APRENDIZAJE PROFUNDO

S06: Redes generativas (Autoencoders)

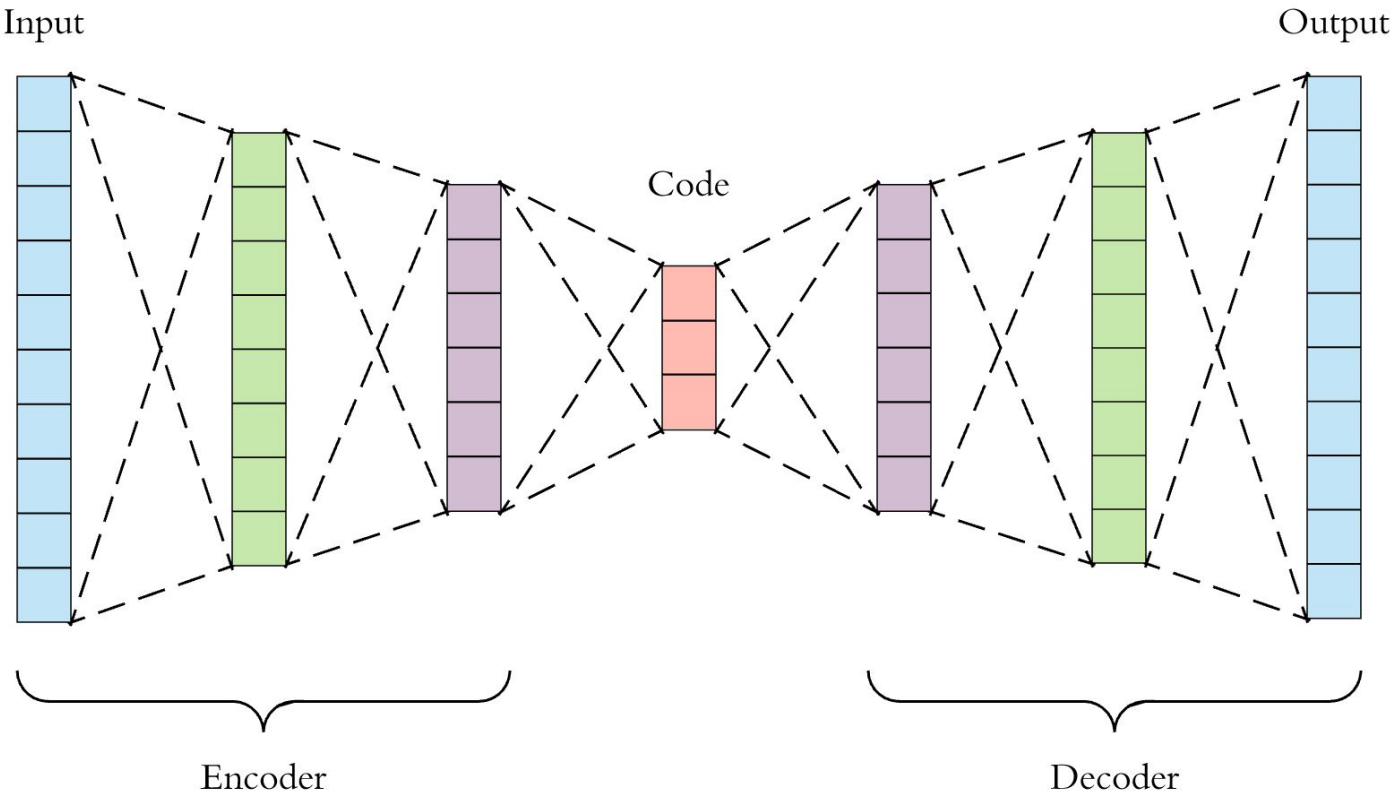
Magíster en Inteligencia Artificial

Expositor: Dr. Juan Bekios Calfa

Contenidos

1. ¿Qué es un autoencoder?
2. Aplicaciones y usos de los autoencoder
3. Funcionamiento de un autoencoder
4. Implementación de un autoencoder
    a. Autoencoder simple
    b. Autoencoder convolucional

¿Qué es un autoencoder?



¿Qué es un autoencoder?

Los **autoencoders** son redes neuronales permiten generar nuevos datos primero comprimiendo la entrada en un espacio de **variables latentes** y luego reconstruyendo la salida con base en la información adquirida. Este tipo de red consta de dos partes:

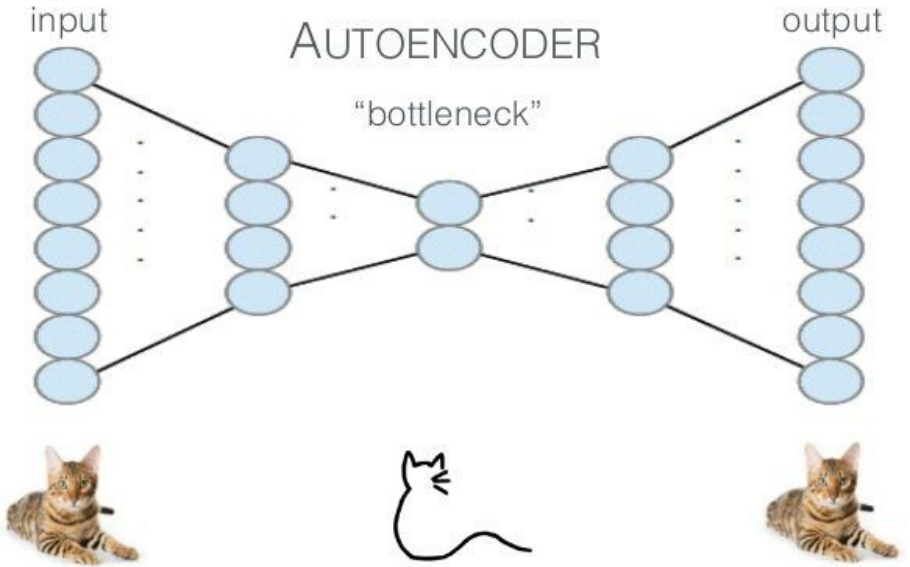
1. **Encoder:** la parte de la red que comprime la entrada en un espacio de variables latentes y que puede representarse mediante la función de codificación h = f (x).
2. **Decoder:** la parte que trata de reconstruir la entrada basándose en la información recolectada previamente. Se representa mediante la función de decodificación r = g (h).

¿Qué es un autoencoder?

Espacios Latentes

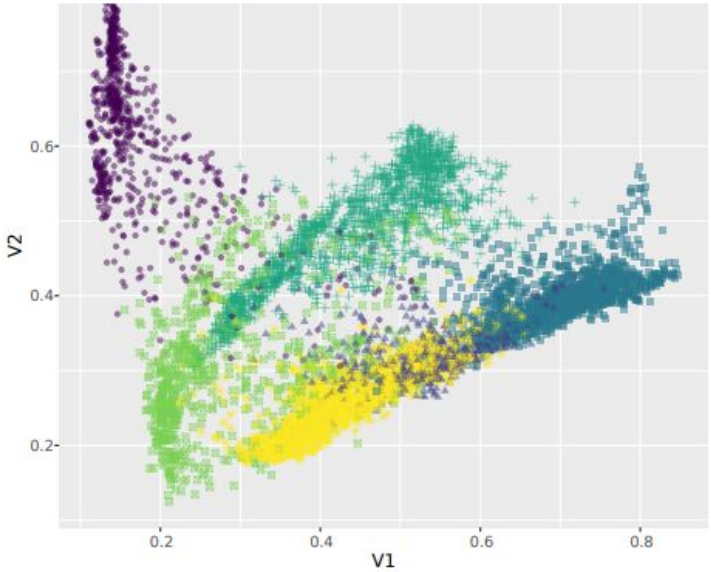
Una vez obtenido un **autoencoder**, podríamos intentar usar el espacio de representaciones intermedias (también llamado **espacio latente**) como un mecanismo independiente para obtener con el decoder salidas similares a los datos de entrada.

Para ello, tomaríamos al azar un punto del espacio latente, le aplicaríamos el decoder, y como resultado obtendríamos una salida que debe reflejar un objeto que se mueve en el mismo espacio que las entradas.



Aplicaciones y usos de los autoencoder

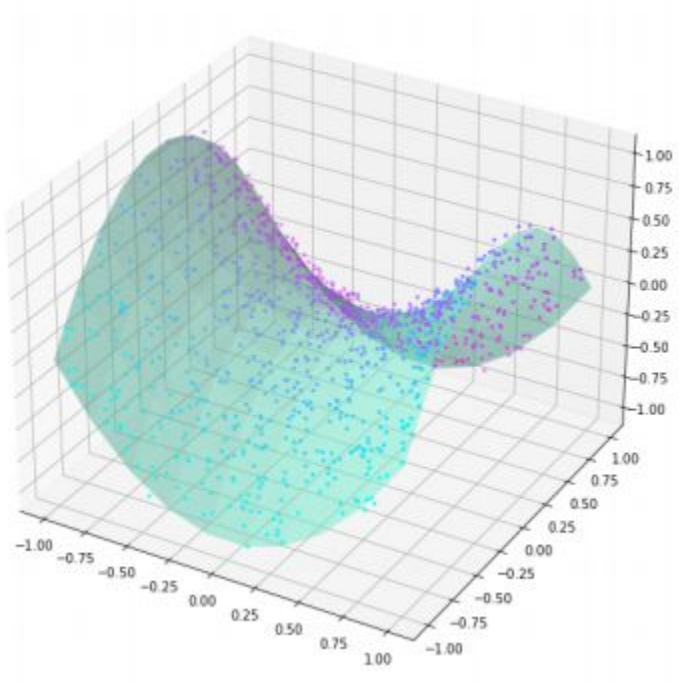
Reducción de dimensiones

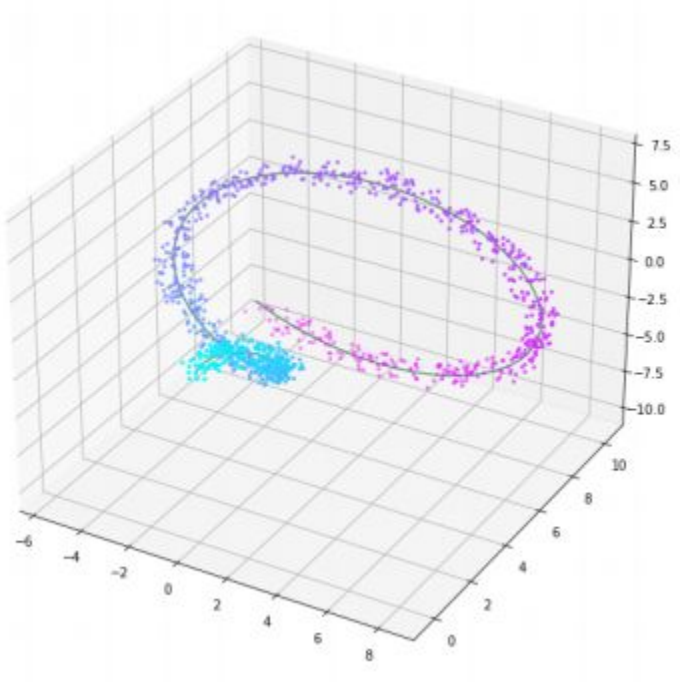


satellite_image - Datos espectrales que identifican el tipo de suelo. Espacio original con 36 dimensiones - https://www.openml.org/d/294

Aplicaciones y usos de los autoencoder

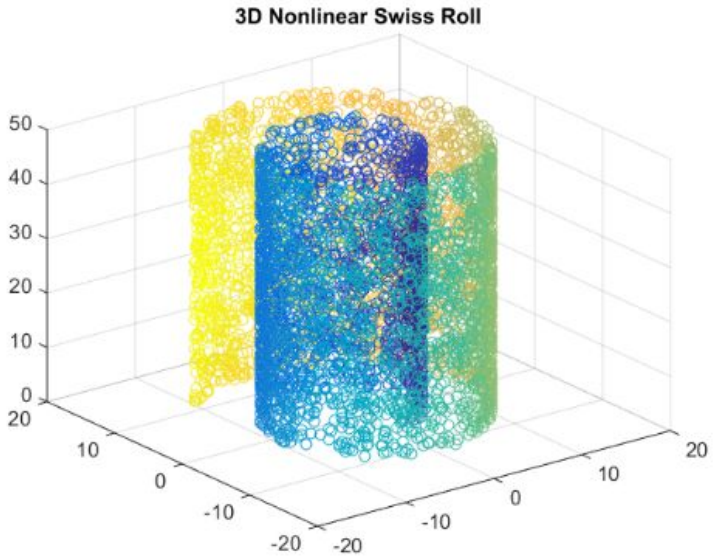
Manifold Learning

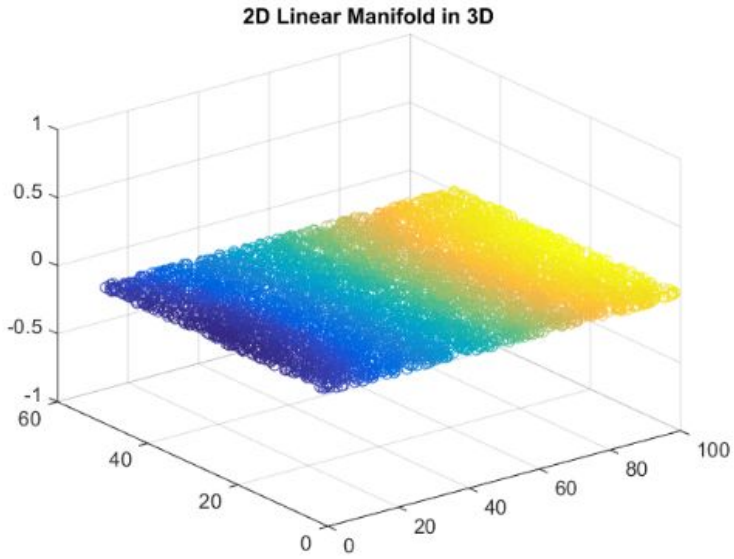




Aplicaciones y usos de los autoencoder

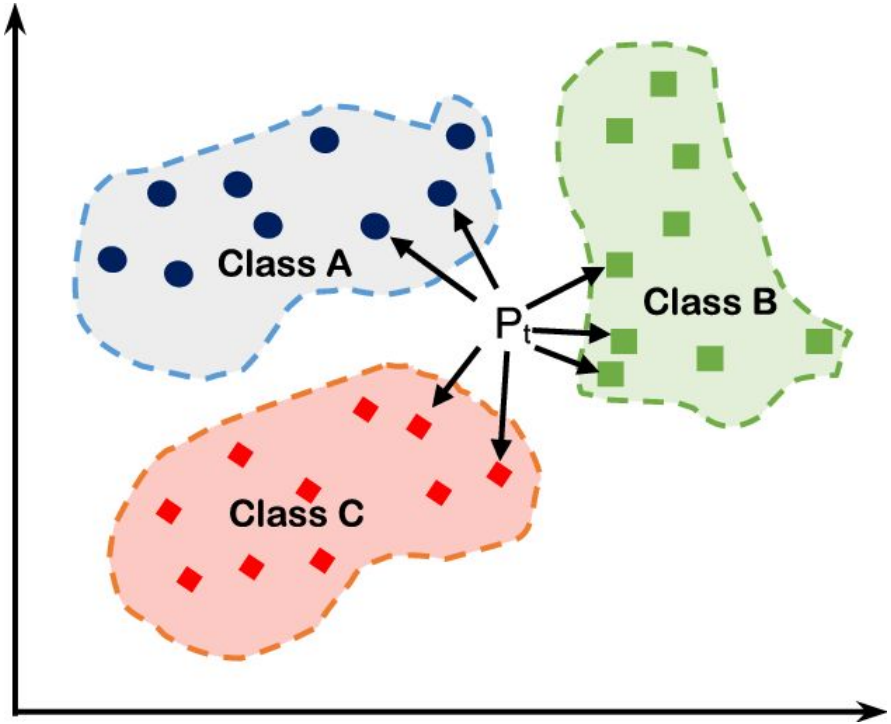
Manifold Learning





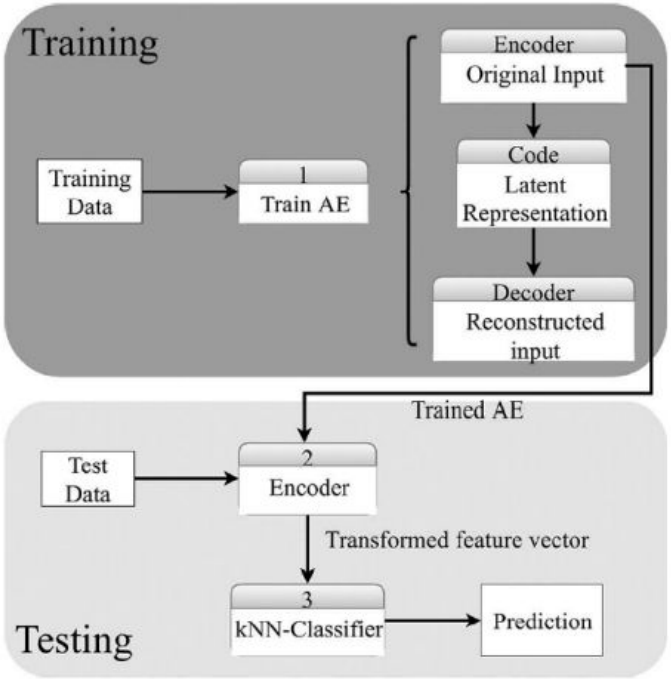
Aplicaciones y usos de los autoencoder

KNN mejorado (I)



Aplicaciones y usos de los autoencoder

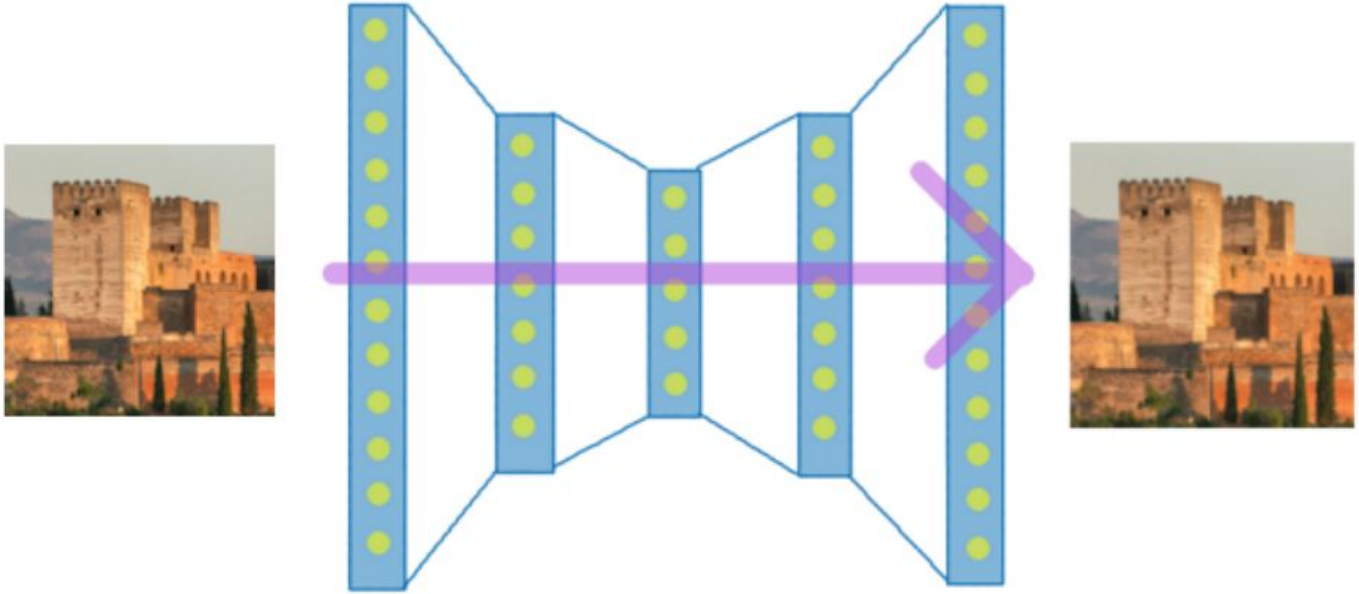
KNN mejorado (II)



Pulgar, F. J., Charte, F., Rivera, A. J., & del Jesus, M. J. (2019). AEkNN: An AutoEncoder kNN-Based Classifier With Built-in Dimensionality Reduction. International Journal of Computational Intelligence Systems, 12(1), 436-452.

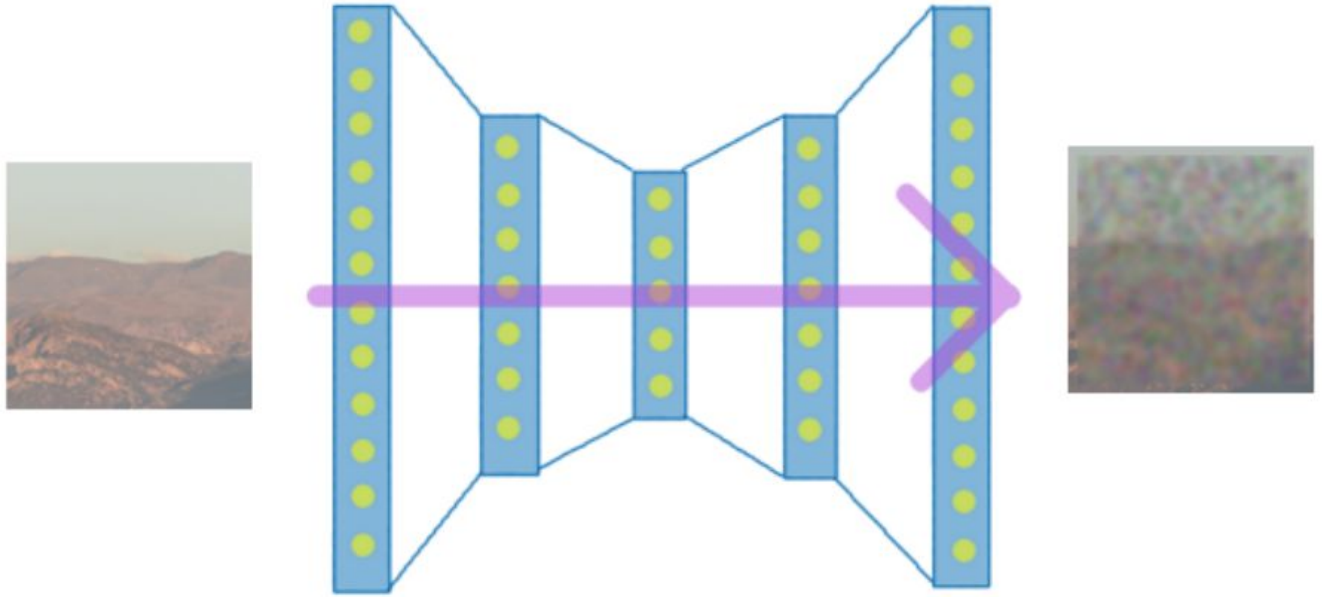
Aplicaciones y usos de los autoencoder

Detección de datos anómalos



Aplicaciones y usos de los autoencoder

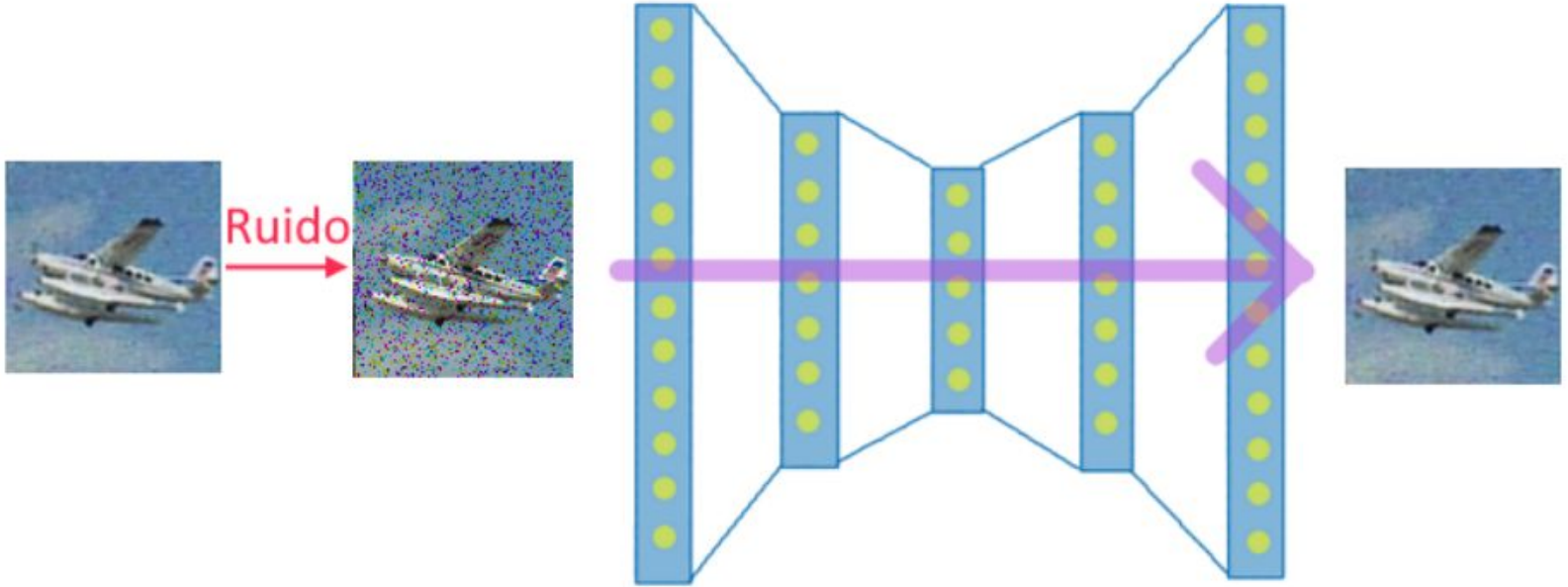
Detección de datos anómalos



DaSCI
Instituto Andaluz de Investigación en Data Science and Computational Intelligence

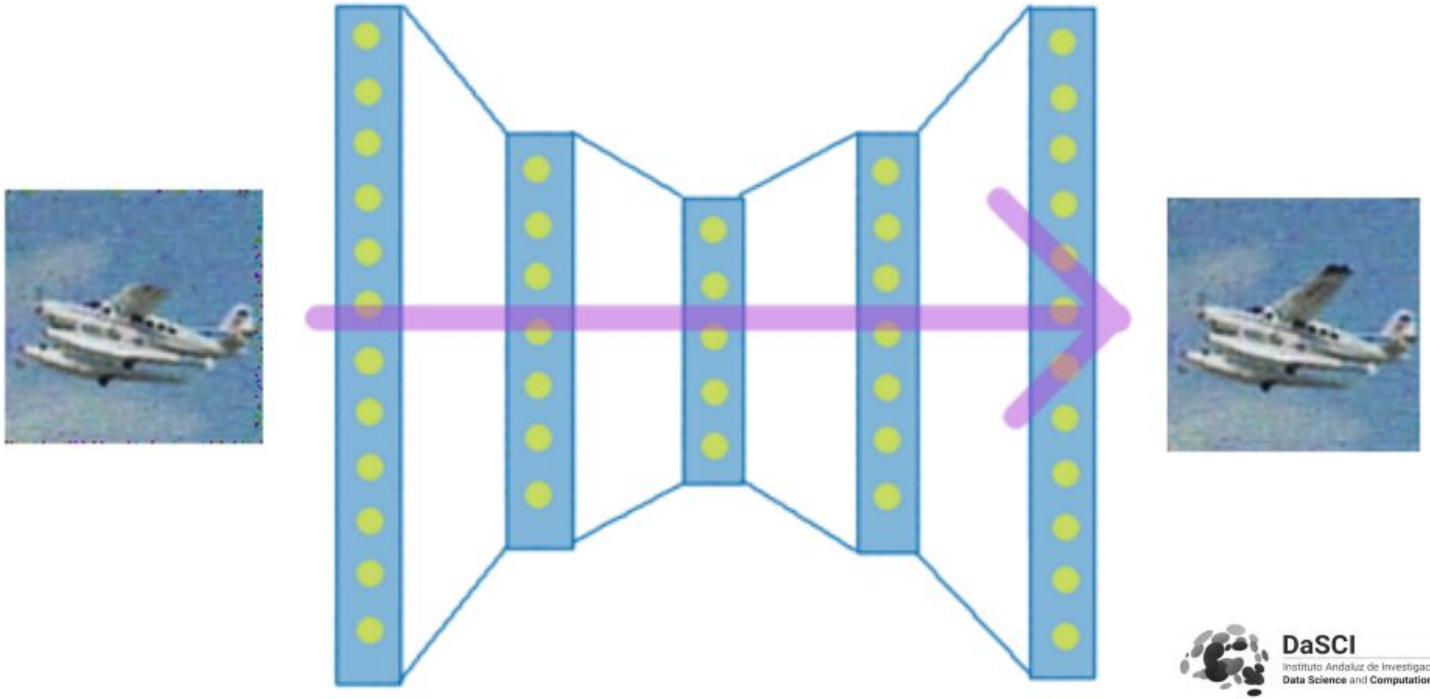
Aplicaciones y usos de los autoencoder

Eliminar ruido de los datos



Aplicaciones y usos de los autoencoder

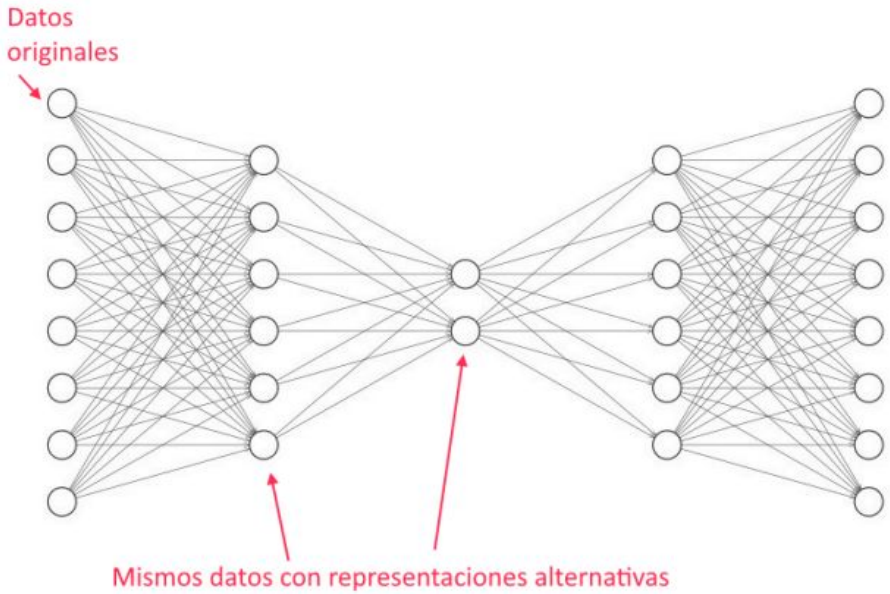
Imputar datos



DaSCI
Instituto Andaluz de Investigación en
Data Science and Computational Intelligence

Aplicaciones y usos de los autoencoder

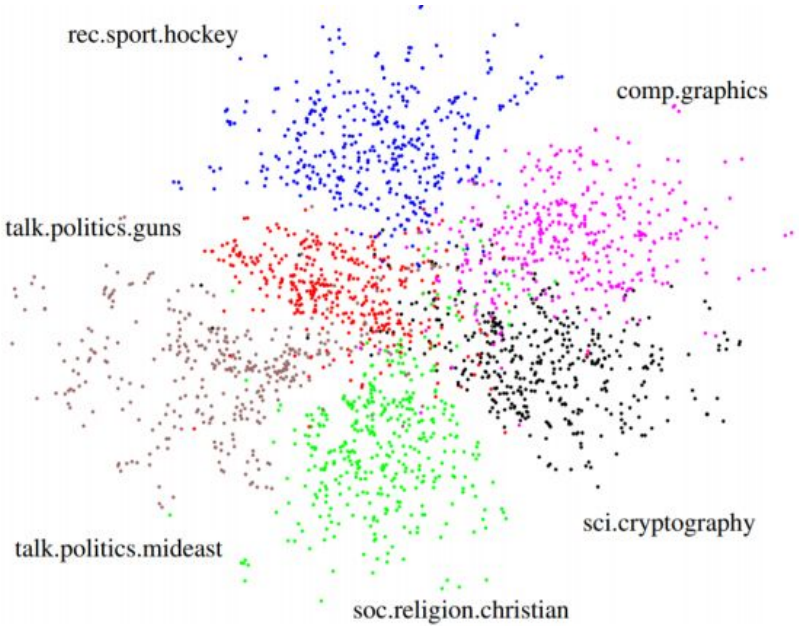
Encriptación y seguridad



La reconstrucción de los datos originales no es posible si no se cuenta con el descodificador del autoencoder

Aplicaciones y usos de los autoencoder

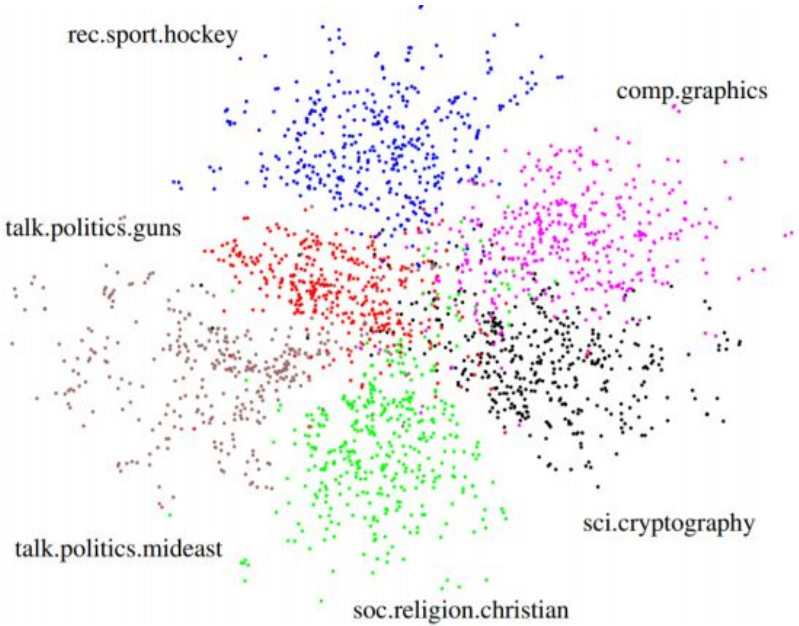
Clustering



Salakhutdinov, R., & Hinton, G. (2009). Semantic hashing. International Journal of Approximate Reasoning, 50(7), 969-978.

Aplicaciones y usos de los autoencoder

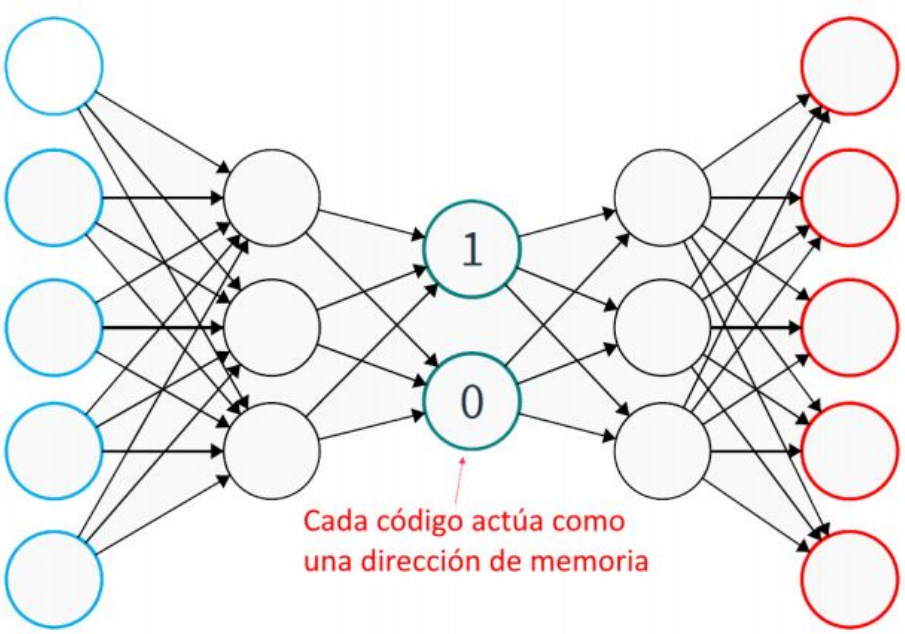
Clustering



Salakhutdinov, R., & Hinton, G. (2009). Semantic hashing. International Journal of Approximate Reasoning, 50(7), 969-978.

Aplicaciones y usos de los autoencoder

Agrupar semánticamente con un autoencoder

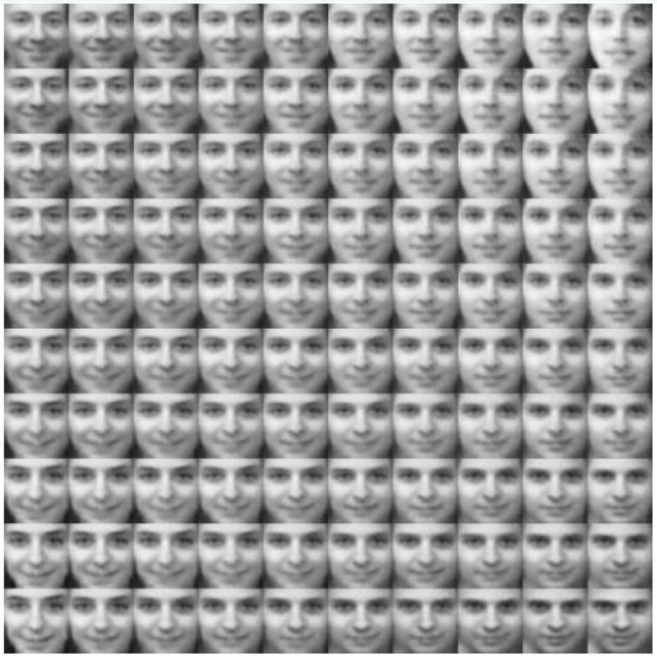


Se fuerza una codificación binaria

- patrones similares generan el mismo código
- se asocian a un mismo grupo con semántica similar

Aplicaciones y usos de los autoencoder

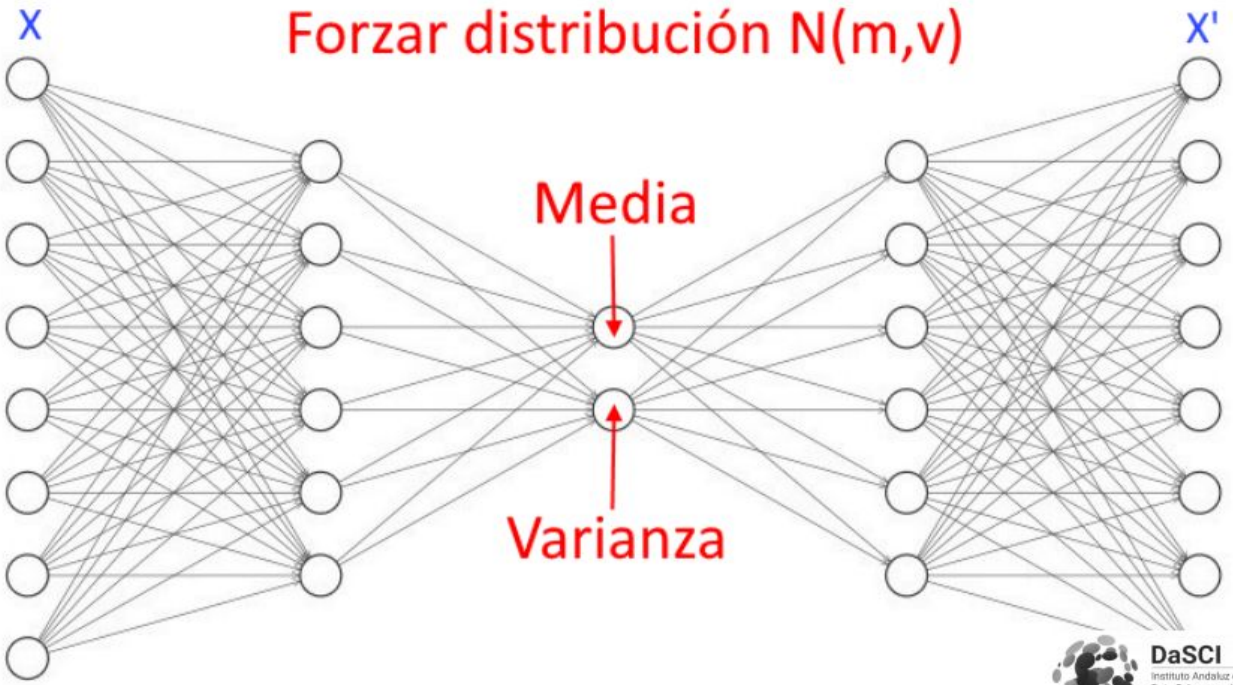
Agrupar semánticamente con un autoencoder



thispersondoesnotexist.com

Aplicaciones y usos de los autoencoder

Generar datos sintéticos con un autoencoder



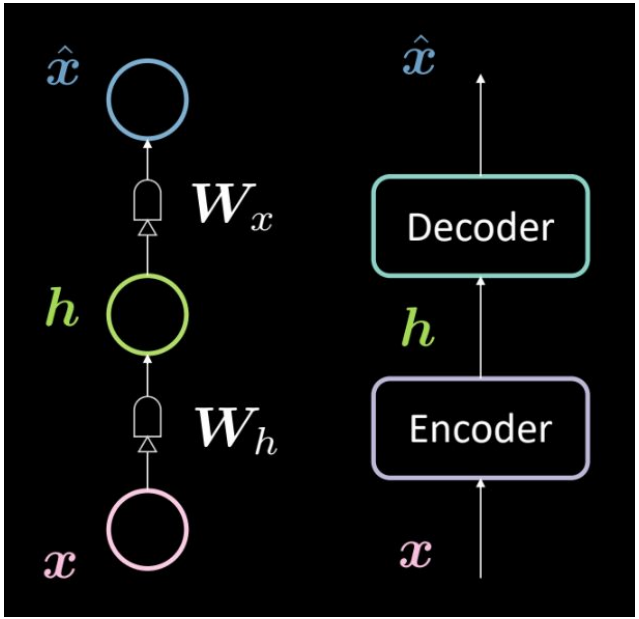
Funcionamiento de un autoencoder

Podemos representar la red anterior matemáticamente usando las siguientes ecuaciones:

$$\boldsymbol{h} = f(\boldsymbol{W}_h \boldsymbol{x} + \boldsymbol{b}_h)$$
$$\hat{\boldsymbol{x}} = g(\boldsymbol{W}_x \boldsymbol{h} + \boldsymbol{b}_x)$$

También especificamos las siguientes dimensionalidades:

$$\boldsymbol{x}, \hat{\boldsymbol{x}} \in \mathbb{R}^n$$
$$\boldsymbol{h} \in \mathbb{R}^d$$
$$\boldsymbol{W}_h \in \mathbb{R}^{d \times n}$$
$$\boldsymbol{W}_x \in \mathbb{R}^{n \times d}$$

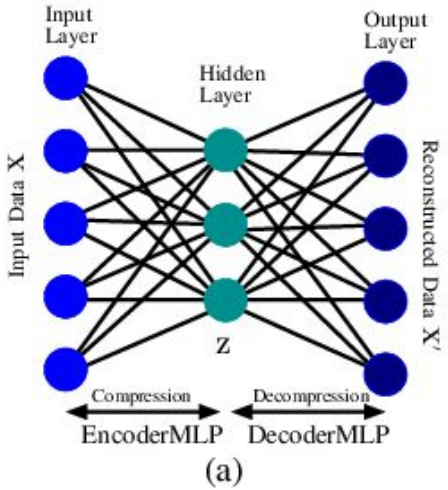


Funcionamiento de un autoencoder

1. *Vanilla autoencoder*
2. *Autoencoder multicapa*
3. *Autoencoder convolucional*
4. *Autoencoder regularizado*

Funcionamiento de un autoencoder

Vanilla autoencoder



Funcionamiento de un autoencoder

Vanilla autoencoder

Función de costo de reconstrucción

Veamos ahora las funciones de costo de reconstrucción que usamos generalmente. La función de costo general para el conjunto de datos se da como el promedio por costo de muestra, es decir,

$$L = \frac{1}{m}\sum_{j=1}^{m} \ell(x^{(j)}, \hat{x}^{(j)})$$

Cuando la entrada es categórica, podríamos usar la función de costo de entropía cruzada (cross-entropy) para calcular el costo por muestra que viene dada por

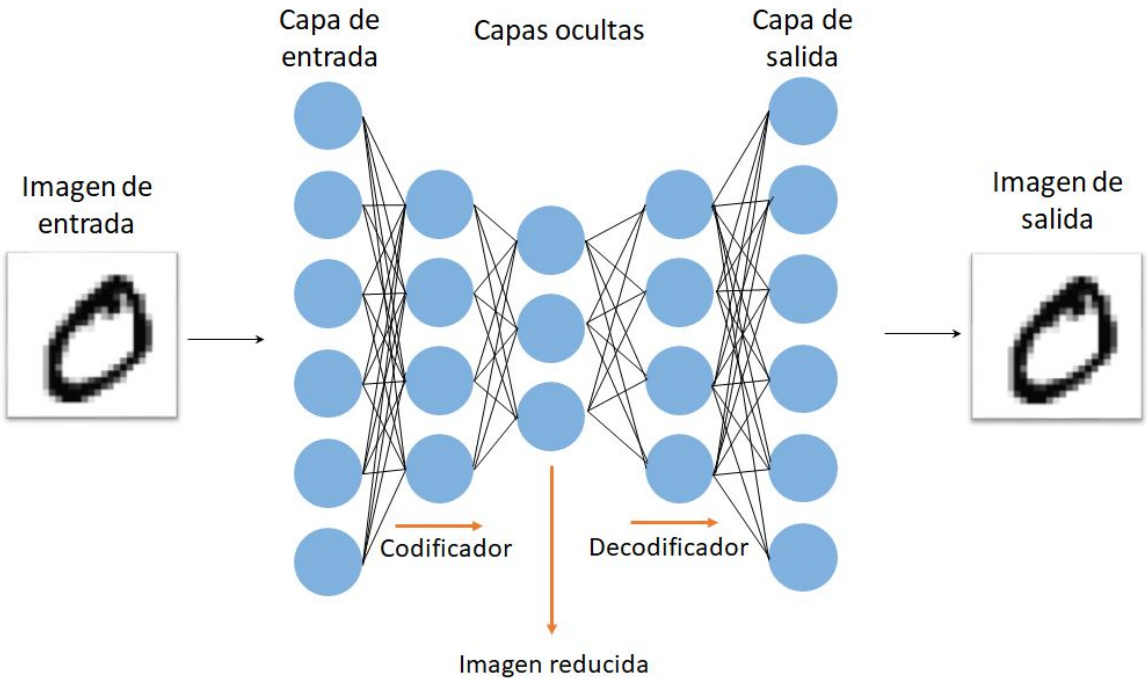
$$\ell(\boldsymbol{x}, \hat{\boldsymbol{x}}) = -\sum_{i=1}^{n} [x_i \log(\hat{x}_i) + (1 - x_i) \log(1 - \hat{x}_i)]$$

Y cuando la entrada tiene un valor real, es posible que deseemos usar la función de costo de error cuadrático medio dada por

$$\ell(\boldsymbol{x}, \hat{\boldsymbol{x}}) = \frac{1}{2}\|\boldsymbol{x} - \hat{\boldsymbol{x}}\|^2$$

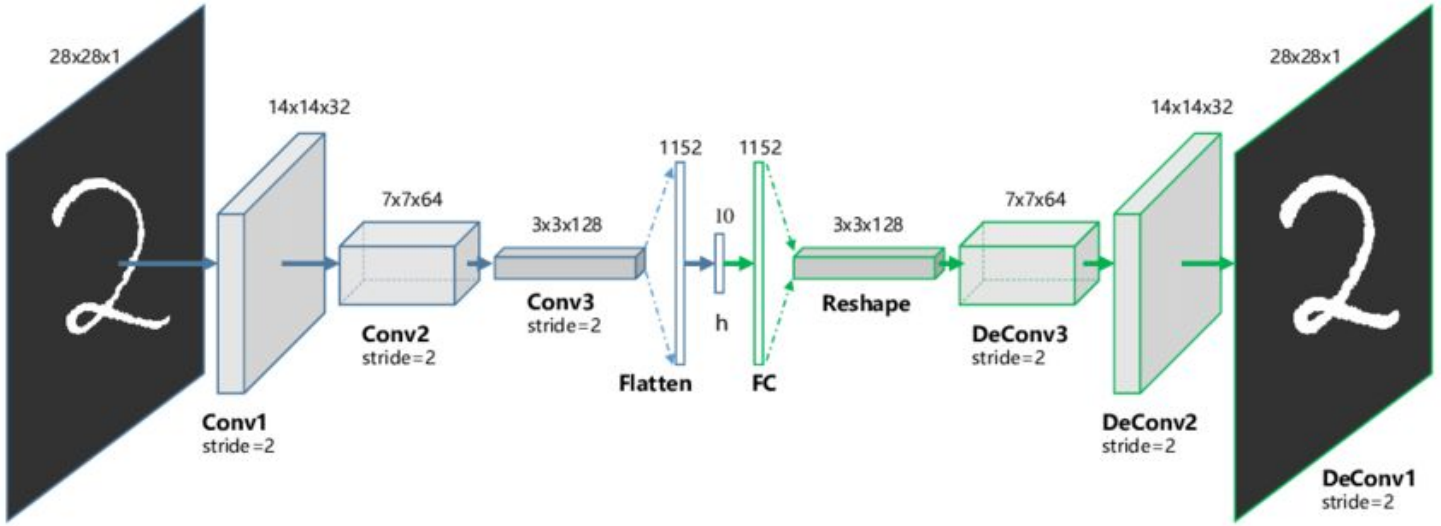
Funcionamiento de un autoencoder

Autoencoder multicapa



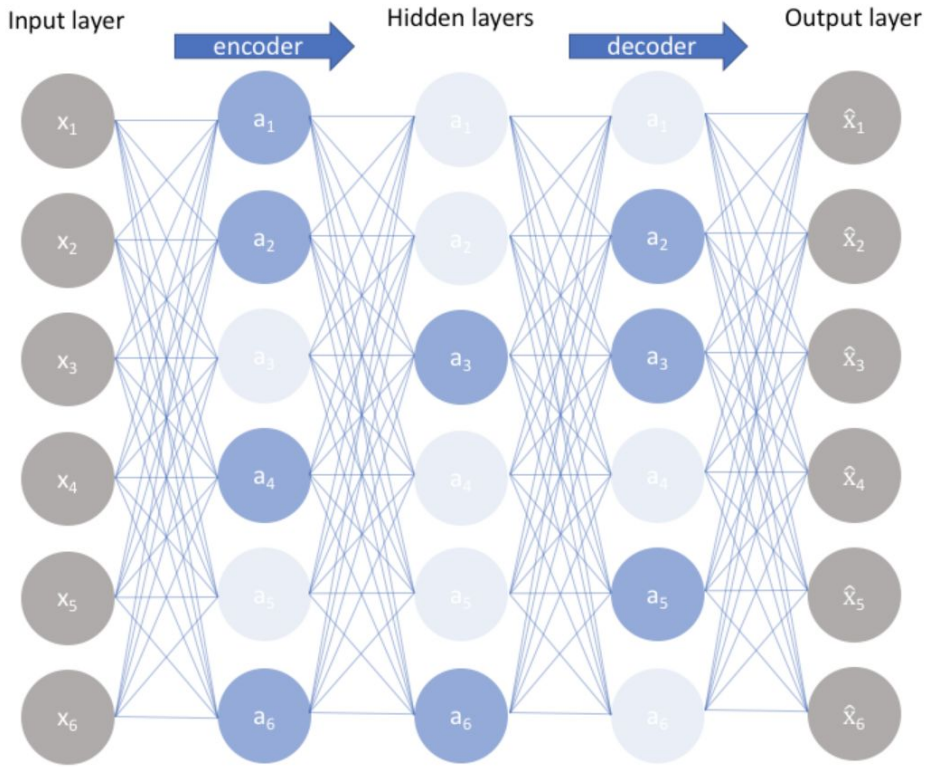
Funcionamiento de un autoencoder

Autoencoder convolucional



Funcionamiento de un autoencoder

Autoencoder regularizado



Funcionamiento de un autoencoder

Autoencoder regularizado

- **L1 Regularization**: We can add a term to our loss function that penalizes the absolute value of the vector of activations a in layer h for observation i, scaled by a tuning parameter λ.

$$\mathcal{L}\left(x, \hat{x}\right) + \lambda \sum_{i} \left| a_i^{(h)} \right|$$

Funcionamiento de un autoencoder

Autoencoder regularizado

- **KL-Divergence**: In essence, KL-divergence is a measure of the difference between two probability distributions. We can define a sparsity parameter ρ which denotes the average activation of a neuron over a collection of samples. This expectation can be calculated as $\hat{\rho}_j = \frac{1}{m} \sum_i \left[a_i^{(h)} (x) \right]$ where the subscript j denotes the specific neuron in layer h, summing the activations for m training observations denoted individually as x. In essence, by constraining the average activation of a neuron over a collection of samples we're encouraging neurons to only fire for a subset of the observations. We can describe ρ as a Bernoulli random variable distribution such that we can leverage the KL divergence (expanded below) to compare the ideal distribution ρ to the observed distributions over all hidden layer nodes $\hat{\rho}$.

$$\mathcal{L}(x, \hat{x}) + \sum_j KL\left(\rho || \hat{\rho}_j\right)$$

Referencias

1. http://cemixugrmadoc.ugr.es/pages/10-banners/siadef/sesion14_transparencias/!
2. https://atcold.github.io/pytorch-Deep-Learning/es/week07/07-3/
3. https://www.deeplearningitalia.com/introduzione-agli-autoencoder-2/
4. http://www.cs.us.es/~fsancho/?e=232
5. https://www.jeremyjordan.me/autoencoders/
6. http://ufldl.stanford.edu/tutorial/unsupervised/Autoencoders/
7. https://github.com/L1aoXingyu/pytorch-beginner/tree/master/08-AutoEncoder