

# Codificadores Neuronales

Pedro Almagro Blanco

July 28, 2016

# Redes Neuronales Artificiales (ANN)

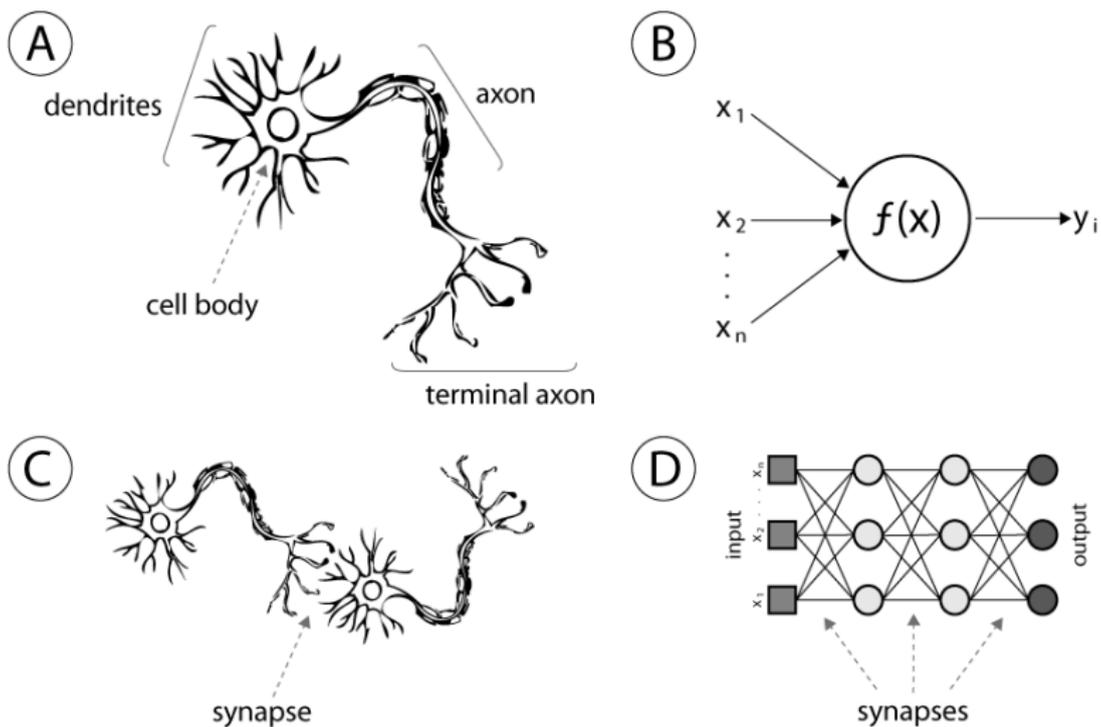


Figure : (A) Neurona humana; (B) Neurona Artificial; (C) Sinapsis biológica; (D) Sinapsis en una ANN.

# Perceptrón

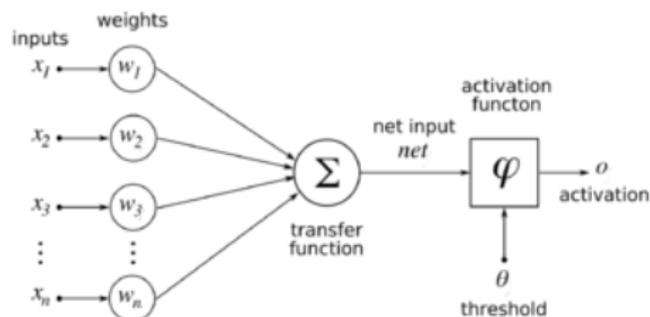
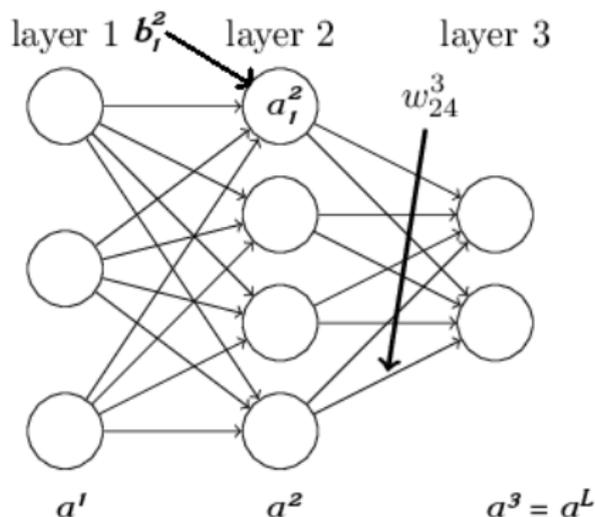


Figure : Arquitectura de un perceptrón.

$$o = \begin{cases} 1 & , \text{ si } \vec{x}\vec{w} \geq \theta \\ 0 & , \text{ si } \vec{x}\vec{w} < \theta \end{cases}$$

# Red Neuronal Artificial Feedforward Multi-cap



**Figure** : Representación de una red neuronal artificial feedforward (las conexiones entre las unidades no forman un ciclo) multicapa.

## Aprendizaje automático en Redes Neuronales

Desde el punto de vista del aprendizaje automático, el problema de aprendizaje se convierte en encontrar los pesos (y bias) adecuados para que el sistema devuelva la salida esperada para un conjunto de muestras dado.

En el caso de las Redes Feedforward Multi-Capa el algoritmo más utilizado para entrenar los pesos y bias es el algoritmo de retropropagación (backpropagation).

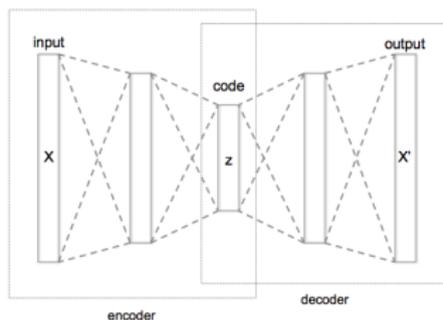
El algoritmo de retropropagación *transmite los errores* cometidos en la última capa hacia las capas anteriores para de esta forma recalcular los pesos y bias. Debido a el funcionamiento del algoritmo, la transmisión de dichos valores provoca que éstos sean cada vez más cercanos a 0 por lo que no suele ser posible entrenar por retropropagación redes neuronales con un número de capas elevado y ha sido necesario buscar métodos alternativos de aproximación o estructuras distintas a las feedforward para estos fines.

# Codificador Neuronal

**Objetivo:** Aprender una codificación a partir de un conjunto de datos.

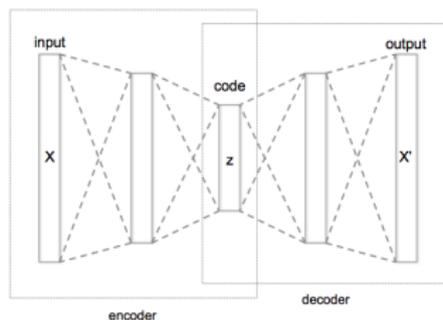
- ▶ Si estamos intentando aproximar una función por medio de una red que tiene una capa oculta, tras el ajuste de los parámetros podemos pensar que la capa oculta mantiene la información necesaria de los datos de entrada que son necesarios para el cálculo de la función.
- ▶ Desde el punto de vista de la función que calcula la red, podemos decir que la capa oculta codifica los datos de entrada.
- ▶ Los pesos (y bias) que se han usado definen la función de codificación entre ambos espacios.

# Codificador Neuronal



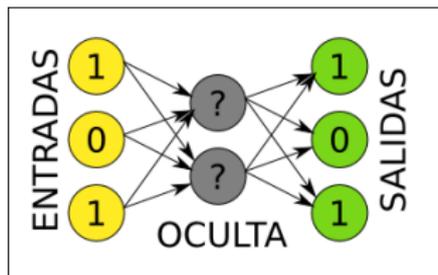
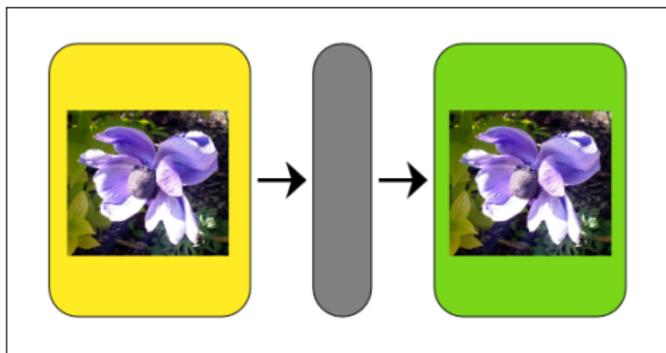
Si prescindimos de las capas posteriores, obtenemos una nueva red neuronal que produce como salida una representación del espacio de entrada en un espacio de dimensión concreta (el número de neuronas en la capa oculta que se ha considerado).

# Codificador Neuronal



Esta representación se consigue como aplicación parcial de una función que se ha obtenido a partir de una red feedforward que aproxima una función prefijada y, consecuentemente, la codificación obtenida es relativa a esta función (y al proceso de aproximación).

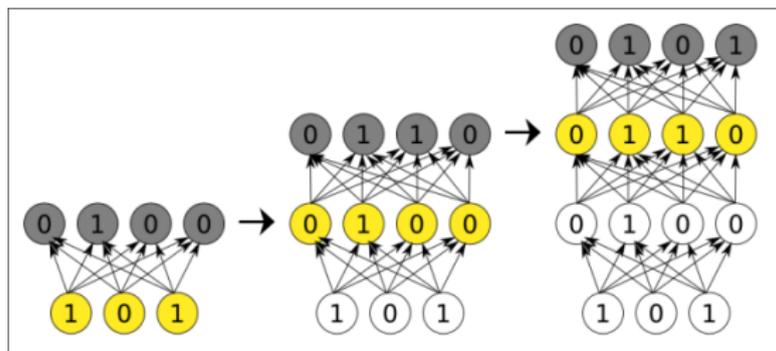
# Autocodificadores neuronales



# Autocodificadores neuronales

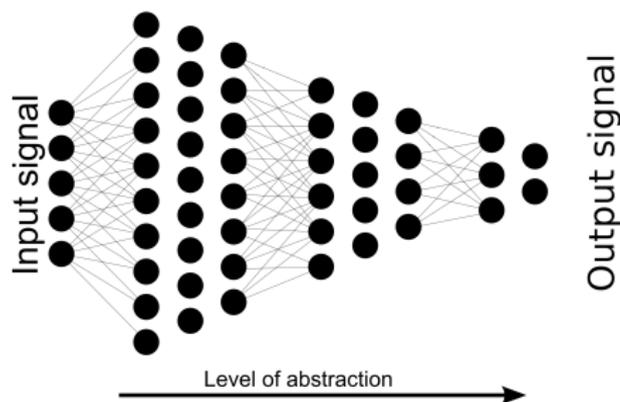
- ▶ Los autocodificadores son un caso concreto de codificador neuronal en el que se ha intentado aprender la función identidad ( $|a^1| = |a^L|$ ).
- ▶ El conjunto de muestras de entrenamiento sería  $\{(\vec{x}^1, \vec{x}^1), \dots, (\vec{x}^N, \vec{x}^N)\}$ .
- ▶ Cuando la red alcanza un estado aceptable, las activaciones en las unidades de las capas ocultas capturan información del dato original  $\vec{x}$  presentado en la capa de entrada.
- ▶ Si el número de unidades en la capa oculta usada para la codificación difiere del número de unidades en la capa de entrada (y salida) estaremos además haciendo un cambio dimensional al realizar la codificación.
- ▶ Si los tamaños de las capas ocultas son los mismos que las capas de entrada y salida se deben imponer condiciones adicionales para no estar aprendiendo la función identidad de forma trivial (factor de dispersión, ruido en la entrada...).

# Autocodificadores y Deep Learning



- ▶ Herramienta utilizada habitualmente para pre-entrenar las capas en una Red Neuronal Profunda (Deep Learning).
- ▶ Una vez entrenado un auto-codificador, la segunda mitad de la red se puede descartar, nos interesa sólo la parte que codifica.
- ▶ Apilaremos la primera mitad de varios autocodificadores, de tal manera que la codificación obtenida en el autocodificador  $l$  será el vector utilizado para entrenar el autocodificador  $l + 1$ .

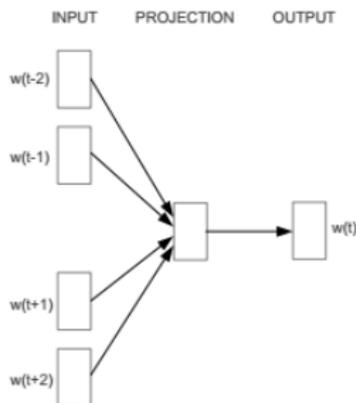
# Autocodificadores y Deep Learning



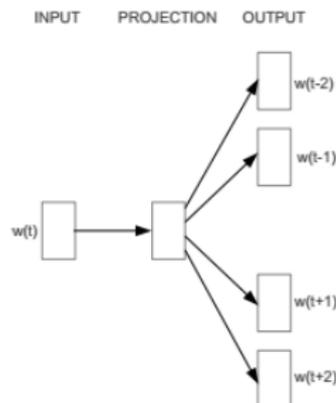
- ▶ Una vez que hemos apilado los diferentes codificadores, obtenemos una red profunda pre-entrenada a la que ahora podemos aplicar *backpropagation* sabiendo que se encuentra *cerca* de una configuración *óptima*.

# Word2Vec

- ▶ Una de las aplicaciones de los codificadores neuronales que mejores resultados ha proporcionado es la realizada con palabras.
- ▶ Word2Vec es el nombre genérico de dos arquitecturas de codificadores neuronales: *Continuous bag-of-words* (CBOW) y *Skip-gram*.

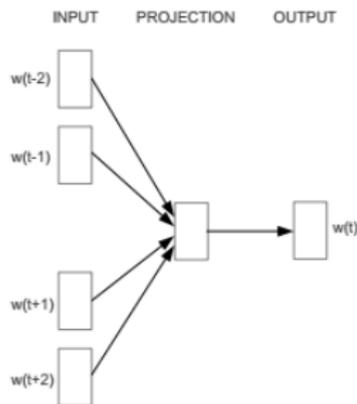


CBOW

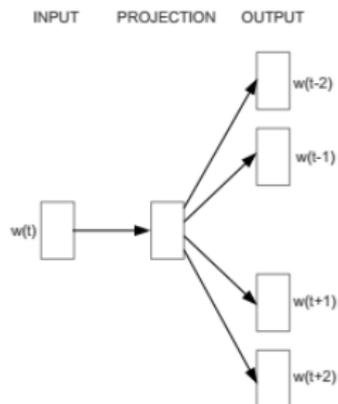


Skip-gram

# Word2Vec



CBOW



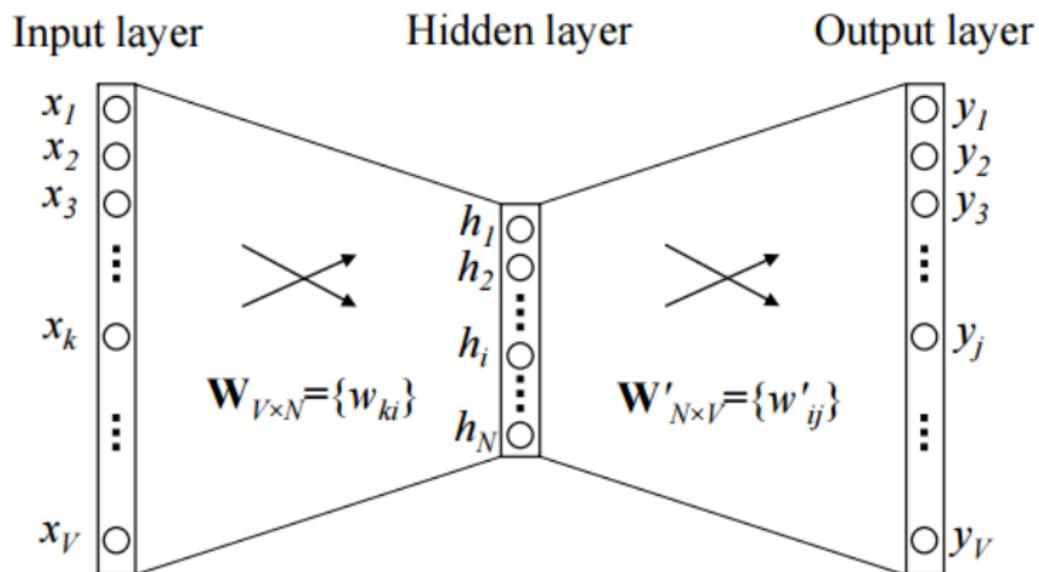
Skip-gram

- ▶ Ambas arquitecturas consisten en redes neuronales artificiales feedforward con 3 capas, pero se diferencian en la función objetivo que intentan aproximar.

## Word2Vec

- ▶ Fijaremos un vocabulario de trabajo  $V$  extraído a partir de los textos que sirven de corpus para el aprendizaje. En ocasiones se trabaja únicamente con los lemas de las palabras y/o se eliminan *stopwords*.
- ▶ Las capas de entrada y de salida tendrán tantas neuronas como palabras distintas haya en el vocabulario, de forma que si ordenamos el vocabulario,  $V = \{w^1, \dots, w^k\}$ , entonces la neurona  $i$ -ésima de esas capas codificará a  $w^i$ .
- ▶ El tamaño del contexto, también llamado *ventana*, es variable y se decide en función de la codificación a realizar, por lo que se convierte en un parámetro libre de los modelos.
- ▶ La tarea de entrenamiento supervisado se realizará con el conjunto de muestras de la forma  $(w, C)$  en el caso de la arquitectura Skip-gram, y a través de un conjunto de muestras de la forma  $(C, w)$  en el caso de la arquitectura CBOW, donde  $w$  representa una de las palabras de  $V$ , y  $C$  representa uno de los posibles contextos asociados a dicha palabra.

# Word2Vec



Tras conseguir la aproximación deseada:

- ▶ El vector de activaciones de la capa oculta  $\vec{a}^2$  que se obtiene al usar como entrada de la red una palabra determinada del vocabulario ( $\vec{a}^1 = w^i$ ) en el caso de la arquitectura Skip-gram representa la codificación de dicha palabra en el nuevo espacio vectorial.
- ▶ En el caso de la arquitectura CBOW, el vector de activaciones de la capa oculta  $\vec{a}^2$  resultante al ingresar el vector promedio de los vectores contexto asociados a una palabra determinada  $w^i$  por las entradas de la red ( $\vec{a}^1 = C^i$ ) representa la codificación de la palabra  $w^i$  asociada a dichos contextos en el nuevo espacio vectorial.

# Word2Vec

El objetivo principal del citado trabajo de Mikolov et al. es reducir la complejidad en el modelo neuronal para permitir al sistema aprender a partir de un volúmen *enorme* de datos textuales. Hasta la llegada de word2vec, ninguna arquitectura de este tipo había sido capaz de entrenarse con más de algunos millones de palabras.

Por medio de la relación que se establece entre las palabras del vocabulario y sus contextos, el modelo captura diferentes tipos de similaridad y proporciona una inmersión en el espacio vectorial que refleja estas similaridades. Así, puede capturar por ejemplo:

$$w2v(\text{"Rey"}) - w2v(\text{"Hombre"}) + w2v(\text{"Mujer"}) \approx w2v(\text{"Reina"})$$

# Word2Vec

