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 (AEs)

¿Qué son los autoencoders?

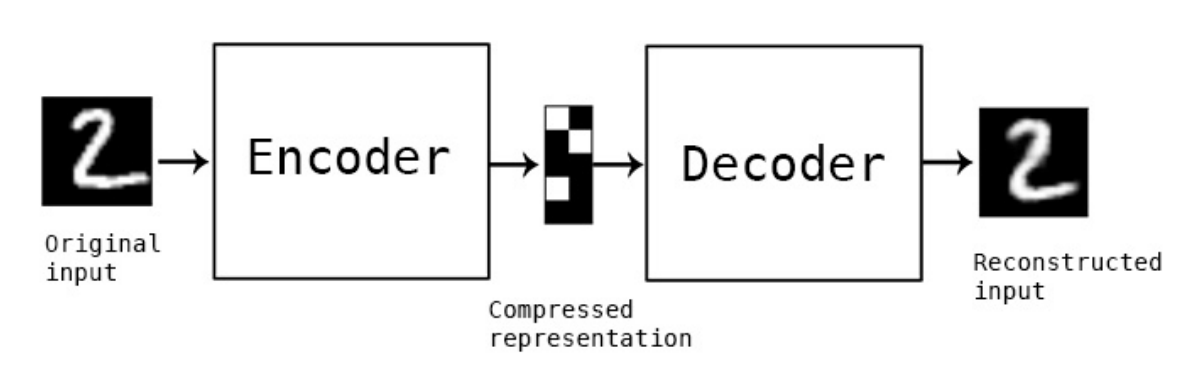
Un tipo de red neuronal que puede aprender a comprimir y luego reconstruir datos.

- Un autoencoder es un tipo de red neuronal utilizada en tareas de aprendizaje no supervisado.
- Su objetivo es aprender una representación **compacta** de los datos de entrada.
- Consiste en dos partes principales: el codificador y el decodificador.

¿Qué son los autoencoders?

Para operar, constan de dos componentes que se **entrenan al mismo tiempo**:

- **Codificador**: Transforma los datos de entrada en una representación de menor dimensión.
- **Decodificador**: Toma esta representación y reconstruye los datos originales.



Estructura básica de un autoencoder.

¿Para qué se emplean?

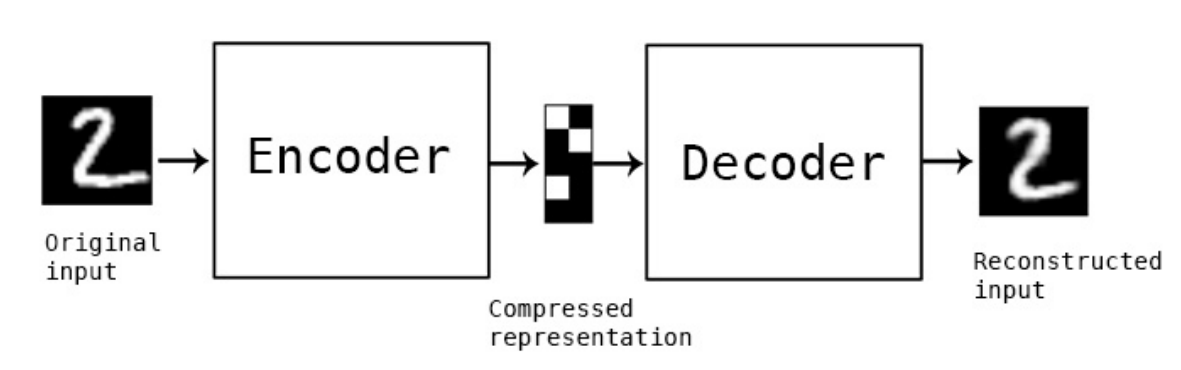
Son muy útiles en una amplia variedad de aplicaciones.

- Reducción de dimensionalidad de datos de alta dimensión.
- Eliminación de ruido en señales.
- Detección de anomalías.
- Generación de nuevos datos, similares a los datos de entrada[1].

[1] Ian Goodfellow menciona que son la **primera red generativa**.

¿Cómo funcionan?

A efectos prácticos, como cualquier otra red neuronal. Únicamente cambian la estructura del modelo, las entradas y salidas, y la función de pérdidas.

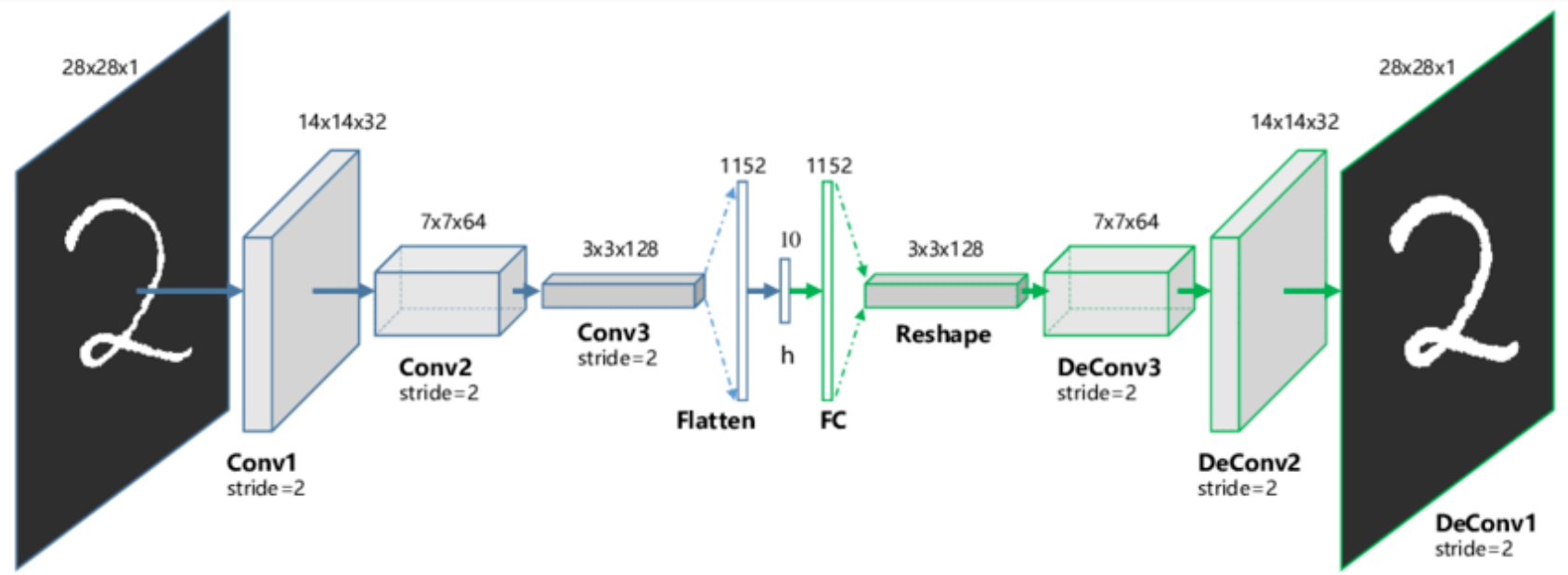


Su entrenamiento se realiza mediante el descenso de gradiente.

- La **retropropagación** ajustará la salida de la red con respecto a la entrada.
- El entrenamiento tenderá a eliminar los datos que contribuyen menos a la salida, es decir, a encontrar una representación **comprimida** de los datos de entrada.

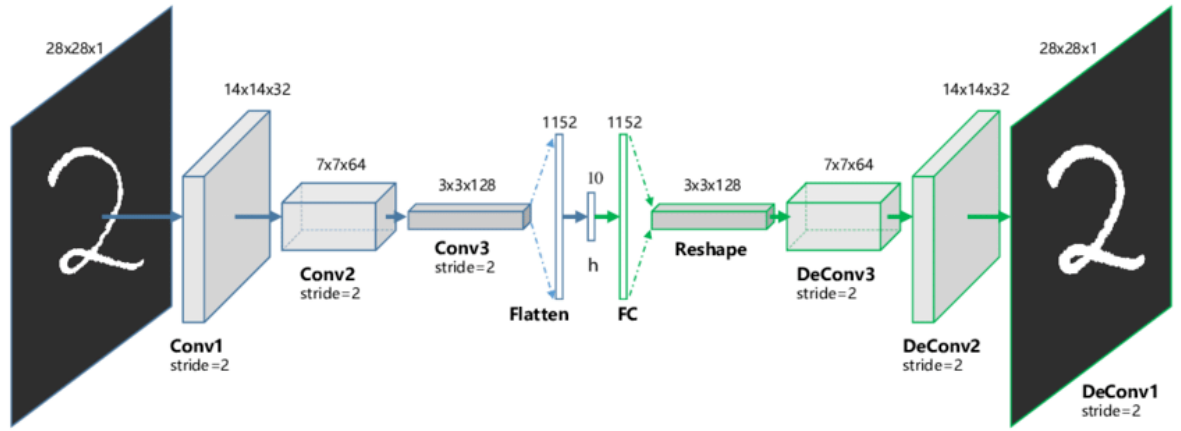
¿Cómo funcionan?

Un ejempo real de un autoencoder podría ser el siguiente:



Esquema de un Autoencoder convolucional (fuente).

¿Cómo funcionan?



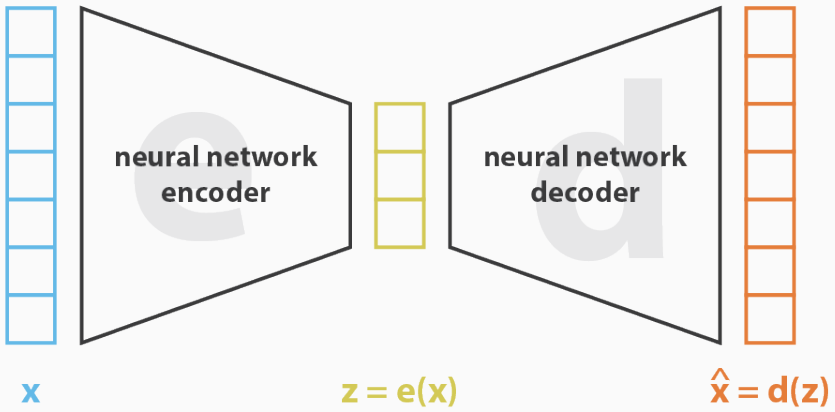
Podemos observar que se compone de:

- El codificador o ***encoder***: calcula una representación comprimida de los datos de entrada.
- El ***bottleneck***: la información que representa los datos de entrada de forma comprimida.
- El decodificador o ***decoder***: trata de reconstruir los datos de entrada a partir de la información del *bottleneck*.

¿Cómo se entrenan?

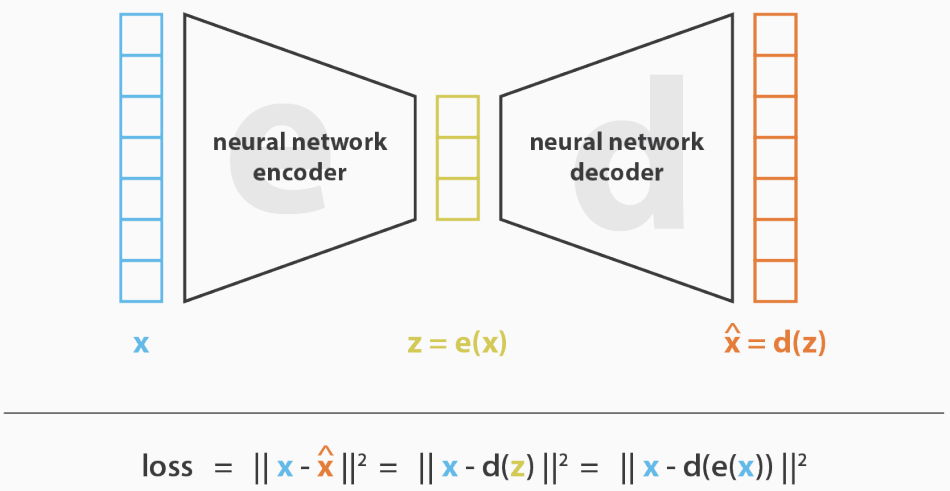
Se entrenan de igual forma que las redes neuronales tradicionales.

La función de pérdidas suele ser parecida a la que se emplearía en problemas de **regresión**, pues, al final, se trata de eso mismo.



$$\text{loss} = \| x - \hat{x} \|^2 = \| x - d(z) \|^2 = \| x - d(e(x)) \|^2$$

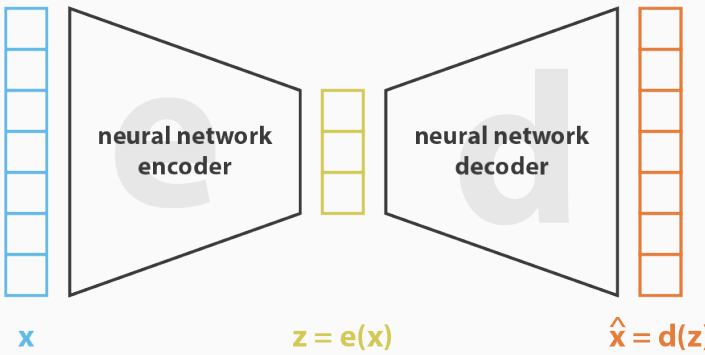
¿Cómo se entrenan?



Podemos observar que se compone de:

- El codificador o ***encoder***: calcula una representación comprimida de los datos de entrada.
- El ***bottleneck***: la información que representa los datos de entrada de forma comprimida.
- El decodificador o ***decoder***: trata de reconstruir los datos de entrada a partir de la información del *bottleneck*.

¿Por qué necesitamos Autoencoders?



$$\text{loss} = \| x - \hat{x} \|^2 = \| x - d(z) \|^2 = \| x - d(e(x)) \|^2$$

Los autoencoders tienen muchas aplicaciones en el mundo real:

- Compresión y reconstrucción de imágenes.
- Generación de música y separación de fuentes de sonido.
- Compresión de archivos para reducir el tamaño sin perder información importante.
- Extracción de características numéricas de señales de voz que se pueden usar para identificar palabras y frases.

Tipos de autoencoders

Los autoencoders son modelos que tienen muchas variaciones. Las más comunmente utilizadas son:

- Vanilla Autoencoder
- Stacked autoencoders
- Denoising Autoencoder
- Convolutional Autoencoder
- *Variational Autoencoder*[2]

[2] Los veremos más adelante, ya que introducen variaciones importantes en la arquitectura básica de un autoencoder.

Vanilla Autoencoder

Vanilla Autoencoder

Son el tipo más simple de autoencoder.

- Introducidos por Hinton y Salakhutdinov en su artículo "*Reducing the dimensionality of data with neural networks*" [**?**].

Consisten en una sola capa oculta.

- Que se llama "espacio latente" y se denota como z

Son una forma simple y efectiva de aprender representaciones comprimidas de datos.

- Se pueden usar para compresión de datos y reducción de dimensionalidad.
- Para otras aplicaciones, como generación de datos sintéticos, no son las mejores elecciones.

Comprimiendo datos con Vanilla Autoencoder

Limitaciones de los Vanilla Autoencoder

Los Vanilla Autoencoder no son muy eficientes en la generación de datos sintéticos.

Problema principal: El **espacio latente generado no es continuo**.

- Está compuesto por regiones separadas entre sí que agrupan características de ejemplos similares.
- Entre estas regiones no hay un espacio continuo de representaciones intermedias.
  - En realidad, lo hay, pero no tiene sentido.
  - ¿Realmente no hay un espacio intermedio entre un 1 y un 7? ¿o entre un 3 y un 8?
- Esto hace que la interpolación entre ejemplos sea imposible.

En resumen, **si el espacio intermedio no es continuo, las salidas del decodificador no son realistas**. –> Esto lo solucionan los autoencoders variacionales

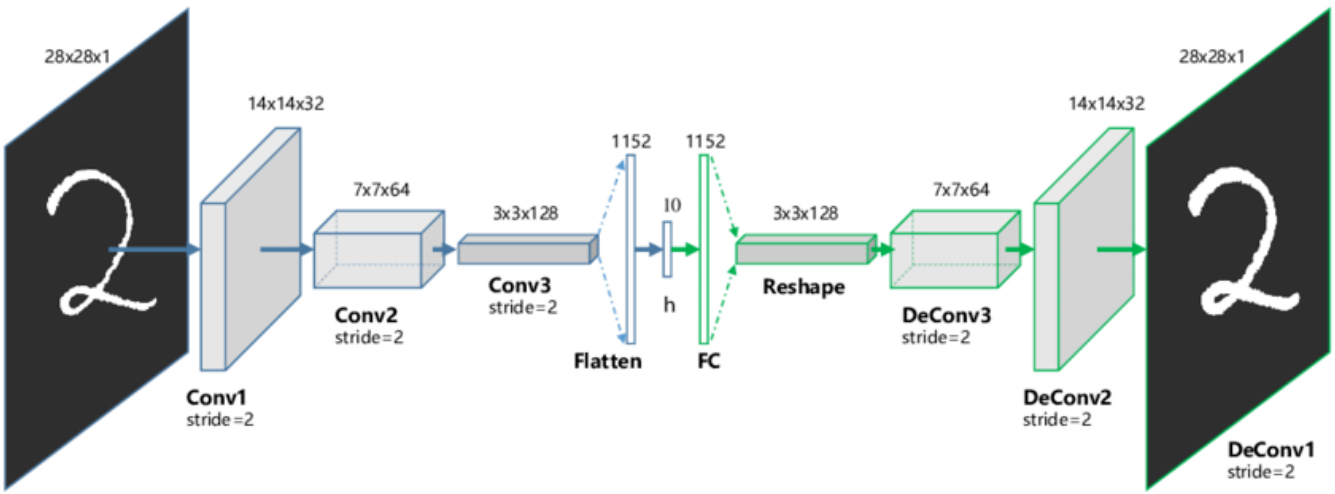
Autoencoders Convolucionales

Autoencoders Convolucionales

Un tipo de red neuronal diseñado para procesar y reconstruir datos de imágenes.

- Utilizan Convolutional Neural Networks (CNNs) debido a su capacidad para capturar relaciones espaciales entre píxeles.
- El codificador típicamente consta de varias capas convolucionales, seguidas de un conjunto de capas totalmente conectadas.
  - Las capas convolucionales extraen características de la imagen de entrada.
  - Las capas totalmente conectadas luego combinan estas características en una representación de menor dimensión de la imagen.
- El decodificador es esencialmente un espejo del codificador, con un conjunto de capas totalmente conectadas seguidas de varias capas deconvolucionales.

Autoencoders Convolucionales



Esquema de un Autoencoder convolucional (fuente).

Pueden aprender a representar datos de imágenes de manera más compacta y eficiente

- Reduciendo así la cantidad de memoria y potencia de procesamiento necesarias para tareas posteriores como clasificación de imágenes o detección de objetos.

Autoencoders convolucionales

Autoencoders de Eliminación de Ruido (Denoising Autoencoders)

Autoencoders de Eliminación de Ruido (Denoising Autoencoders)

Son similares a un autoencoder vanilla [**?**]:

- La red típicamente consta de un codificador y un decodificador.
- La diferencia: una capa de ruido que agrega ruido a los datos de entrada.
- Agrega ruido a los datos de entrada antes de que se alimenten al codificador.

Autoencoders de Eliminación de Ruido (Denoising Autoencoders)

Durante el entrenamiento, se presentan pares de datos de entrada con ruido y datos de salida limpios.

- La red se entrena para minimizar la diferencia entre los datos de entrada originales y los datos de salida reconstruidos, teniendo en cuenta el ruido añadido.

La capa de ruido se puede personalizar según el tipo de ruido:

- e.g. Ruido gaussiano: Capa que agrega ruido gaussiano a los datos de entrada.
- e.g. Ruido sal y pimienta: Capa que establece algunos píxeles en cero.

Pueden aprender representaciones robustas de datos menos sensibles al ruido.

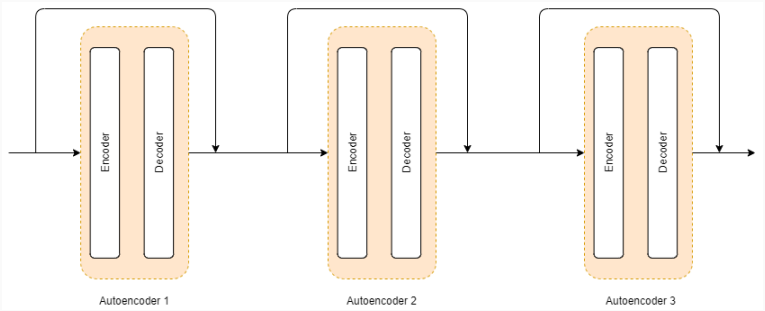
Supresión de ruido en imágenes con autoencoders de eliminación de ruido

Autoencoders Apilados (Stacked autoencoders)

Stacked Autoencoders (SAEs)

Tipo de red neuronal compuesta por múltiples capas de autoencoders [**?**].

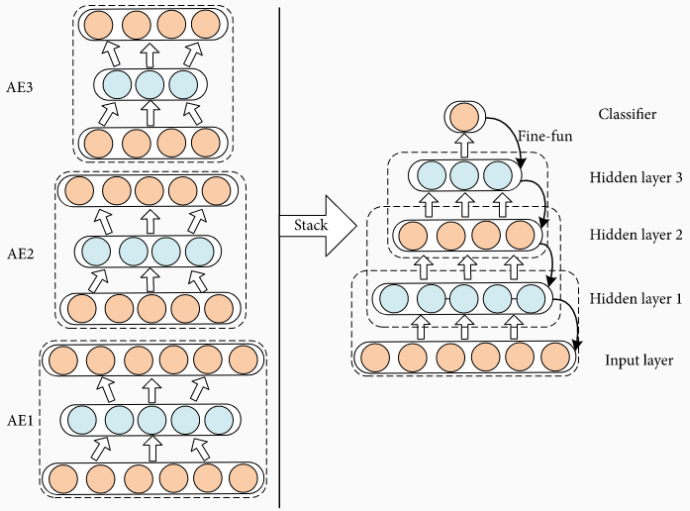
- Cada capa consta de un codificador y un decodificador, similar a un autoencoder vanilla.
- La salida de una capa se alimenta como entrada a la siguiente, creando una red neuronal profunda.



Autoencoder apilado

Permiten una representación más poderosa de los datos que se puede utilizar en tareas posteriores.

SAE como método de entrenamiento de redes neuronales profundas



De hecho, los SAEs pueden usarse como técnica para el entrenamiento de MLPs:

1. Cada capa del autoencoder apilado se entrena por separado para reconstruir sus datos de entrada.
2. Una vez que una capa se entrena, su salida se usa como entrada para la siguiente capa.
3. Cuando toda la red ha sido entrenada, los espacios latentes resultantes se apilan.

Luego, se ajusta finamente la red resultante para minimizar el error de reconstrucción entre los datos de entrada originales y los datos de salida finales.

- Esta fue una de las primeras técnicas para entrenar redes

Autoencoders apilados y la reconstrucción de fashion MNIST

Recursos

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

Referencias i