos algo diferente.
- **Desventaja:** no tenemos control sobre el resultado → nunca sabemos qué clase de imagen aparecerá.



Necesidad de control

- Para **aplicaciones creativas** no basta con muestrear al azar.
- Requerimos **mecanismos** adicionales **que guíen la generación hacia atributos deseados** (ej. clase, estilo, características visuales).

Estrategias de control generativo

Estrategias generales aplicables a AE deterministas y VAE

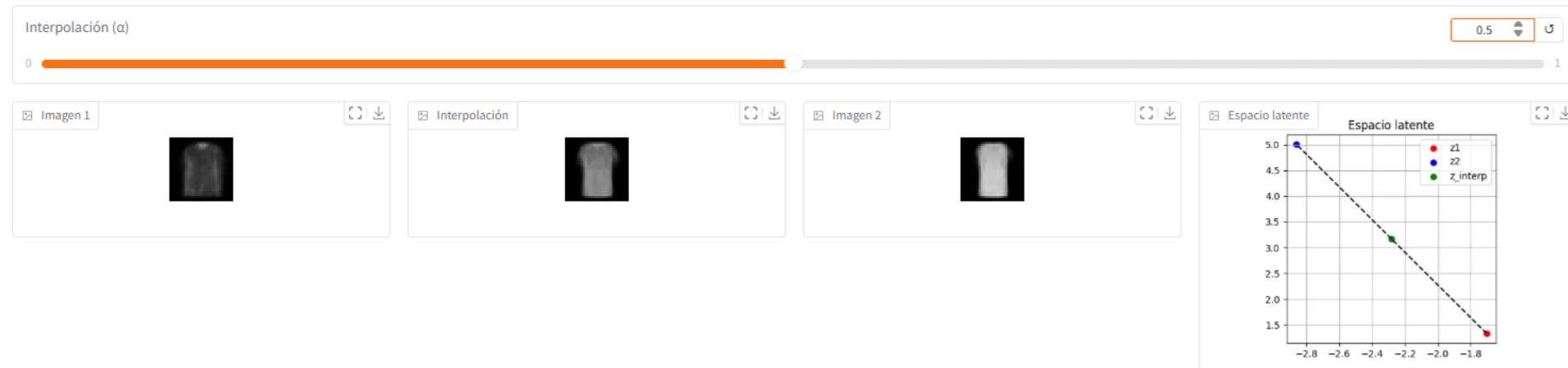
1. Interpolación en el espacio latente

- Generar imágenes intermedias entre dos puntos latentes (interpolaciones lineales).
- Útil para visualizar transiciones suaves (ej. de “camisa” a “vestido”).

interpolacion_ae_y_vae_fashion.ipynb



Interpolación latente entre dos prendas de Fashion-MNIST usando un Autoencoder Determinista



Estrategias de control generativo

Estrategias generales aplicables a AE deterministas y VAE

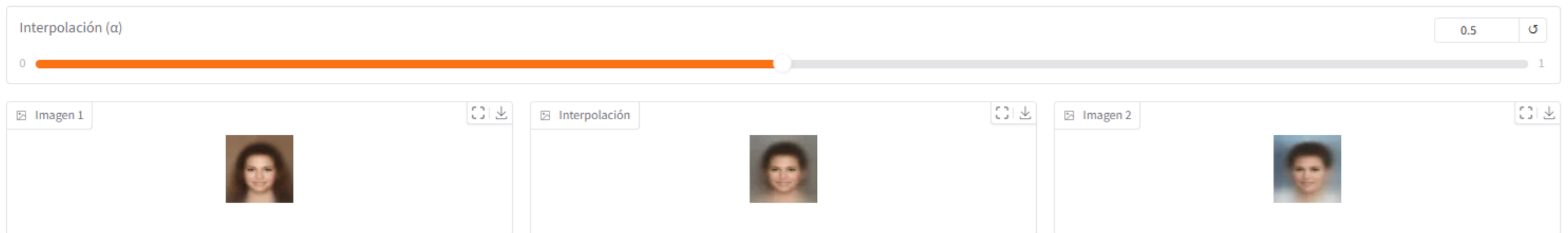
1. Interpolación en el espacio latente

- Generar imágenes intermedias entre dos puntos latentes (interpolaciones lineales).
- Útil para visualizar transiciones suaves (ej. de “camisa” a “vestido”).

interpolacion_vae_faces.ipynb



Interpolación latente entre dos rostros de CelebA usando un VAE

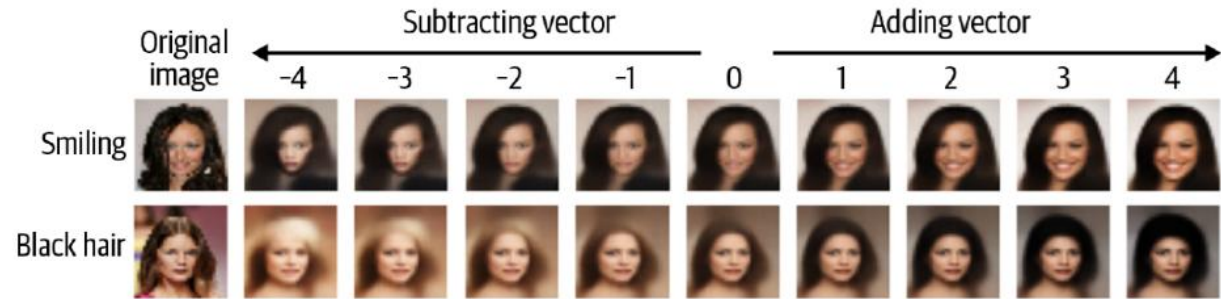


Estrategias de control generativo

Estrategias generales aplicables a AE deterministas y VAE

2. Edición en el espacio latente

- Ejemplo: sumar la "dirección de sonrisa" para transformar una cara neutra en sonriente o seria.



1. Representar las imágenes en el espacio latente

z_{neutro} = vector latente de una cara neutra.

z_{sonrisa} = vector latente de la misma persona sonriendo.

2. Encontrar direcciones latentes (atributos)

Substracción de pares de imágenes

- Si tienes ejemplos de caras con y sin sonrisa:

$$\vec{d}_{\text{sonrisa}} = \text{mean}(z_{\text{sonrisa}}) - \text{mean}(z_{\text{neutro}})$$

- Esto da la "dirección de sonrisa" en el espacio latente.

3. Aplicar la edición

Una vez que tienes la dirección \vec{d}_{sonrisa} , puedes modificar un vector latente z :

$$z' = z + \alpha \cdot \vec{d}_{\text{sonrisa}}$$

- Si $\alpha > 0$, haces la cara más sonriente.
- Si $\alpha < 0$, la haces más seria.

Finalmente, pasas z' al **decodificador/generador** → obtienes la nueva imagen editada.

vae_faces.ipynb

Estrategias de control generativo

Estrategias generales aplicables a AE deterministas y VAE

3. Condicionamiento

Conditional Variational Autoencoder (CVAE)

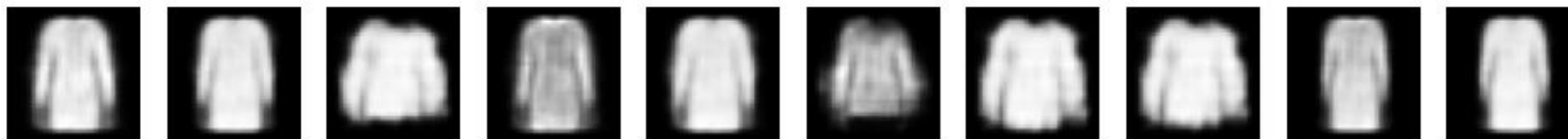
- Incorporar etiquetas o atributos al encoder y decoder.
- Permite controlar explícitamente características como clase, color o estilo.
- La diferencia principal no está en la arquitectura base, sino en cómo incorporamos las etiquetas o atributos para guiar la generación.

- En un VAE estándar, el espacio latente aprende a organizar las imágenes según similitudes estadísticas, pero sin ninguna garantía de alinearse con atributos semánticos (ej. "sonrisa", "género", "gafas").
- En un CVAE, al pasar la etiqueta:
 - El encoder aprende $q(z|x,y)$, cómo representar en el espacio latente una imagen sabiendo su clase o atributo.
 - El decoder aprende $p(x|z,y)$: cómo reconstruir la imagen dado el vector latente y la condición.

Generación condicional: clase 0 - T-shirt/top



Generación condicional: clase 3 - Dress



`cvae_fashion_mnist.ipynb`

