Carrera de Especialización en Inteligencia Artificial - 3ra cohorte 2021

APRENDIZAJE PROFUNDO

CLASE 7

AUTOENCODER

REPRESENTATION LEARNING / EMBEDDINGS

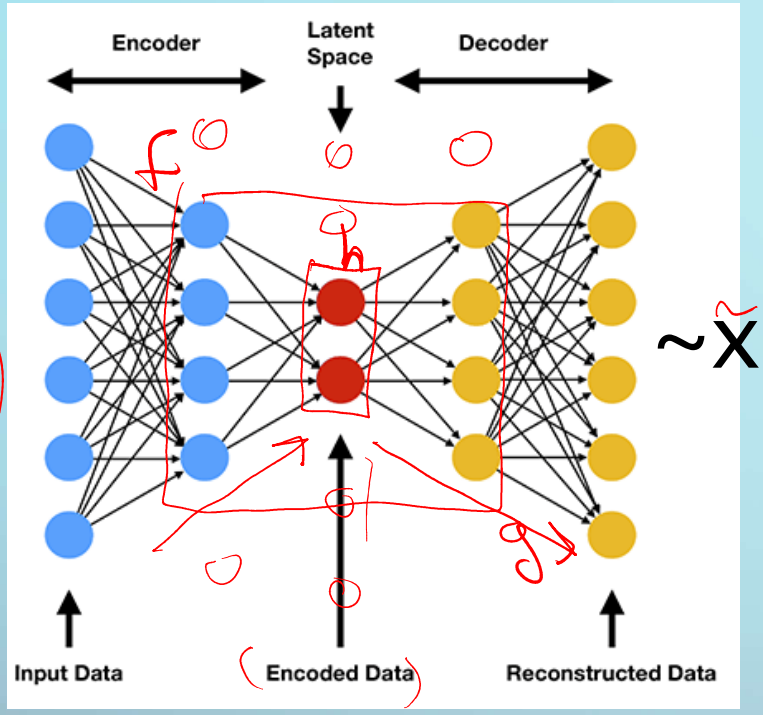
GERATIVE ADVERSARIAL NETWORKS

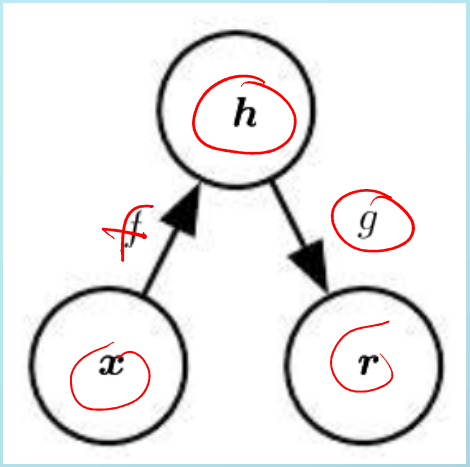
TRANSFER LEARNING

Docente: Dr. Ing. Marcos Uriel Maillot

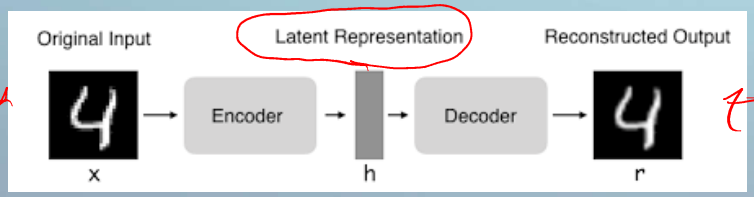
Noviembre 2021

Autoencoders





$$L(\boldsymbol{x}, g(f(\boldsymbol{x}))),$$

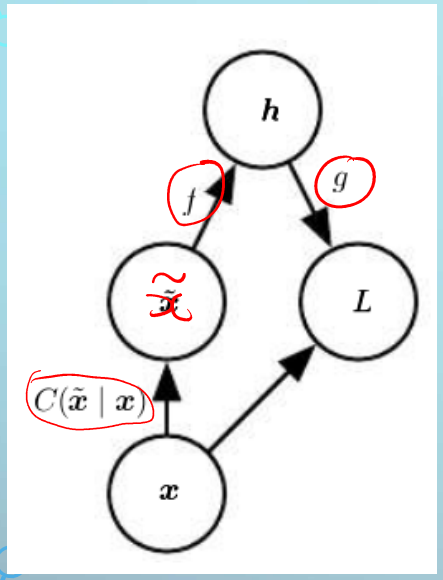


PCA??

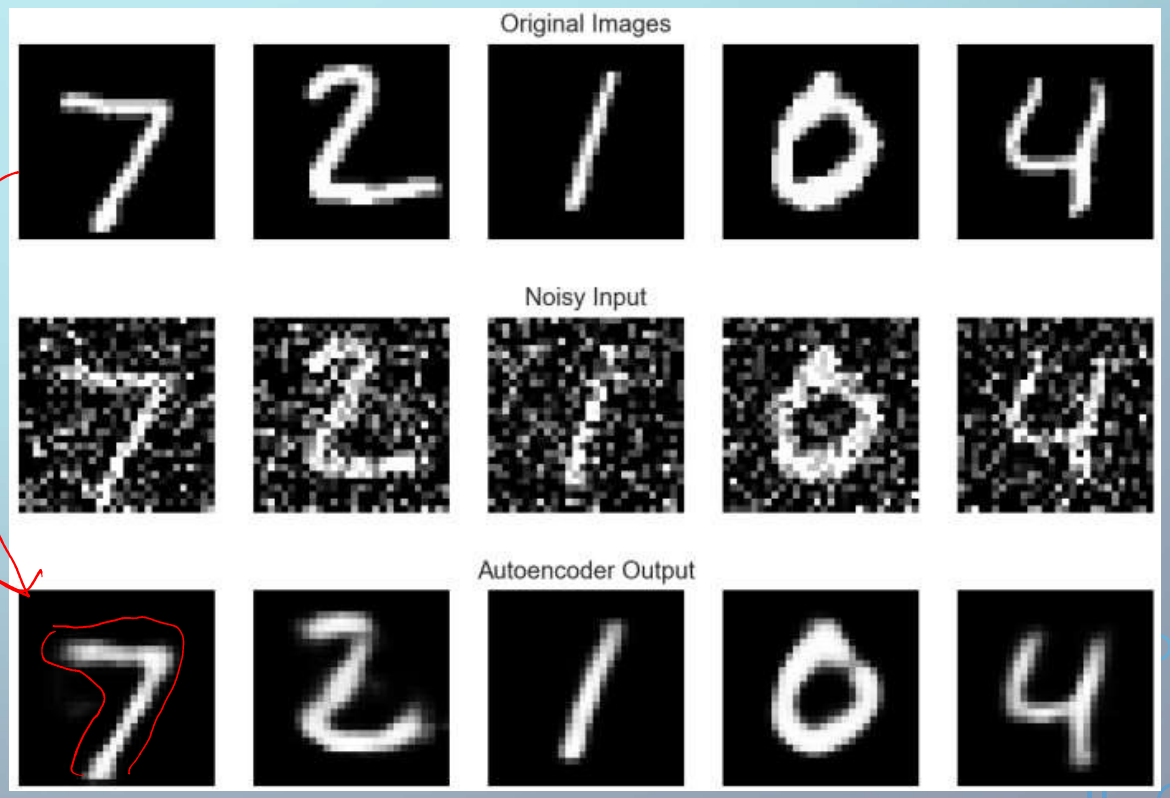
undercomplete autoencoders vs regularized autoencoders

Autoencoders

Denoising autoencoders



$$||g(f(\tilde{x})) - x||^2$$



Autoencoders

undercomplete autoencoders vs **regularized** autoencoders

Si en autoencoder tiene capacidad suficiente copiará al entrada (aprende la función identidad).

Se reduce la capacidad del autoencoder (**undercomplete** …) para que aprenda los "aspectos relevante" de la entrada. Aprenden la *latent variable* del dataset.

Otra manera es "regularizar" sus pesos en el entrenamiento… **regularized** … para que aprenda otras características del dataset.

$$L(\boldsymbol{x}, g(f(\boldsymbol{x}))) + \Omega(\boldsymbol{h}, \boldsymbol{x}),$$

Autoencoders

Sparse autoencodes → limitación de activación neuronas

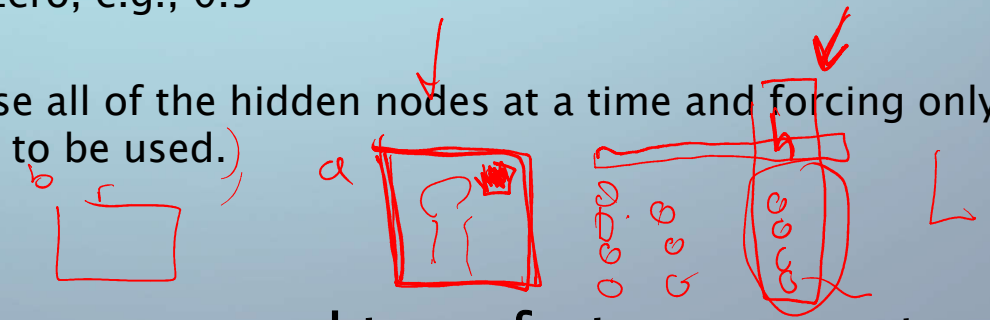
$$L(\boldsymbol{x}, g(f(\boldsymbol{x}))) + \Omega(\boldsymbol{h}, \boldsymbol{x}),$$

$$\Omega(\boldsymbol{h}) = \lambda \sum_i |h_i|,$$

- **Sparse activation** – for a given input, most of the hidden neurons only produce a very small activation.

- For a given hidden node, **it's average activation value (over all the training samples)** should be a small value close to zero, e.g., 0.5

- This prevents autoencoders to use all of the hidden nodes at a time and forcing only a reduced number of hidden nodes to be used.



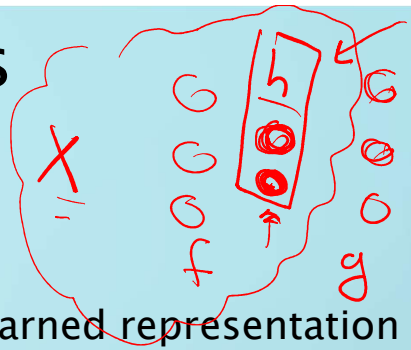
Sparse autoencodes → se usan para **obtener features** para otra tarea.

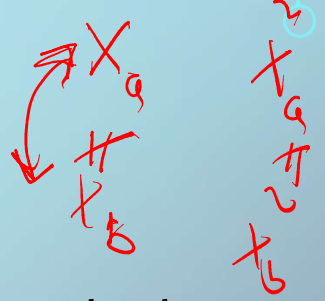
Autoencoders

constractive autoencoders CAE

$$\Omega(\boldsymbol{h}) = \lambda \left\| \frac{\partial f(\boldsymbol{x})}{\partial \boldsymbol{x}} \right\|_F^2$$

The objective is to have a robust learned representation which is **less sensitive to small variation in the data**

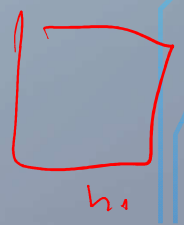




•similar inputs have similar encodings. Hence, we're forcing the model to learn how to **contract** a neighborhood of inputs into a smaller neighborhood of outputs.

•Undercomplete + constractive → se aprenden d f(**x**)/d**x** pequeñas. Solo un número pequeño de **hi** (que se corresponden con un número reducido de direcciones en **x**), pueden tener una derivada considerable.

•Esas direcciones marcarán los planos tangentes en el *manifold hiperplane*

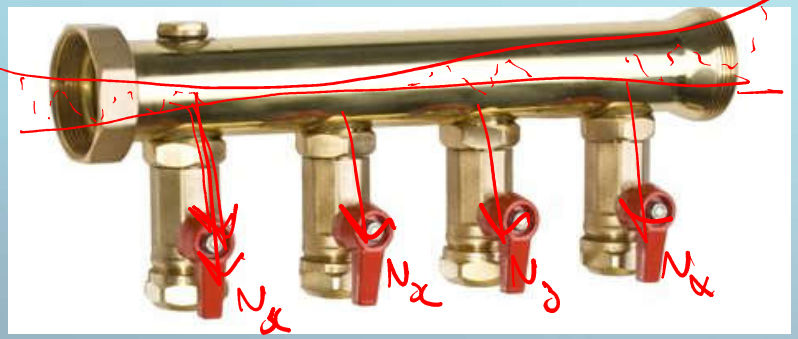


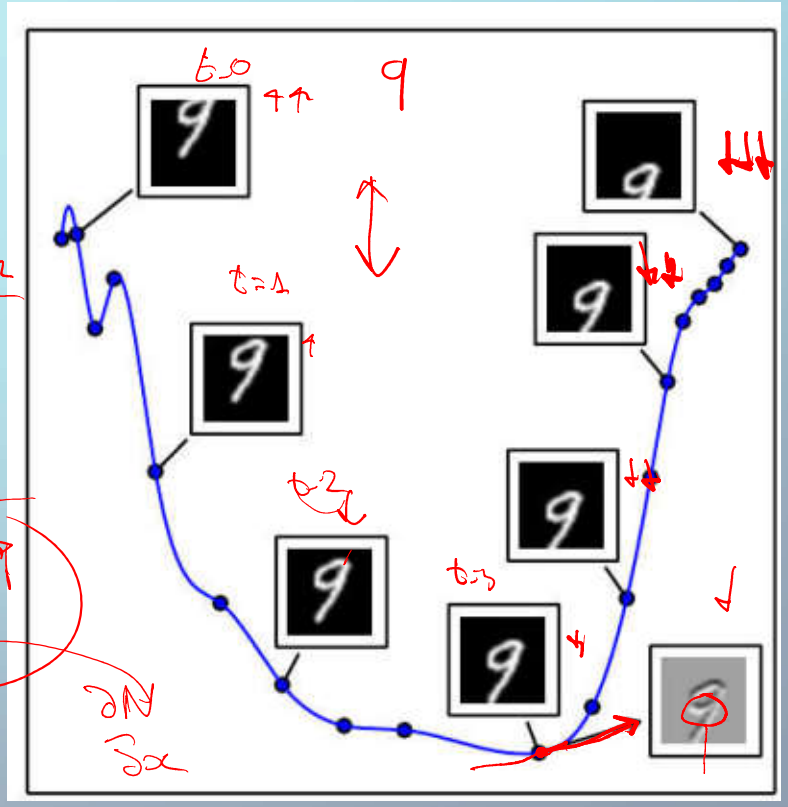
Constractive autoencodes → hace una f(x) que no varia mucho con pequeños cambios en x

Autoencoders

Learning Manifolds with Autoencoders

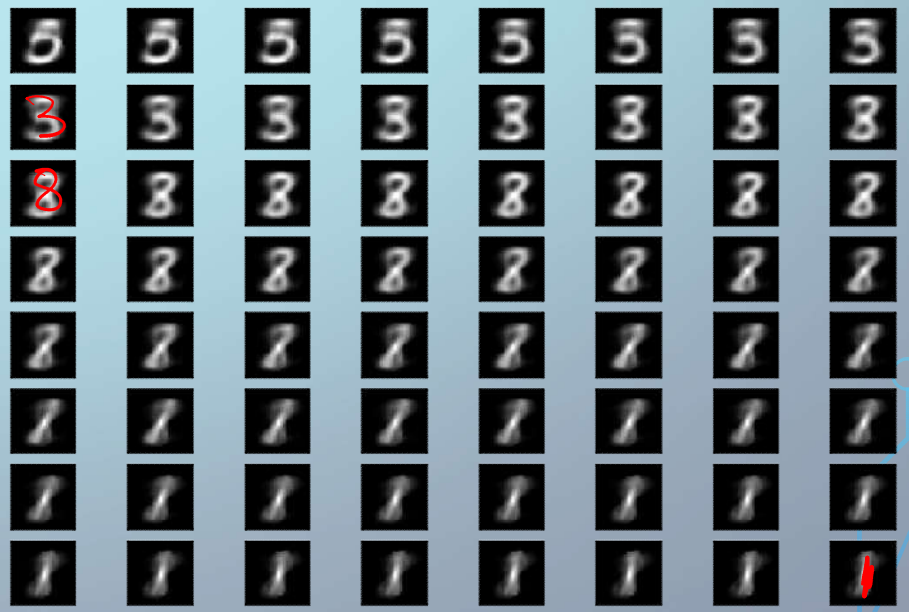
Un “manifold” para Marcos





Autoencoders

Learning Manifolds with Autoencoders



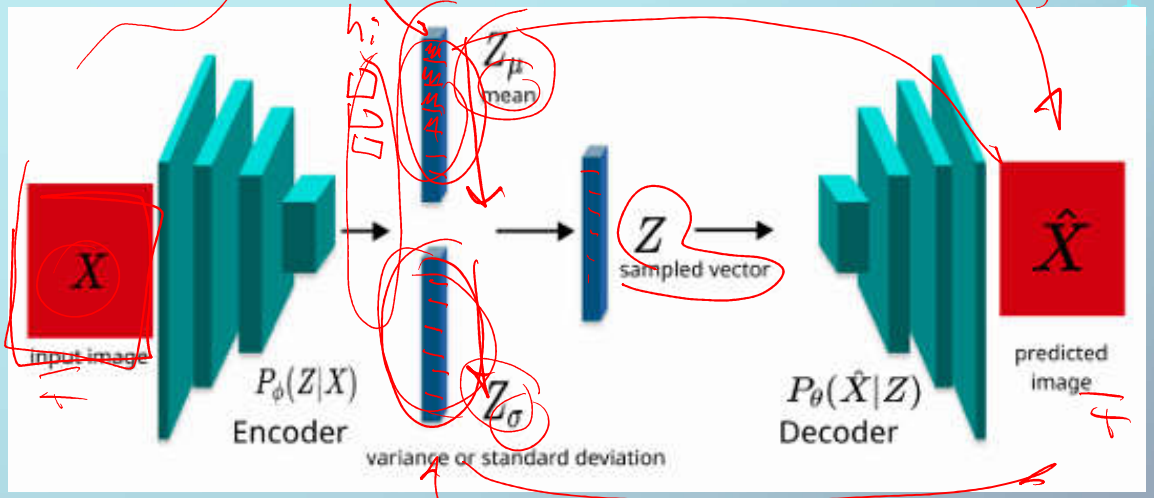
https://www.kaggle.com/apapiu/manifold-learning-and-autoencoders

Autoencoders

varationals autoencoder – VAE

Encoder genera *Gaussian density function* con sigma y mean en lugar de un vector de *latent variable*

$$Z_\mu, Z_{ln(\sigma^2)} = enc(X)$$
$$\epsilon \in N(0,1)$$
$$Z = Z_\mu + \epsilon \sqrt{exp(Z_{ln(\sigma^2)})}$$
$$X_{recon\mu} = dec(Z)$$

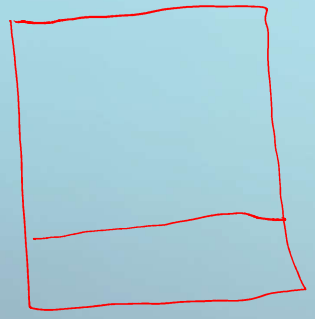


- It gives significant control over how we want to model our latent distribution unlike the other models.
- After training you can just sample from the distribution followed by decoding and generating new data.

In a VAE, one views our encoding vector z as a latent variable with a probability density function P(z) such that if we sample z from P(z) we have a high probability that the decoded vector d(z) is a good example from or very near the manifold for our data X.

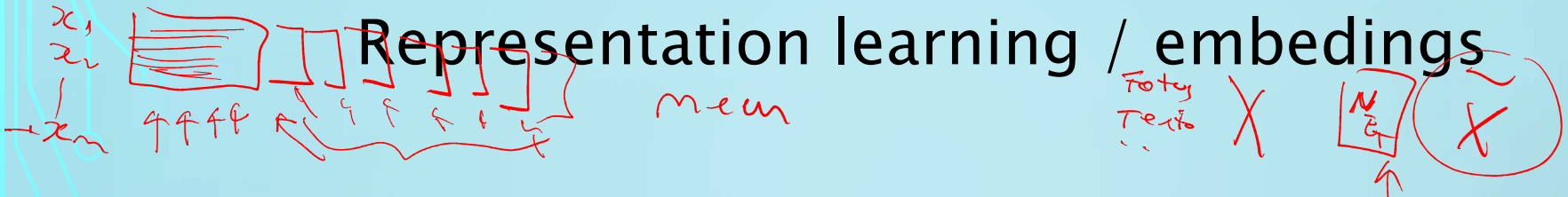
Autoencoders

ver github autoencoder



¡Un merecido descanso!

Representation learning / embedings



Representation learning (featuring engineer automático)

Obtener **features** de unlabeled data siguiendo un entrenamiento supervisado bajo una NN secundaria (autoencoder o semejantes)

+ Reducción de dimension del input
+ Encontrar latent variables

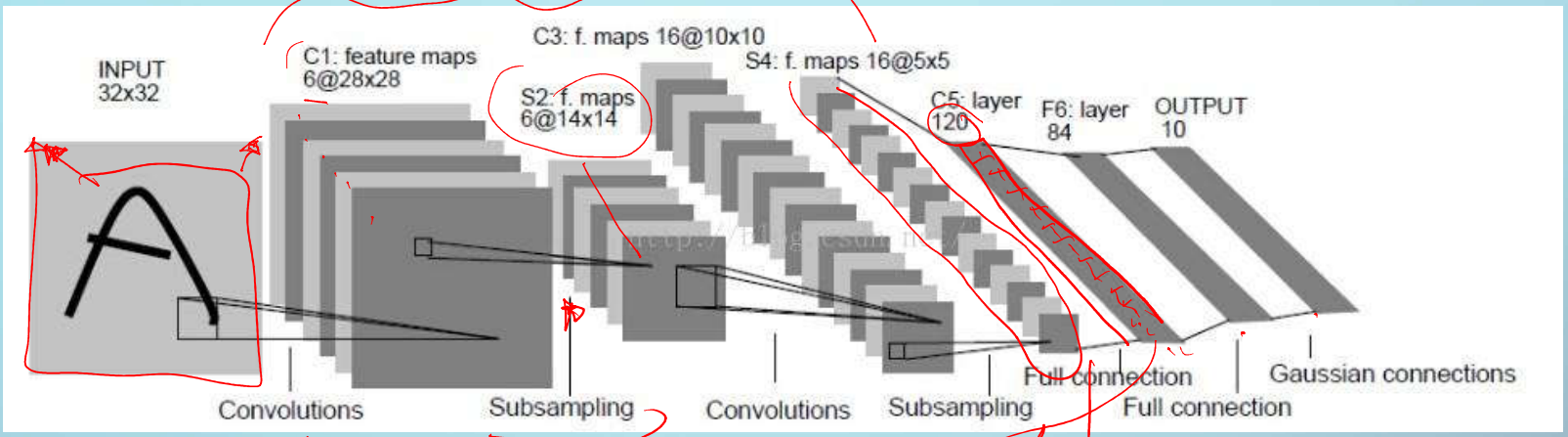
Se reduce la complejidad del dataset → se reducen las anomalias y el ruido

Hacemos operaciones sobre ellos

- Clustering maps
- Reducción de dimensiones
- Operdores lógicos (auto > bicicleta?)
- Medir distancias (manzana mas cerca que torta?)
- Proyecciones o multiplicaciones

Representation learning / embedings

Representation learning



rep learning = embedding

Mas datos (de entrenamiento) no necesariamente garantiza llegar a un buen modelo.

Con features correctos, las tarea de la red puede ser mejor alcanzada.

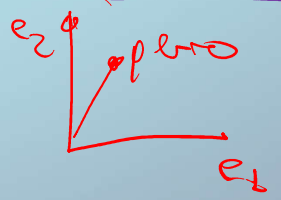
When the learned features are passed into the supervised learning algorithm, it can improve the prediction accuracy up to 17%.
[https://dl.acm.org/doi/10.1145/3303772.3303795].

Representation learning / embeddings

Embeddings → se trata de crear espacio continuo de representación de los inputs.

Se crean ad-hoc o dentro del frame de la NN

Cada palabra es representada por un vector N-dimensional en un sub-espacio contínuo (en el embedding)



Perro → [e1, e2, e3, … eN]

La posición que toma cada palabra (input del embedding) se aprende a partir de su entorno.

Representation learning / embeddings

Embeddings

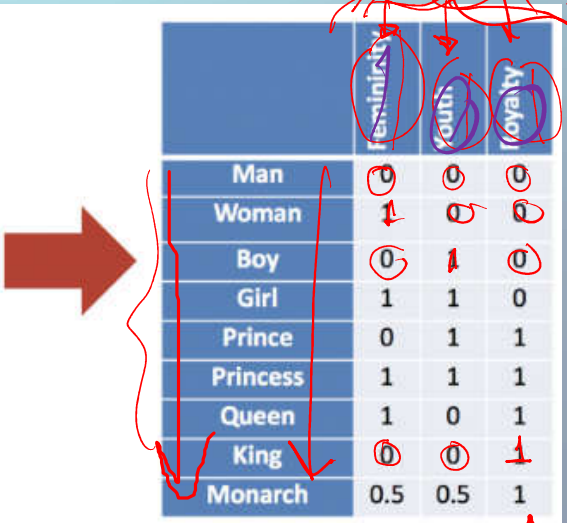
Vocabulary:
Man, woman, boy, girl, prince, princess, queen, king, monarch

	1	2	3	4	5	6	7	8	9
man	1	0	0	0	0	0	0	0	0
woman	0	1	0	0	0	0	0	0	0
boy	0	0	1	0	0	0	0	0	0
girl	0	0	0	1	0	0	0	0	0
prince	0	0	0	0	1	0	0	0	0
princess	0	0	0	0	0	1	0	0	0
queen	0	0	0	0	0	0	1	0	0
king	0	0	0	0	0	0	0	1	0
monarch	0	0	0	0	0	0	0	0	1

One-hot-encoding
!=
Word embedding

- Reducción de dimensiones
- Se aprenden propiedades intrínsecas de palabas que pertenecen al mismo grupo.

Vocabulary:
Man, woman, boy, girl, prince, princess, queen, king, monarch

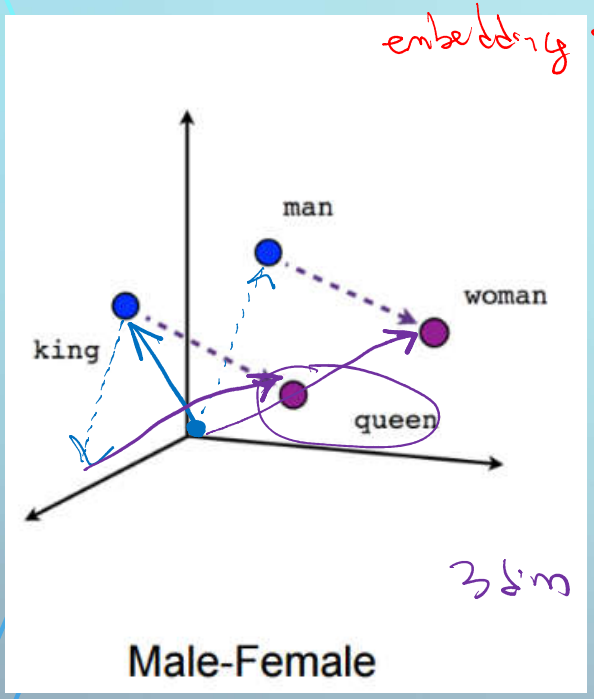


	[feminine]	[youth]	[royalty]
Man	0	0	0
Woman	1	0	0
Boy	0	1	0
Girl	1	1	0
Prince	0	1	1
Princess	1	1	1
Queen	1	0	1
King	0	0	1
Monarch	0.5	0.5	1

https://www.shanelynn.ie/get-busy-with-word-embeddings-introduction/

Representation learning / embeddings

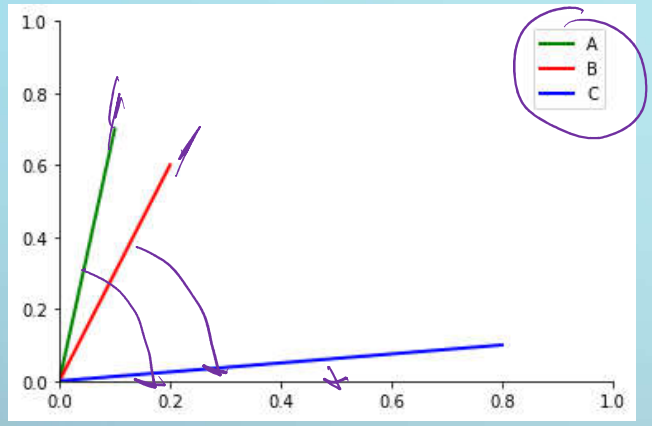
Embeddings → interpretaciones



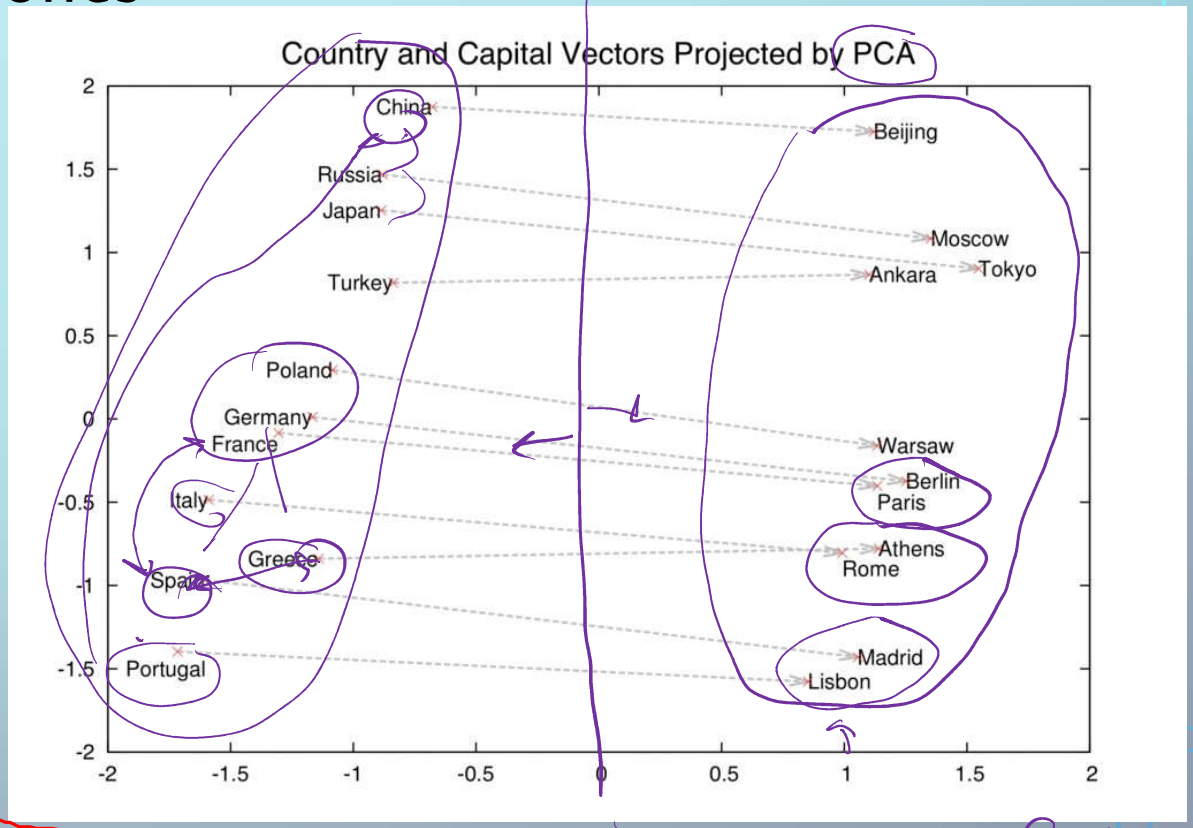
[[king]] – [[man]] + [[woman]] = [[queen]]

Representation learning / embeddings

Embeddings → interpretaciones



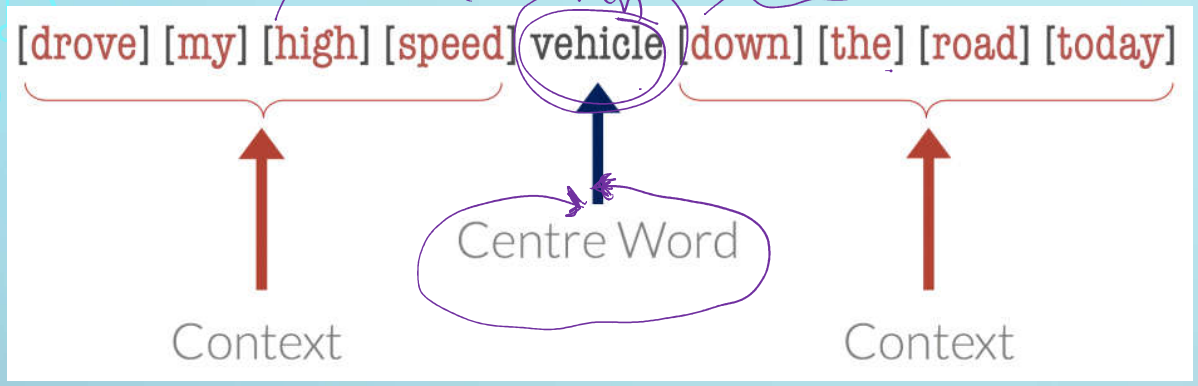
$$cos(\theta) = \frac{A \cdot B}{\|A\| \, \|B\|}$$

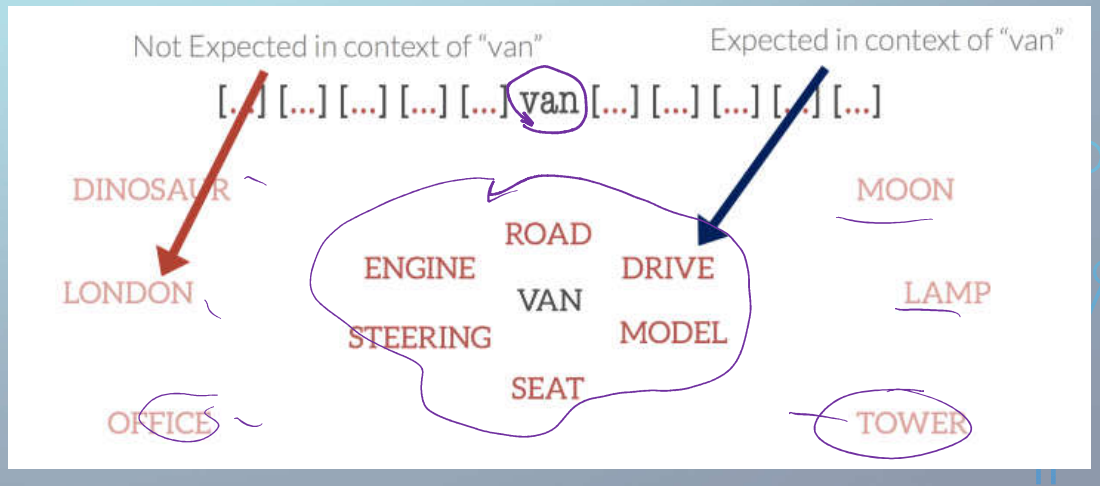


[[Paris]] - [[France]] + [[Germany]] = [[Berlin]]

Representation learning / embeddings

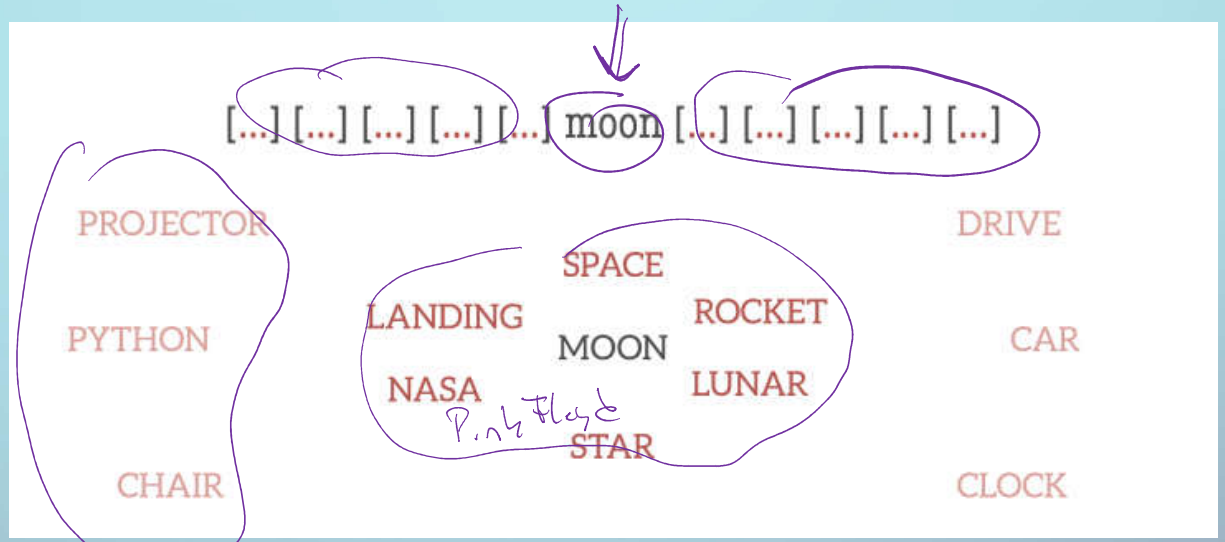
Embeddings → entrenamiento en base del contexto





Representation learning / embeddings

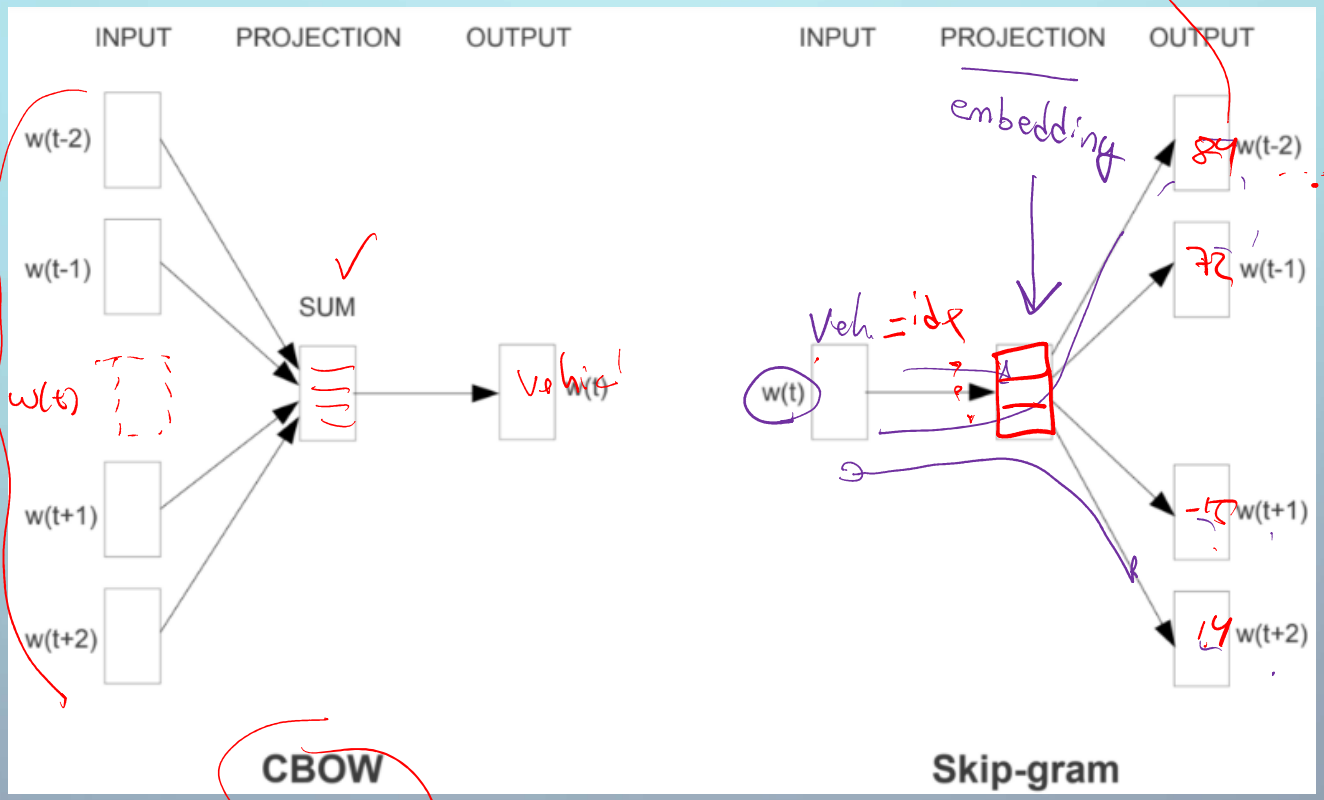
Embeddings → entrenamiento en base del contexto



Entrenamiento con predictores, usando esquemas tales como: Skip Gram, Continuous Bag of Words (CBOW), and Word2Vec… siguiendo una false task

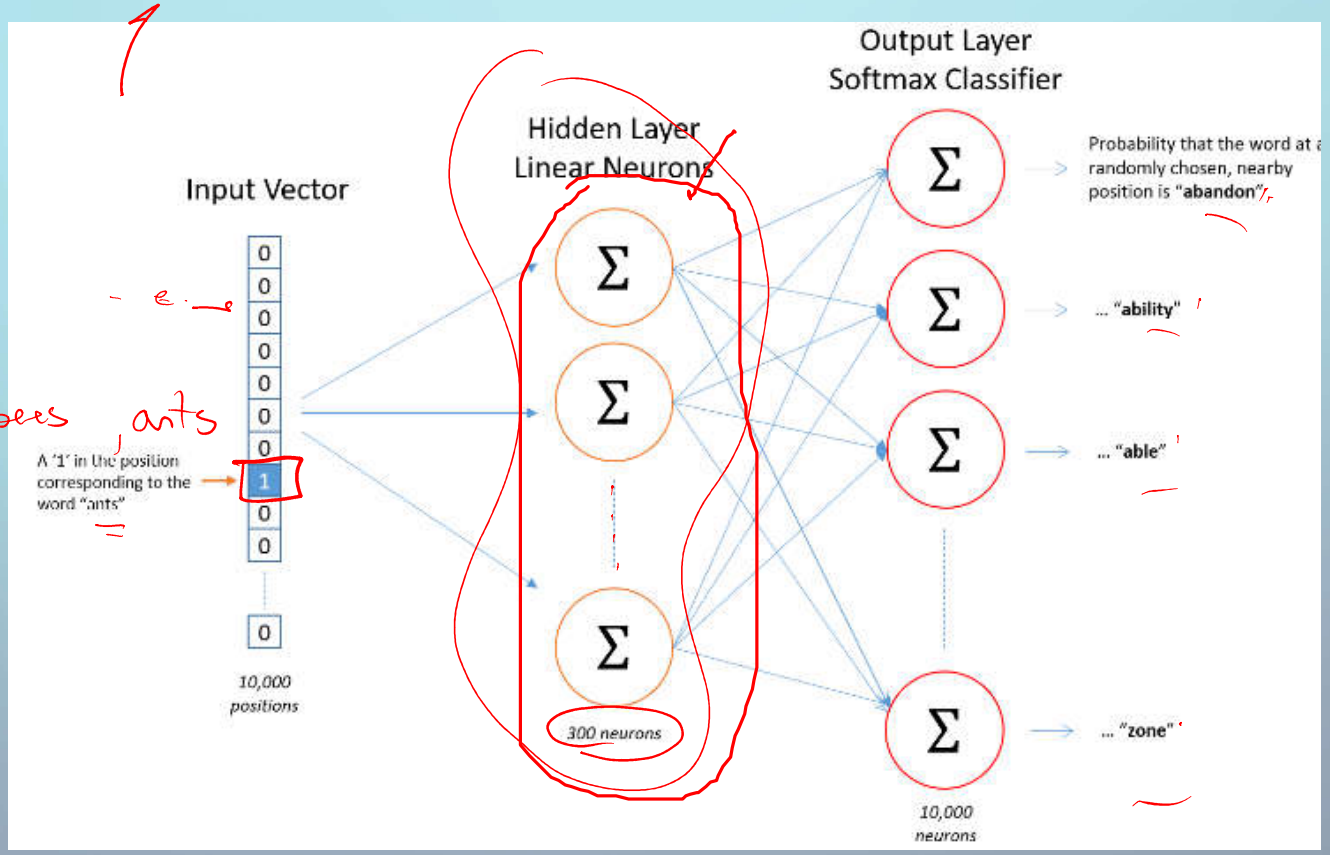
Representation learning / embeddings

Embeddings → CBOW y skip-gram



Representation learning / embeddings

Embeddings → word2vec = skip-gram + CBOW



Representation learning / embeddings

Embeddings → no solo se aplica a palabras….

Entrenamiento con predictores, usando esquemas tales como: Skip Gram, Continuous Bag of Words (CBOW), and Word2Vec

ver colab

¡Un merecido descanso!

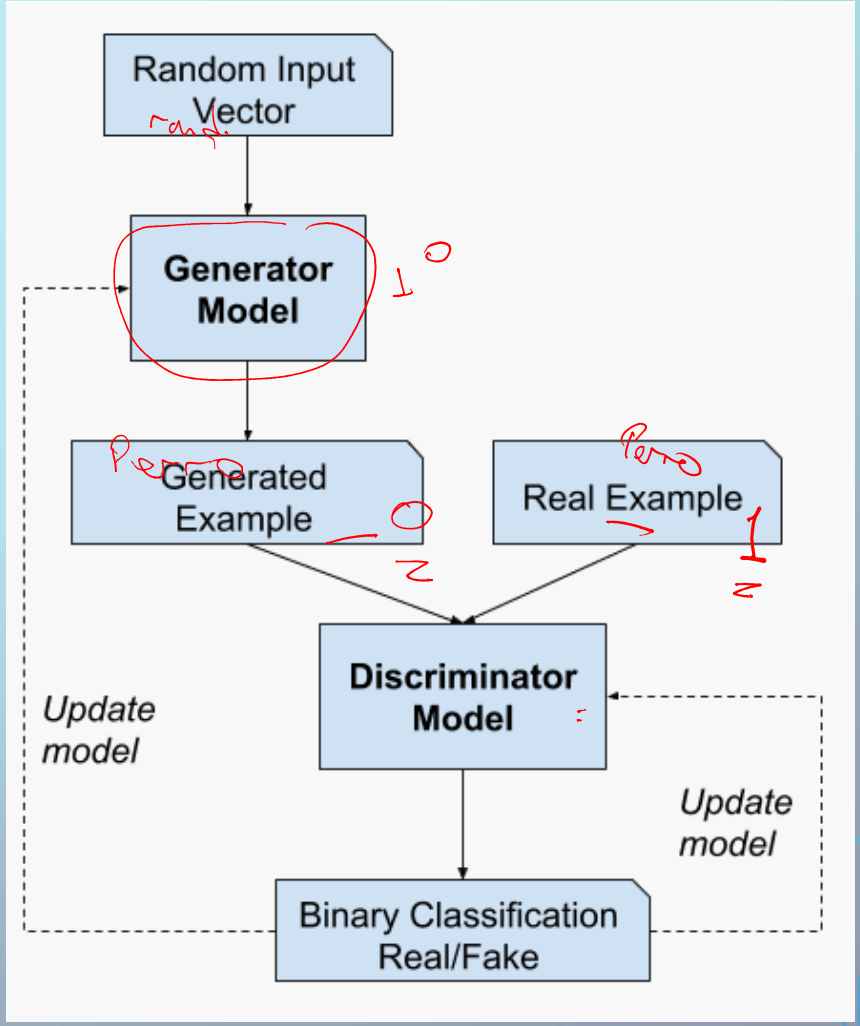
Generative Adversarial Network (GAN)

2 redes neuronales enfrentadas:
Generador - Discriminador

G → debe aprender una func de prob sobre los objetos que deseamos crear a partir de un RND (vector)

Quiero generar un perro → ¿cual es la func de prob para que tome un valor de ella y saque un perro?

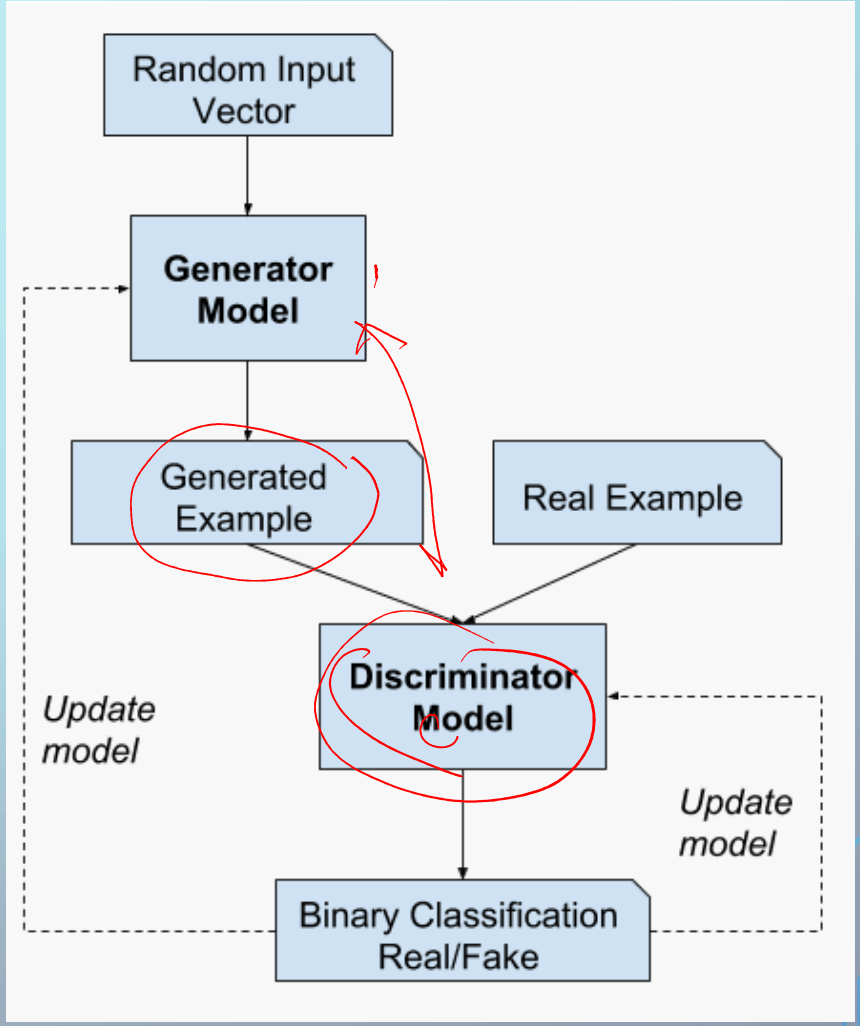
generative



Generative Adversarial Network (GAN)

G → se entrena de forma indirecta (“supervisada”… pero de distinta forma)

D → determina si una muestra es real 1 o falsa 0



entrenamiento

G → se entrena para que D falle

D → se entrena para no fallar

adversarial

Generative Adversarial Network (GAN)

z – vector aleatorio

x – vector muestra

pz – func prob z

pg – func prob G

pr – func prob x

G – generator (NN)

D – discriminator (NN)

$$\min_G \max_D L(D,G) = \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z)))]$$
$$= \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{x\sim p_g(x)}[\log(1-D(x)]$$

$$L(G,D) = \int_x \left(p_r(x)\log(D(x)) + p_g(x)\log(1-D(x)) \right) dx$$

Generative Adversarial Network (GAN)

z – vector aleatorio

x – vector muestra

pz – func prob z

pg – func prob G

pr – func prob x

G – generator (NN)

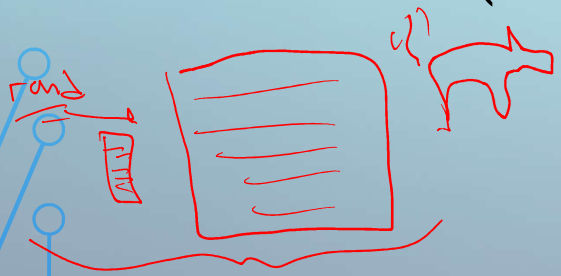
D – discriminator (NN)

$$\min_G \max_D L(D,G) = \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z)))]$$
$$= \mathbb{E}_{x\sim p_r(x)}[\log D(x)] + \mathbb{E}_{x\sim p_g(x)}[\log(1-D(x)]$$

KL (Kullback–Leibler) divergence

$$D_{KL}(p\|q) = \int_x p(x)\log\frac{p(x)}{q(x)}dx$$

D_{KL} achieves the minimum zero when $p(x) == q(x)$ everywhere.



Generative Adversarial Network (GAN)

$$L(G,D) = \int_x \left(p_r(x)\log(D(x)) + p_g(x)\log(1-D(x)) \right) dx$$

$$\tilde{x} = D(x), A = p_r(x), B = p_g(x)$$

$$\begin{aligned} f(\tilde{x}) &= A log\tilde{x} + Blog(1-\tilde{x}) \\ \frac{df(\tilde{x})}{d\tilde{x}} &= A\frac{1}{ln10}\frac{1}{\tilde{x}} - B\frac{1}{ln10}\frac{1}{1-\tilde{x}} \\ &= \frac{1}{ln10}(\frac{A}{\tilde{x}} - \frac{B}{1-\tilde{x}}) \\ &= \frac{1}{ln10}\frac{A-(A+B)\tilde{x}}{\tilde{x}(1-\tilde{x})} \end{aligned}$$

$$D^*(x) = \tilde{x}^* = \frac{A}{A+B} = \frac{p_r(x)}{p_r(x)+p_g(x)} \in [0,1].$$

Se asume que G y D tiene capacidad infinita y convergen…

Entonces pg ≅ pr => D* = 1/2

Generative Adversarial Network (GAN)

What is the global optimal?

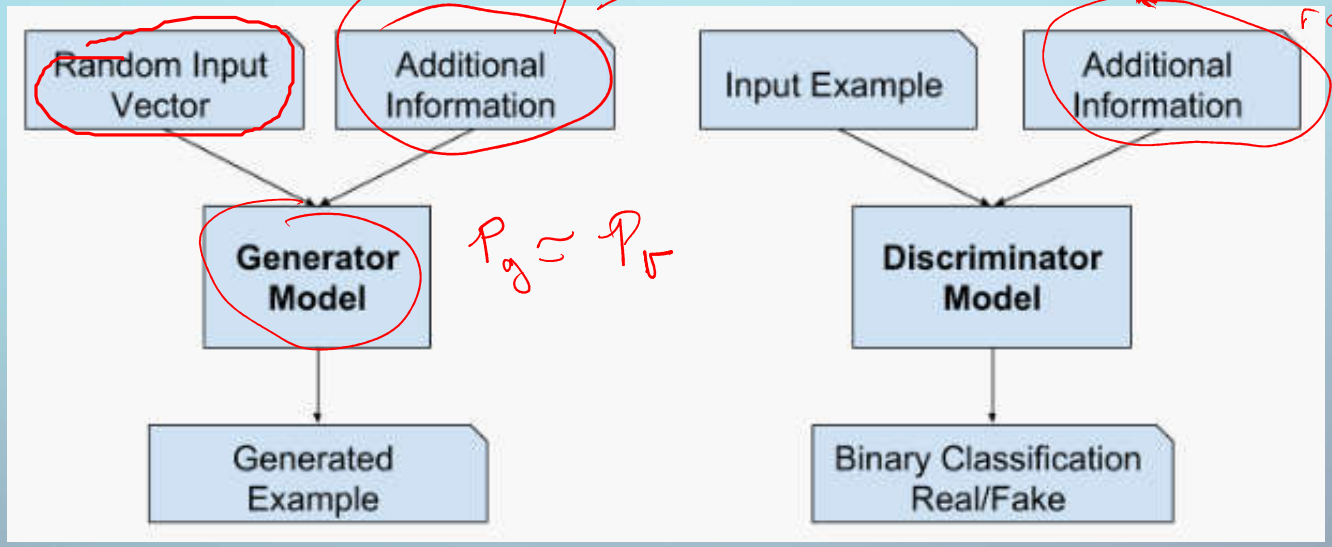
When both G and D are at their optimal values, we have $p_g = p_r$ and $D^*(x) = 1/2$ and the loss function becomes:

$$
\begin{aligned}
L(G, D^*) &= \int_x \Big(p_r(x) \log(D^*(x)) + p_g(x) \log(1 - D^*(x)) \Big) dx \\
&= \log \frac{1}{2} \int_x p_r(x) dx + \log \frac{1}{2} \int_x p_g(x) dx \\
&= -2 \log 2
\end{aligned}
$$

Generative Adversarial Network (GAN)

Conditionals GANs

Se le pasa un 'label' para que genere algo bajo ese 'label'

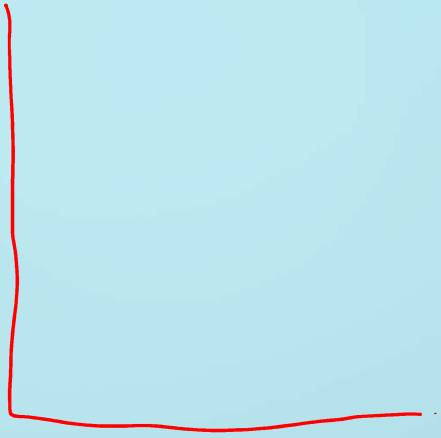


https://machinelearningmastery.com/impressive-applications-of-generative-adversarial-networks/

Generative Adversarial Network (GAN)

GANs

- ver colab



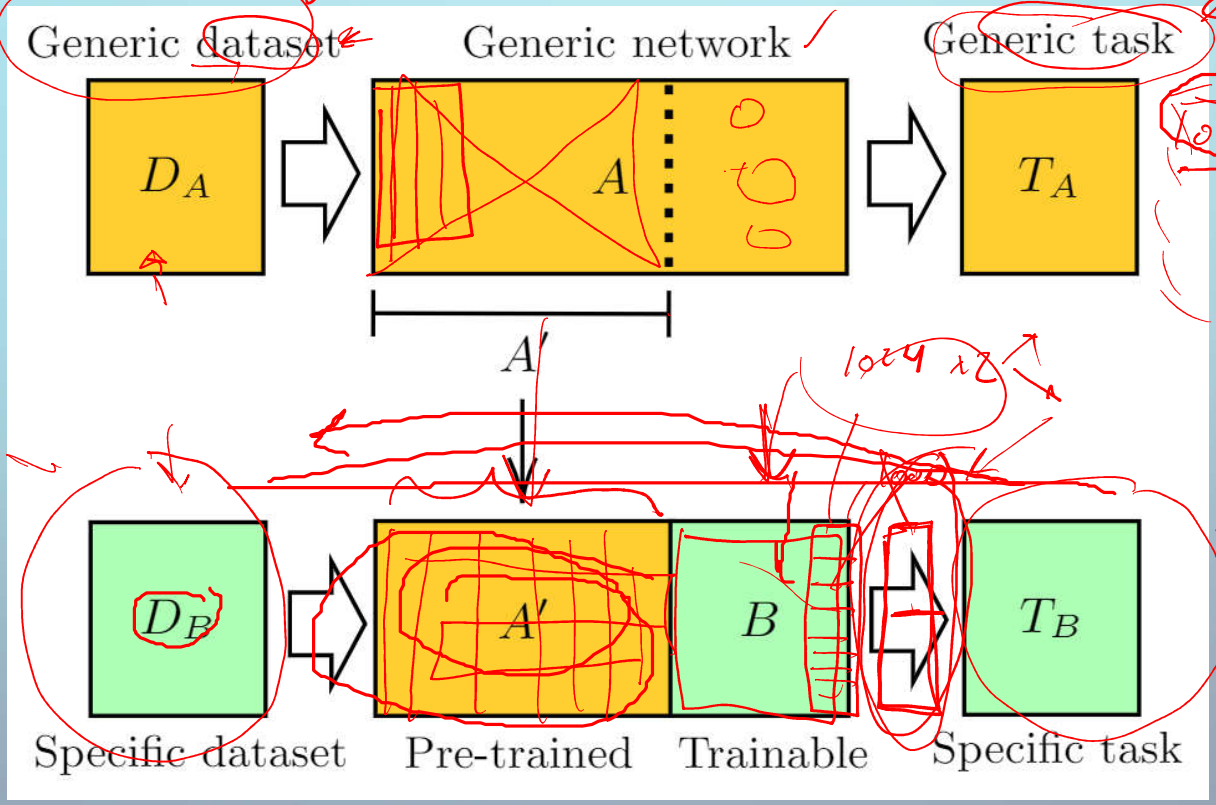
https://github.com/Yangyangii/GAN-Tutorial

¡Un merecido descanso!

Transfer Learning

No se suele entrenar un modelo desde CERO

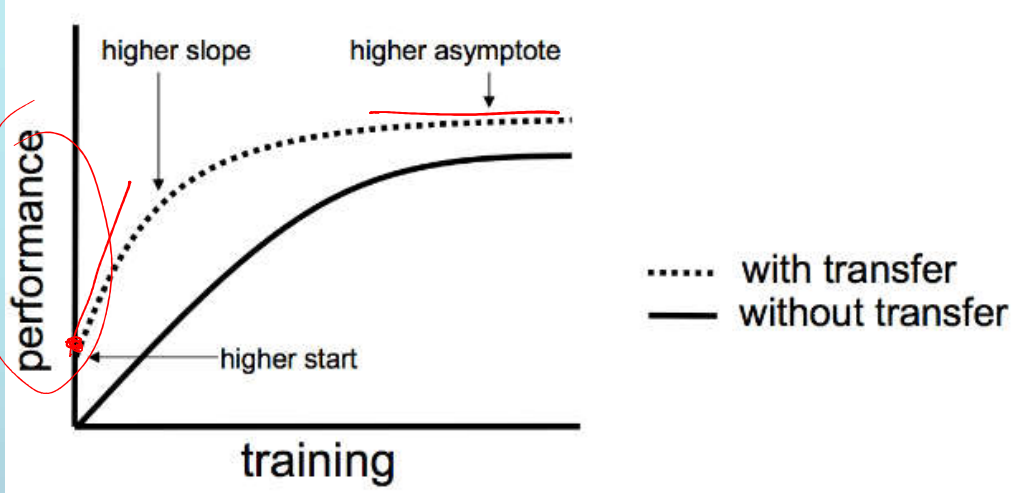
Se emplean modelos ya entrenados y se hace el ajuste necesario



Transfer Learning

Transfer learning tiene un concepto más amplio:

- pocos datos
- pre-trained models
- pre-trained embeddings
- Simulations
- Cambio de dominio



Transfer learning básico:

- ver colab