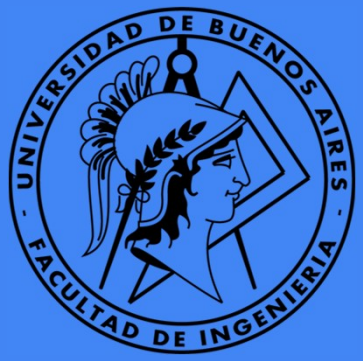
Aprendizaje Profundo

Facultad de Ingeniería
Universidad de Buenos Aires



Profesores:
Alfonso Rafael
Marcos Maillot

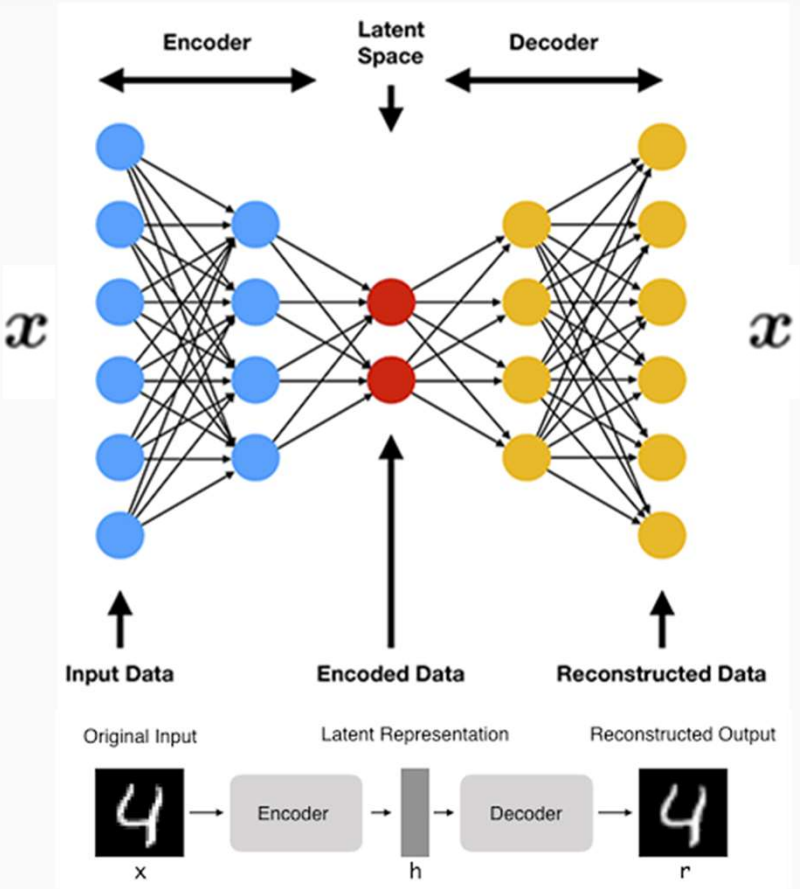
AUTOENCODER

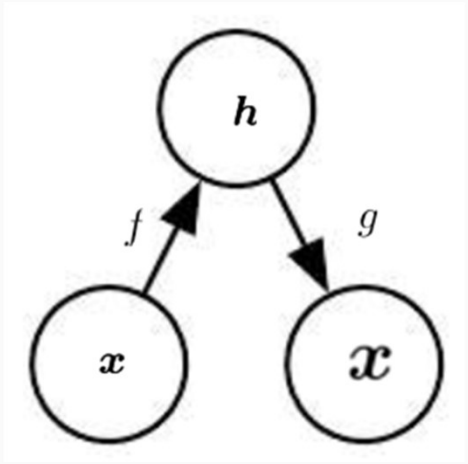
REPRESENTATION LEARNING / EMBEDDINGS

. Autoencoders

. Representation learning / embeddings

Autoencoders





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

PCA??

undercomplete autoencoders vs regularized autoencoders

Autoencoders

Undercomplete autoencoders vs **Regularized** autoencoders

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

Se reduce la capacidad del autoencoder (**undercomplete …**) para que aprenda los "aspectos relevantes" 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

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

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

Regularización L1

•**Sparse activation** – Para una entrada dada, la mayoría de los h_i producirán una activación muy pequeña.

•Para una h_i dada, su activación promedio (sobre todas las muestras) será un valor cercano a cero.

•Esto previene que el autoencoder use todas las h_i al mismo tiempo y fuerza a un uso reducido de h_i

Sparse autoencodes → para **obtener features** para otra tarea → pre-training.

Autoencoders

Constractive autoencoders CAE

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

El objetivo es aprender un representación que sea **poco sensible a pequeñas variaciones** en los datos de entrada.

- Inputs similares tendrán similares representaciones. Se fuerza al modelo a aprender un vecindario reducido de X y mapearlo a un vecindario reducido de h_i
- Undercomplete + constractive → se aprenden dh_i/dx pequeñas. Solo un número pequeño de h_i (que se corresponden con un número reducido de direcciones en X), pueden tener una derivada considerable.

Constractive autoencodes → hace que el **encoder** no varie mucho ante pequeños cambios en **x**

Autoencoders

Manifold learning con autoencoders

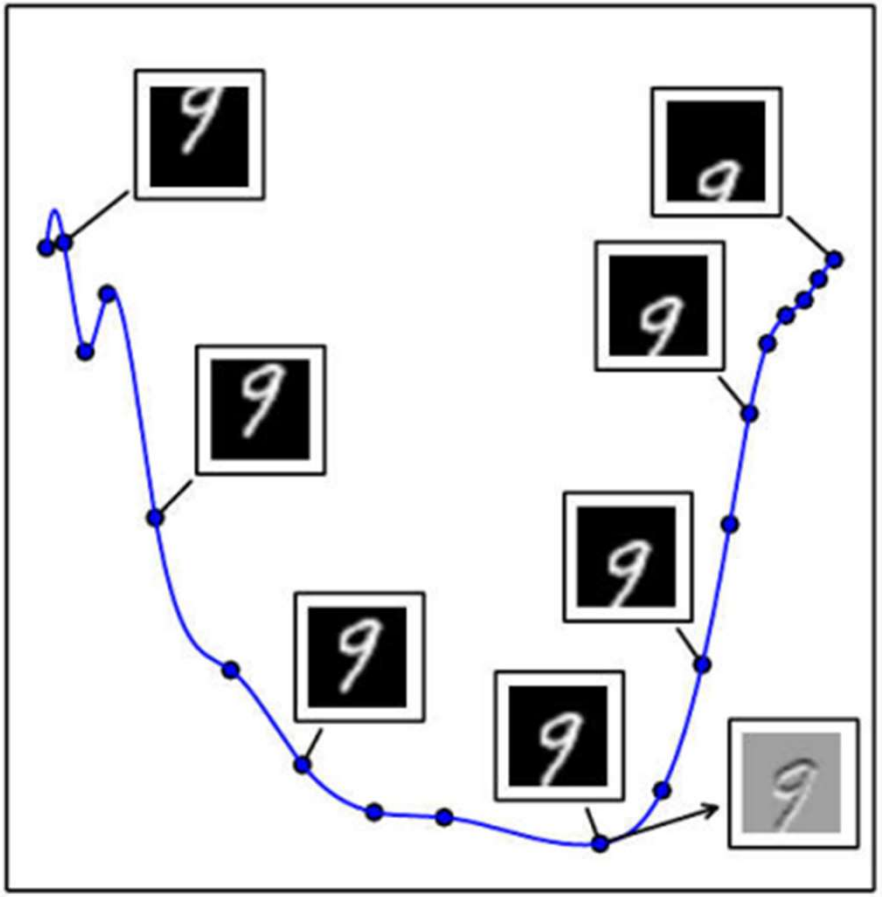
- **Manifold →** es un espacio N-dimensional, de menor dimensión del espacio original donde los datos se concentran.

- Si nos desplazamos dentro del manifold del MNIST, siempre encontraremos un número MNIST.

- Si nos salimos de dicho **manifold**, no encontraremos un número MNIST.

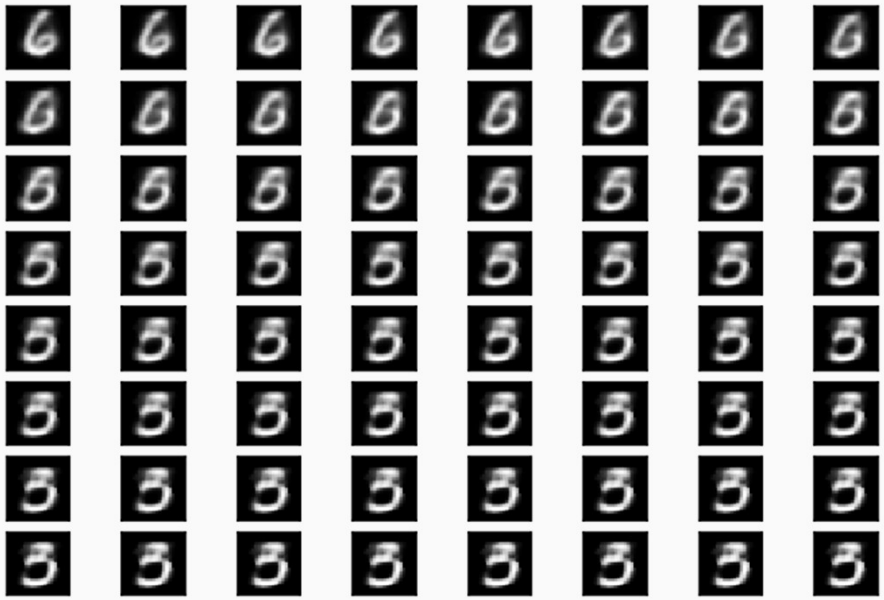
TODAS LAS NN "ENCUENTRAN" EL MANIFOLD DE LOS DATOS

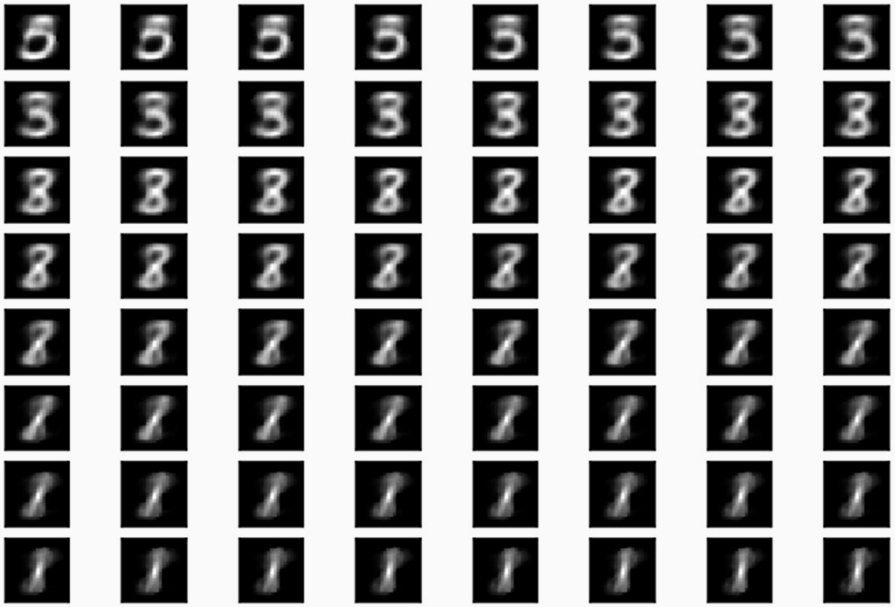
https://www.youtube.com/watch?v=QHj9uVmwA_0&t=12s&ab_channel=ArtemKirsanov



Autoencoders

Manifold learning con autoencoders



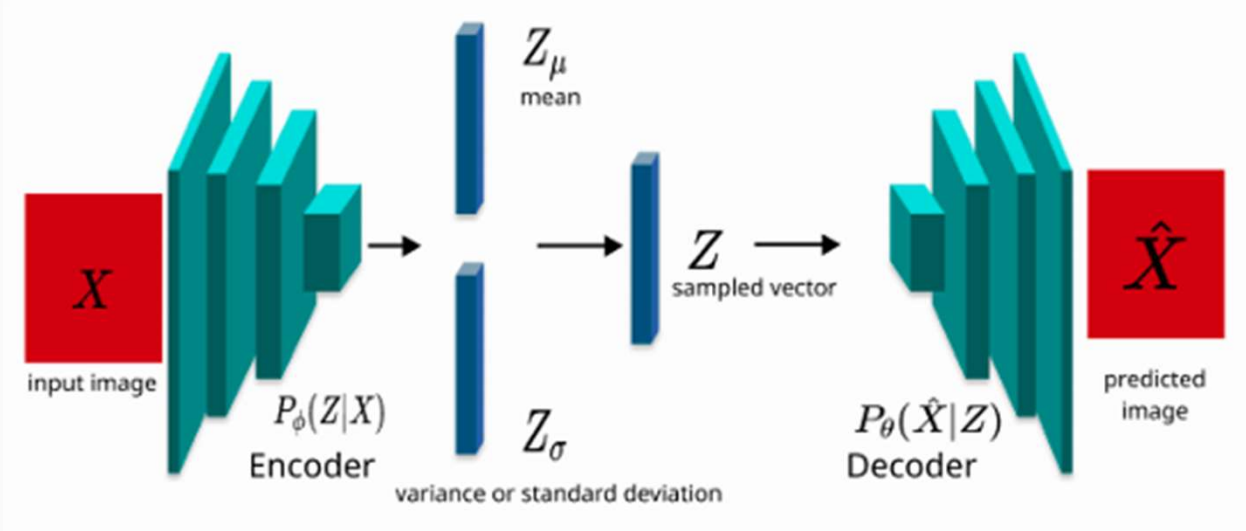


https://www.kaggle.com/apapiu/manifold-learning-and-autoencoders

Autoencoders

Varationals autoencoder - VAE

Encoder genera el espacio latente bajo una función de densidad de probabilidad.



- Otorga control sobre cómo se distribuye el espacio latent de nuestro modelo.

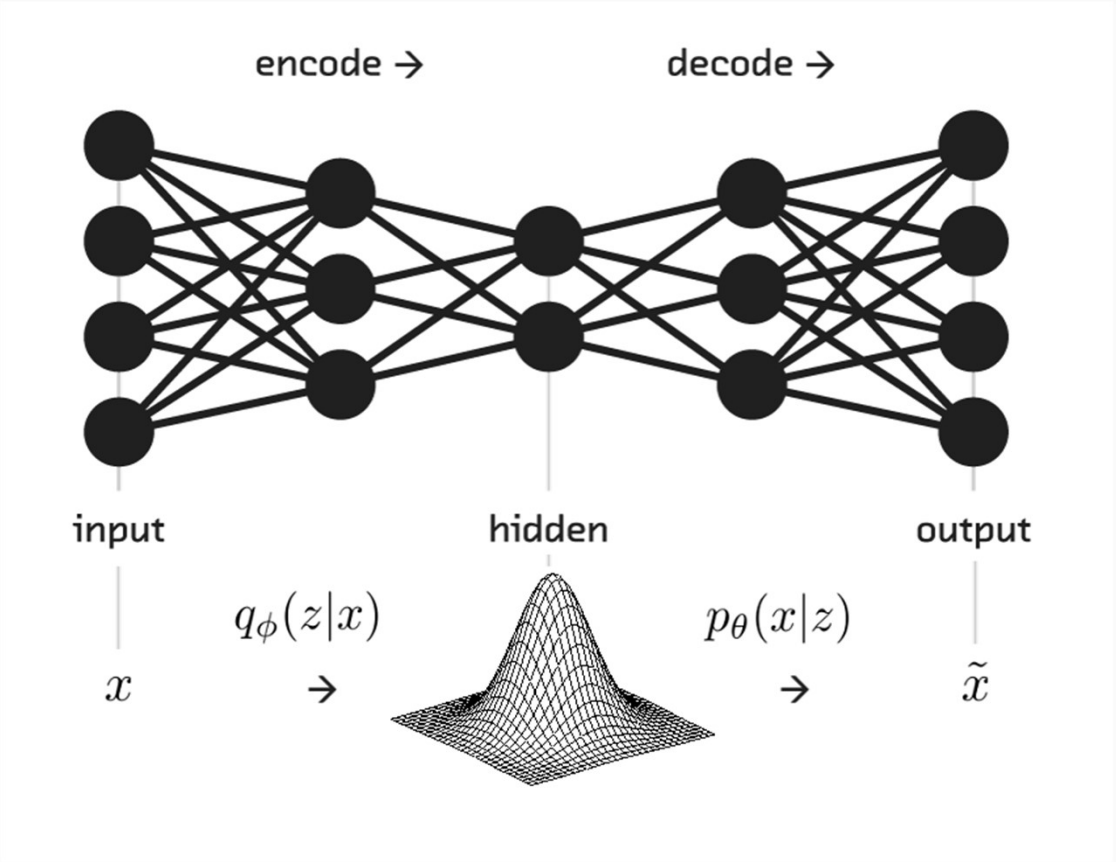
- Luego de entrenar, se toma una muesta aleatoria de dicha función de densidad de probabilidad para alimentar al decoder.

- En VAE el espacio latente Z se ve como una variable latent con una probabilidad P(Z) de que dicha variable z pertencezca al manifold de los datos de entrada.

Autoencoders

Varationals autoencoder - VAE

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

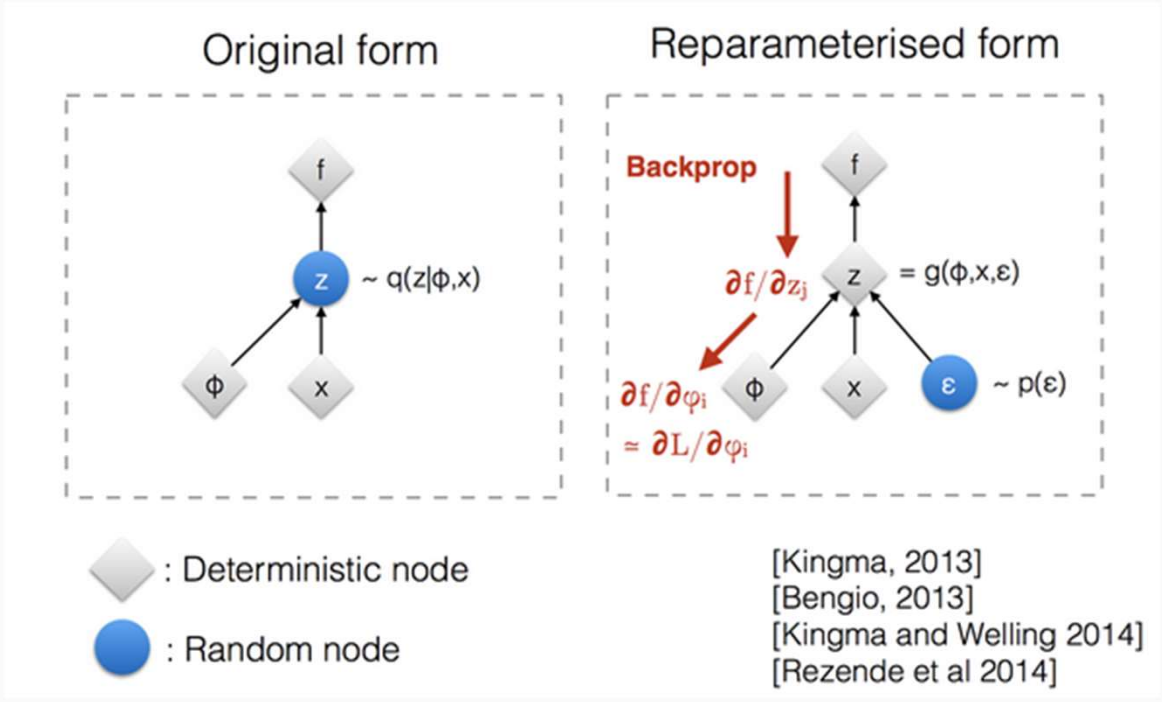


https://towardsdatascience.com/what-the-heck-are-vae-gans-17b86023588a

Autoencoders

Reparameterization Trick

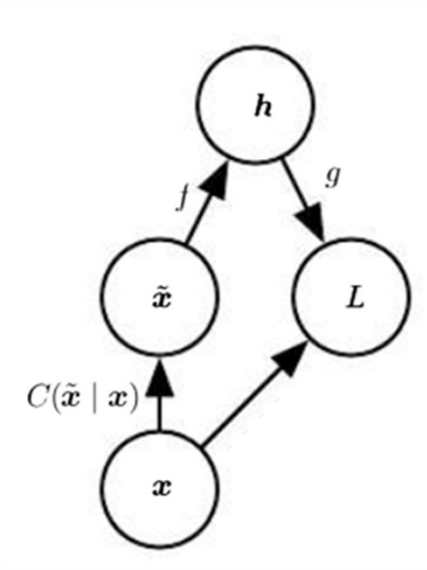
$$Z_\mu, Z_{ln(\sigma^2)} = enc(X)$$
$$\epsilon \in N(0,1)$$
$$Z = Z_\mu + \epsilon \sqrt{exp(Z_{ln(\sigma^2)})}$$
$$X_{recon\mu} = dec(Z)$$



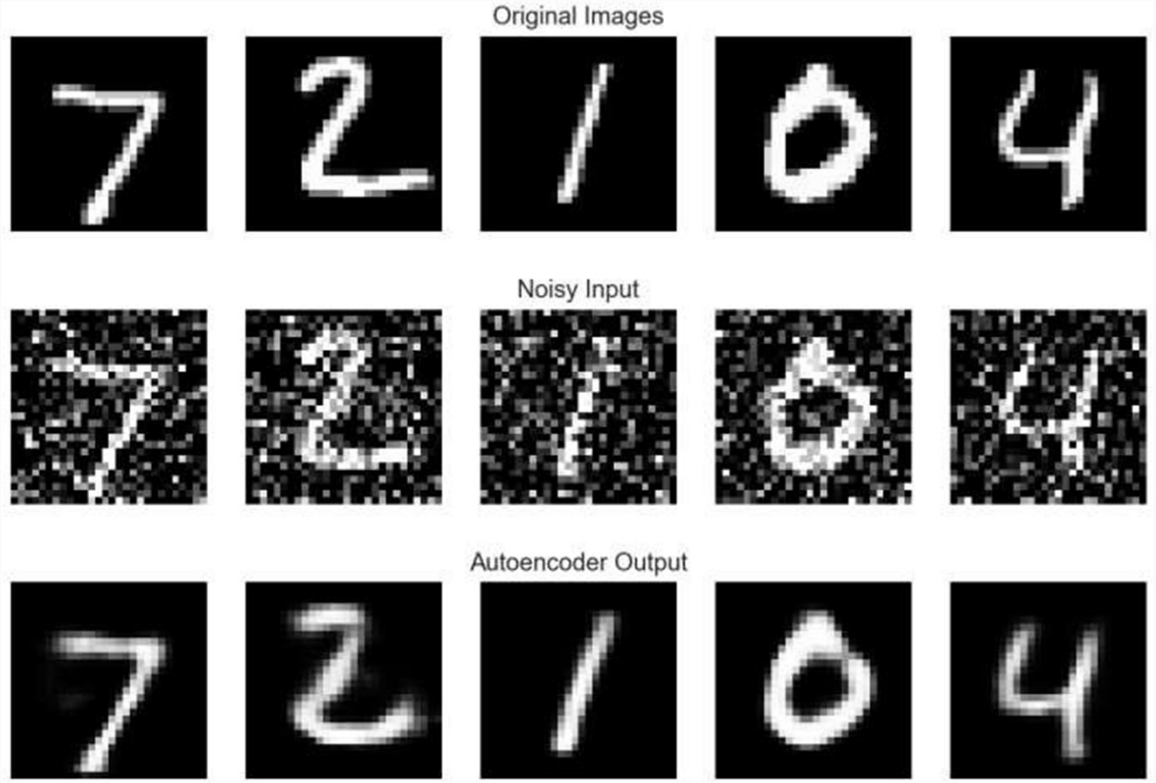
https://lilianweng.github.io/posts/2018-08-12-vae/

Autoencoders

Denoising autoencoders



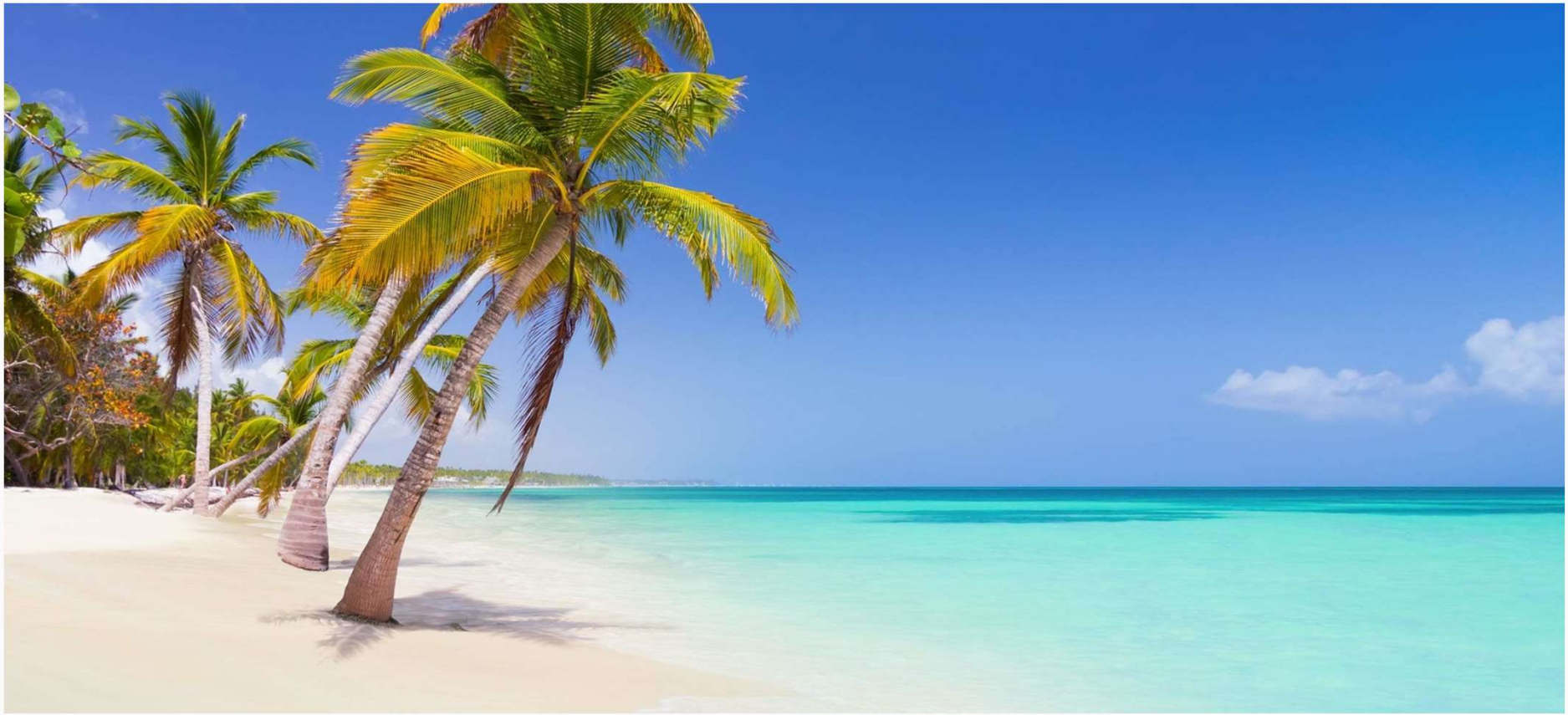
$$||g(f(\tilde{\boldsymbol{x}})) - \boldsymbol{x}||^2$$



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) siguiendo una false task.

+ Reducción de dimension del input
+ Encontrar latent variables

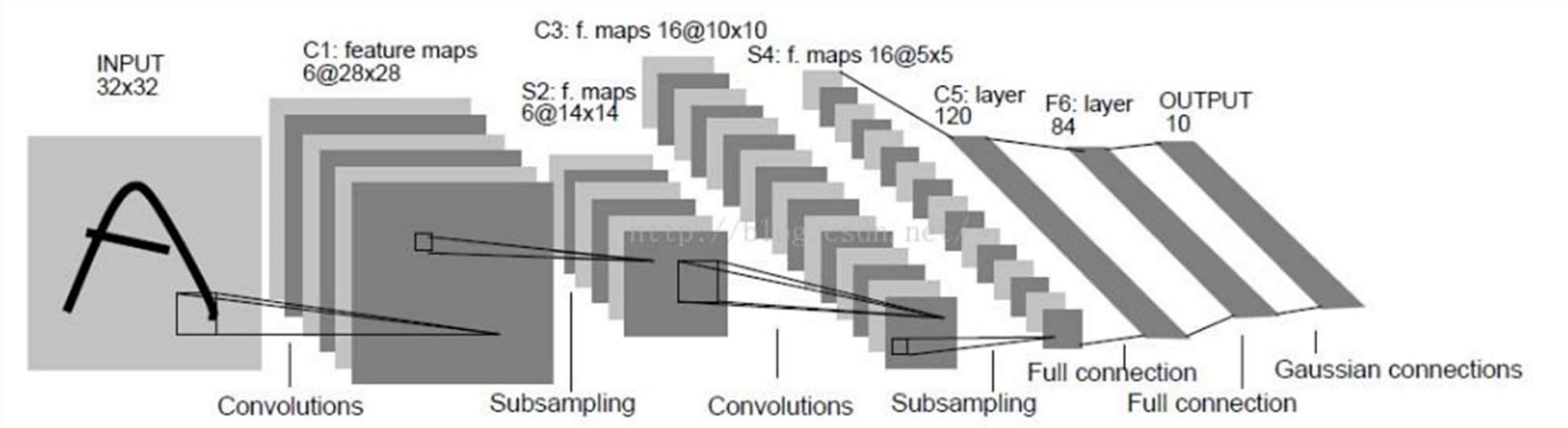
Se reduce la complejidad del dataset → se reducen las anomalias y el ruido

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



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 el vector N-dimensional se entrena a partir de su entorno.

Representation learning / embeddings

Ejemplo: vocabulario 4 palabra, embedding dimensión 2:

"pesos" inicializados "rand"

Perro → **[e00, e01] = [0.123, 0.541]**

Gato → **[e10, e11] = [0.287, 0.459]**

Caballo → **[e20, e21] = [0.594, 0.180]**

Árbol de levas → **[e30, e31] = [0.251, 0.328]**

Entrenamiento supervisado siguiendo una false task

"pesos" entrenados

Perro → **[e00, e01] = [0.873, 0.241]**

Gato → **[e10, e11] = [0.921, 0.147]**

Caballo → **[e20, e21] = [0.723, 0.541]**

Árbol de levas → **[e30, e31] = [0.003, 0.981]**

Representation learning / embeddings

Embeddings

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

	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

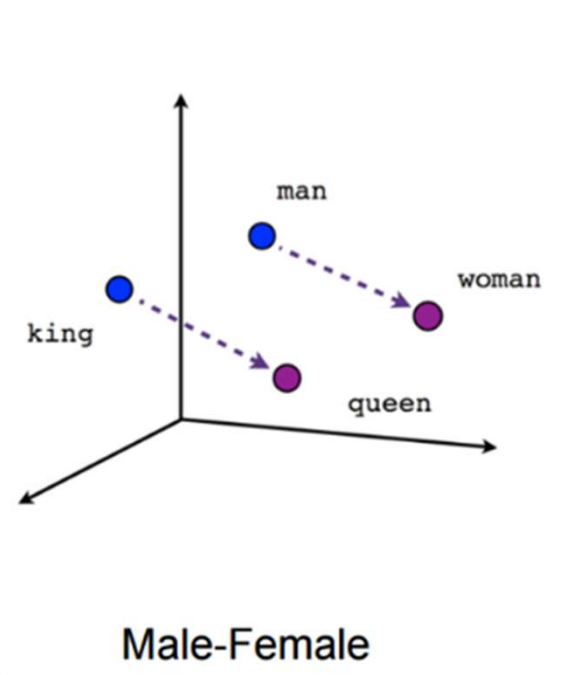
Vocabulary:
Man, woman, boy, girl, prince, princess, queen, king, monarch

	Femininity	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

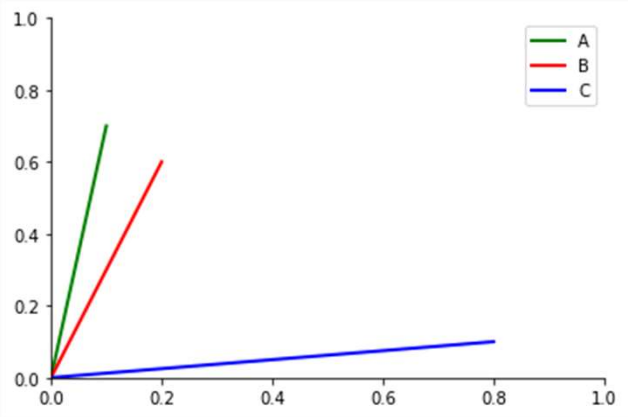
Embeddings → interpretaciones



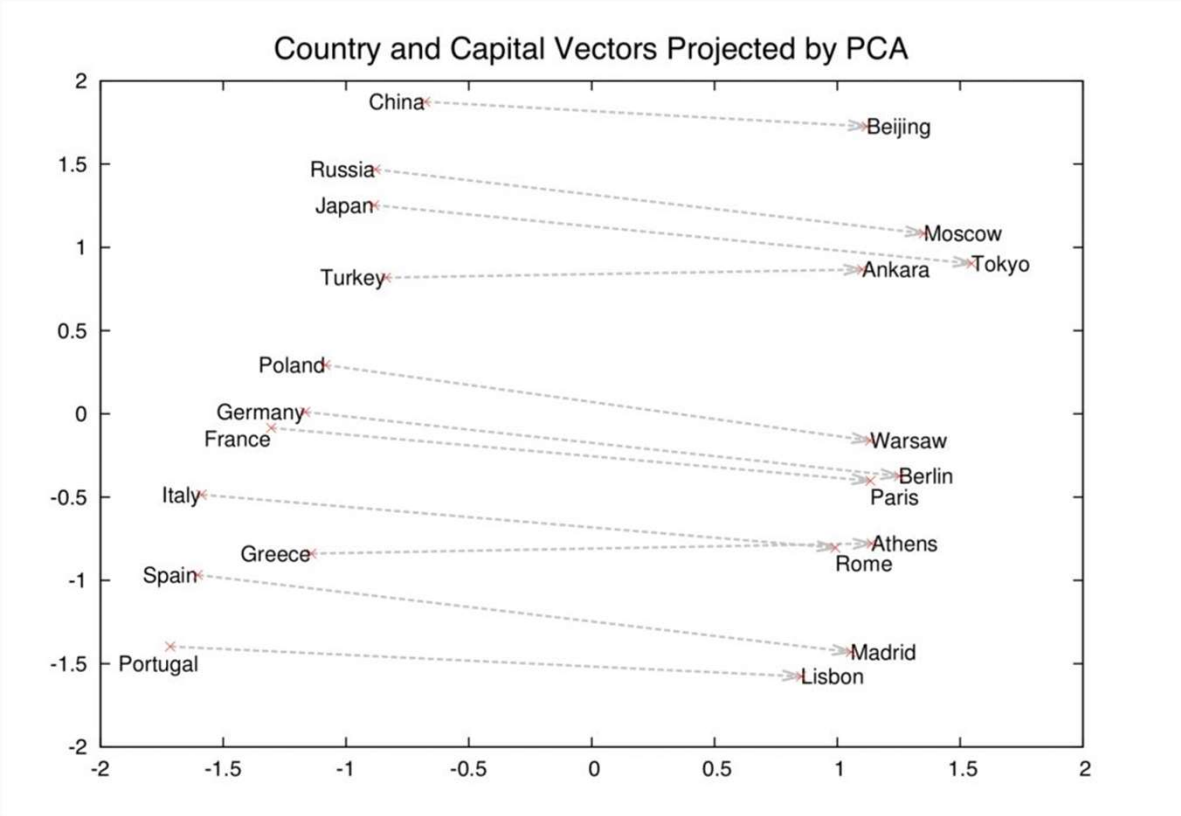
[[king]] – [[man]] + [[woman]] = [[queen]]

Representation learning / embeddings

Embeddings → interpretaciones



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

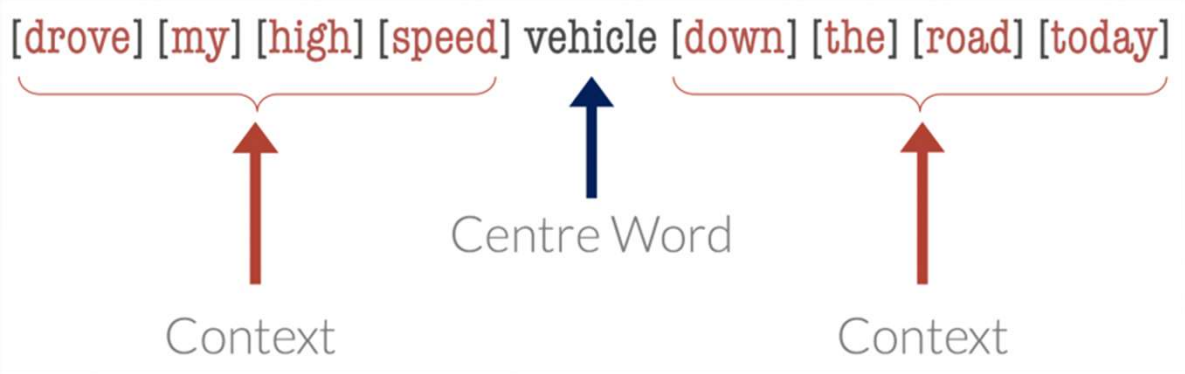


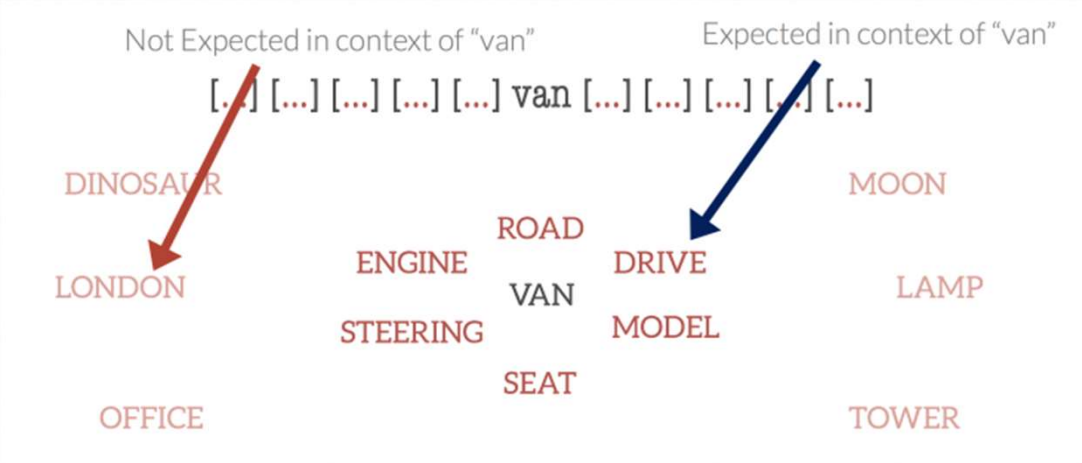
[[Paris]] – [[France]] + [[Germany]] = [[Berlin]]

https://arxiv.org/abs/1310.4546v1
Distributed Representations of Words and Phrases and their Compositionality

Representation learning / embeddings

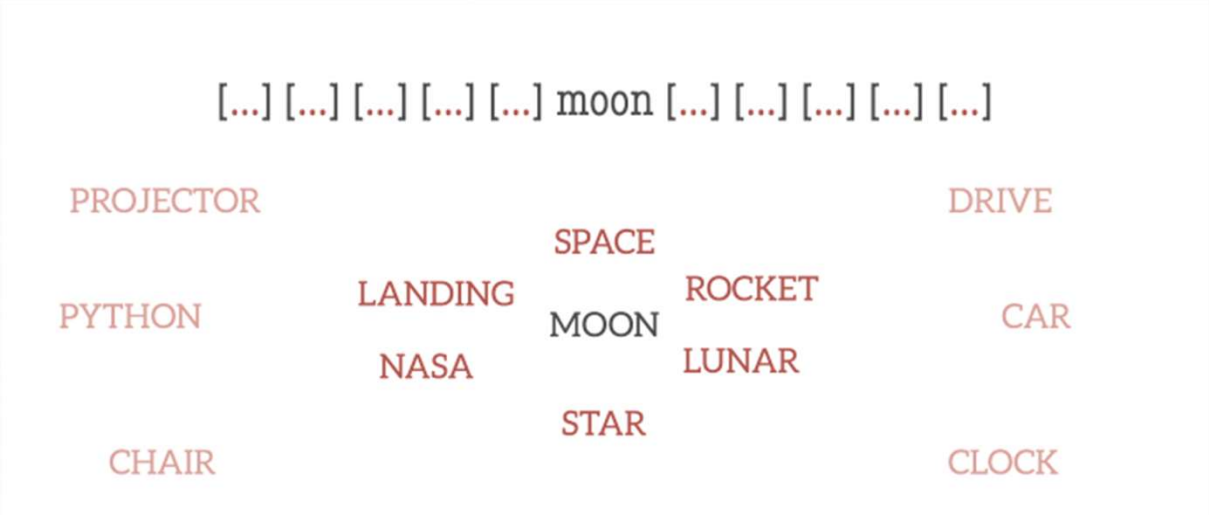
Embeddings → entrenamiento en base del contexto





Representation learning / embeddings

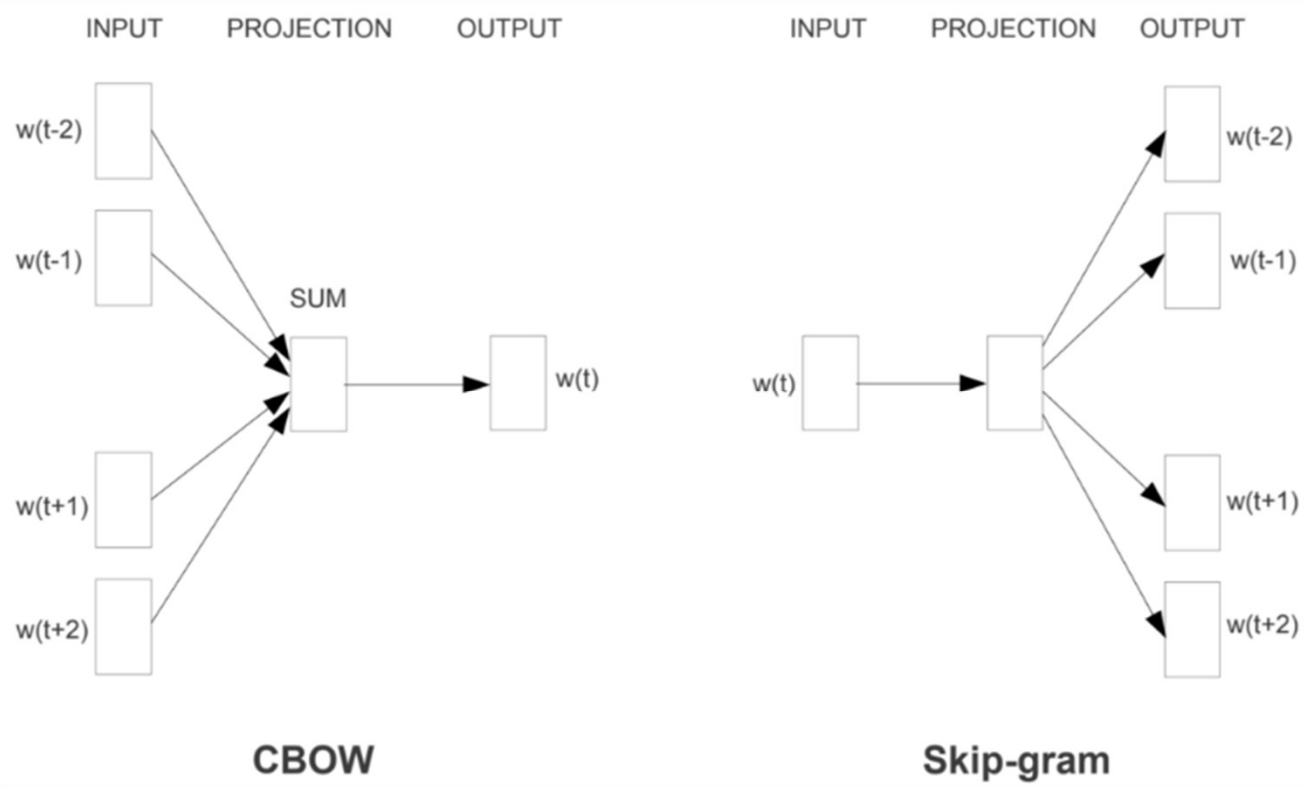
Embeddings → entrenamiento en base del contexto



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 → no solo se aplica a palabras….

ver colab