



**CURSO ESPECIALIZACIÓN INTELIGENCIA ARTIFICIAL Y BIG  
DATA**

**Tema 4: Redes Neuronales**

Profesor: Sebastián Rubio Valero

Diciembre 2025



ARTIFICIAL INTELLIGENCE

# 1 Introducción Redes Neuronales

Las Redes Neuronales son versátiles, potentes y escalables; lo que las hacen ideales para abordar tareas grandes y muy complicadas.

Empezamos con una panorámica de las primeras redes neuronales hasta a las neuronas multicapa. Estudiaremos los fundamentos de la R.R.N.N y los diferentes tipos.

En una segunda parte estudiaremos las biblioteca de TensorFlow y Pytorch para programar Redes Neuronales. Nos apoyaremos en algunos ejemplos sencillos. Por último veremos algunas aplicaciones reales creadas con Redes Neuronales.

Existen muchos modelos de redes diferentes, tanto en su diseño como en sus reglas de aprendizaje y funciones de activación.

Una arquitectura clásica de Redes Neuronales consta de:

- Número de capas
- Neuronas por capas
- Conexiones entre capas

Por otro lado cada Neurona también consta de otros elementos:

- La función de activación: Cada neurona se activa o se inhibe
- Reglas de aprendizaje: Modificación de los pesos en respuesta a una información de entrada

Los cambios que se producen durante el proceso de aprendizaje se reducen a la destrucción, modificación y creación de conexiones entre neuronas.

Se puede afirmar que el proceso de aprendizaje ha finalizado (la red ha aprendido) cuando los valores de los pesos permanecen constantes. Hay tres formas de entrenar un modelo de Redes Neuronales:

1. Aprendizaje por corrección de errores: Se ajustan los pesos en función de la diferencia entre los valores deseados y los obtenidos; es decir, en función del error
2. Aprendizaje por refuerzo: Durante el entrenamiento se indica mediante una señal de refuerzo si la salida obtenida en la red sea ajusta a la deseada, y en función de ello se ajusta los pesos basándose en un mecanismo de probabilidad.

3. Aprendizaje estocástico: Consiste en realizar cambios aleatorios en los valores de los pesos de las conexiones de la red y evaluar su efecto a partir del objetivo deseado y de la distribución de probabilidad

## 2 Cálculos Lógicos con Redes Neuronales

Las primeras Redes Neuronales fueron creadas por McCulloch y Pitts y se basaron en la biología de las neuronas biológicas.

Lo primero que pensaron fué construir redes neuronales que hiciesen cálculos lógicos, asumiendo que una neurona se activa[ cuando al menos dos de sus conexiones de entrada están activa.

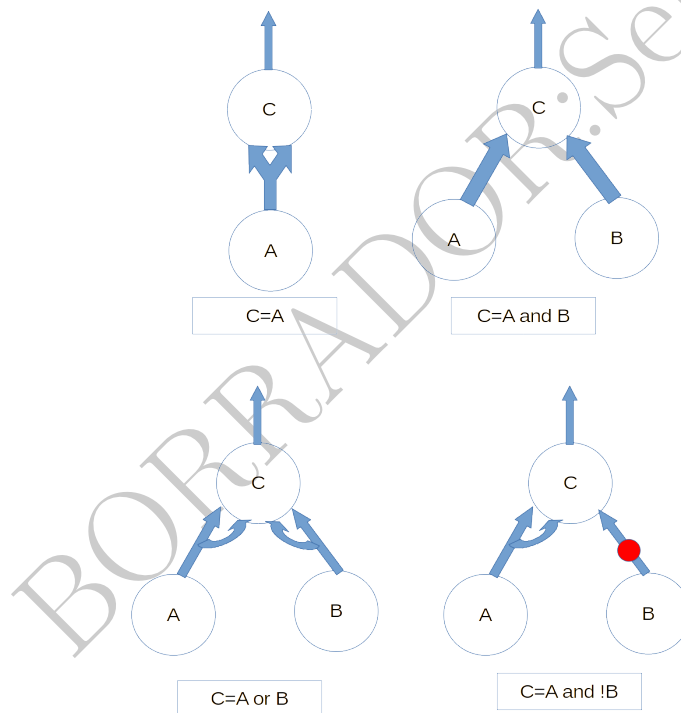


Figure 1: Enter Caption

- $C=A$  : Es la función identidad, si la neurona A esta activada entonces la neurona C también está activa( y.q, recibe dos señales de entrada de la neurona A); pero

si A está desactivada, C también lo está.

- $C=A \text{ and } B$ : Esta segunda realiza un and lógico; la neurona C se activa sólo cuando las neuronas A y B están activadas. Deben estar las dos activadas con una sola no es suficiente
- $C=A \text{ or } B$ : Esta tercera realiza una operación lógica or; la neurona C se activa si la neurona A o la B (o ambas) están activadas.
- $C=A \text{ and } !B$ : La neurona C se activa sólo si la neurona A está activada y la B desactivada. Si la neurona A está activa todo el tiempo, obtenemos un Not lógico; la neurona C se activa cuando B es activada y se desactiva cuando B es desactivada.

### 3 Concepto de aprendizaje neuronal

Decimos que una neurona puede aprender, cuando la dotamos de unos cálculos que le permitan cambiar su estructura y/o función, que le permita adoptarse a los resultados deseado. Esta adaptación se realiza mediante cálculos entre las salidas deseadas y las salidas obtenidas.

El aprendizaje estará asociado a la modificación de la función local y de las conexiones entre neuronas, a la eliminación o adición de nuevos arcos. Hay tantos tipos de aprendizaje como métodos efectivos para llevar a cabo las modificaciones. Es importante señalar que la red neuronal es, de forma muy ostensible, un asociador numérico. Es decir, que al comienzo, cuando se hace análisis de datos, las instancias tienen que codificarse de forma numérica y dentro de un rango dinámico. De la misma forma, al final del cálculo el vector numérico de salida de la última capa tiene que reescribirse en términos lingüísticos.

Dentro de cada capa, el modelo de computación sigue el esquema general, que se repite en cada neurona. Ahora bien en cada capa puede aplicarse diferentes funciones locales y salida en todas las neuronas de la misma capa. Otro aspecto importante es la retroalimentación con la salida de cada neurona. El modelo formal subyacente a la computación neuronal es un grafo paralelo y dirigido en el que los nodos están ocupados por procesadores elementales paramétricos y adaptables ("neuronas artificiales" y los arcos representan las interconexiones entre procesadores y también son ajustables.

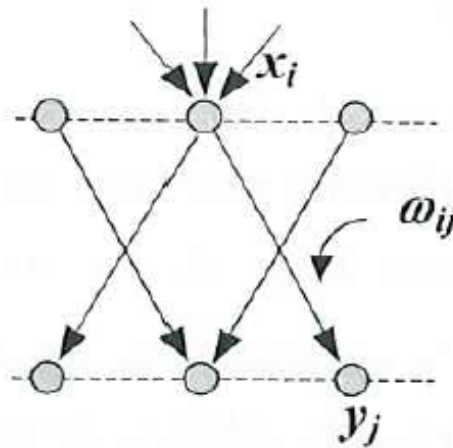


Figure 2: Enter Caption

Para describir la función realizada por un red de neuronas artificiales en cada nivel del grafo necesitamos especificar:

- Los esquemas de conectividad entre los nodos
- Las funciones asociadas a los nodos
- Las funciones de aprendizaje

Los esquemas de conectividad especifican los espacios de entrada y salida (las interconexiones) de los elementos de cálculo de una red y se clasifica en dos tipos generales:

- Redes convergentes-divergentes "hacia adelante", sin realimentación
- Redes con realimentación ("recurrentes"), en las que el resultado del cálculo de un elemento interviene como entrada al mismo o a otros elementos de su grupo funcional (capa) en el intervalo temporal siguiente.

A su vez, en ambos caso, la topología de la red puede ser:

- De conectividad total
- De conectividad local

En la figura a) es una red sin realimentación, en la b) es una red con realimentación, la c) tiene conectividad total d)conectividad local

Suponiendo que tratamos con redes de anatomía fija, en la que no podemos cambiar

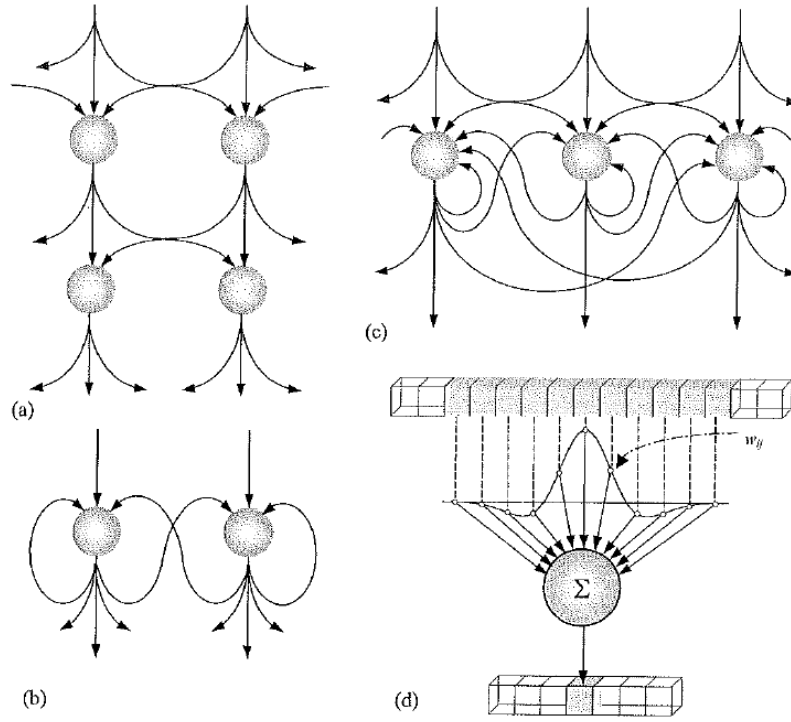


Figure 3: Enter Caption

el número de nodos ni el número de arcos, y la función local es la suma ponderada. El aprendizaje se limita a un conjunto de procedimientos de cálculo adecuados para modificar de forma autónoma y en paralelo los valores de los pesos  $w_i$ , según algún criterio de valoración del comportamiento global de la red o de las correlaciones espacio temporales entre los valores de las entradas. En el siguiente diagrama podemos observar que el aprendizaje se realiza mediante el recálculo de los pesos aplicando alguna/s funciones locales. Según esta forma de concebir el aprendizaje, se pueden distinguir tres métodos de aprendizaje.

1. Métodos no supervisados: El cálculo de los pesos  $w_{i,j}(t)$ , se realiza por una correlación espacio temporal entre las entradas  $x_i(t)$  y las salidas  $y_j(t)$
2. Métodos supervisados: En este método se conoce la salida deseada  $y_i(t)$  con respecto a una entrada  $x_i(t)$ . El cálculo de los nuevos pesos se realizan mediante el error cuadrático entre las salidas deseadas  $d_j(t)$  y las salidas obtenidas  $y_j(t)$  para la misma entrada  $x_i(t)$
3. Métodos de aprendizaje por refuerzo: Se obtiene una medida global de la red neuronal, y se minimiza esta medida global sobre las respuestas porque no se

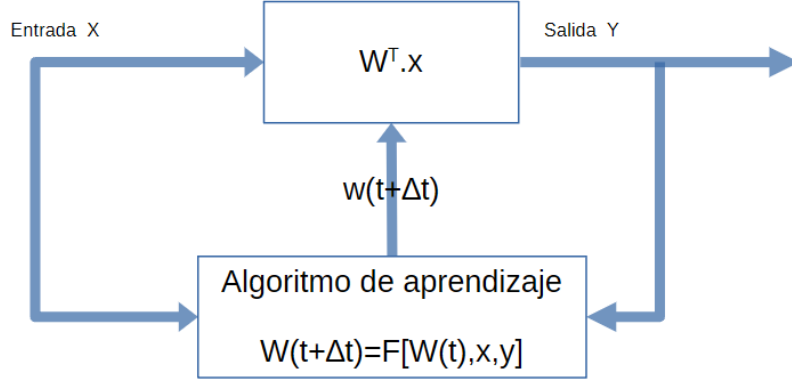


Figure 4: Enter Caption

conocen los valores  $(x_i, y_j, d_j)$  para un conjunto de entrenamiento suficientemente amplio, sino sólo el carácter positivo o negativo de la conducta global de la red. Suelen usarse, en general, asociados a formulaciones probabilísticas de la función local

Para en entrenamiento no supervisado, la forma más elemental de correlaciones es:

$$w_{j,i}(t + \Delta t) = w_{j,i} + C_l \cdot x_i(t) \cdot y_j(t)$$

En el aprendizaje supervisado, trata de construir de forma incremental la matriz de pesos,  $\mathbf{W}(\mathbf{t})$ , que mejor adapta la red a su medio a partir del conocimiento de las salidas deseadas,  $d_j$ , para un conjunto de entradas,  $x_i$ , y de las salidas actuales que produce la red para esas mismas entradas,  $y_j$ . Para la actualización incremental de la matriz de pesos se usa una medida de la distancia entre el vector deseado a la salida,  $d_j(t)$ , y el que realmente se obtiene,  $y_j(t)$ , y se modifican los peso  $w_{j,i}$  con el propósito de minimizar el error.

$$\epsilon(t) = |\mathbf{d}_j(\mathbf{t}) - \mathbf{y}_j(\mathbf{t})|$$

En otras ocasiones no se intenta minimizar directamente el error sino una función del mismo. En general, esta función es el error cuadrático medio para un conjunto de P pares de entrenamiento,  $\{\mathbf{x}_i, d_j\}$

$$E = \frac{1}{2} \sum_{j=1}^P (\mathbf{d}_j^n - \mathbf{y}_j^n)^2$$

Entre los distintos procedimientos de minimizar esta función de coste haremos énfasis en los métodos iterativos de descenso del gradiente, en los que cada iteración el nuevo

valor de  $W$  se obtiene a partir del valor antiguo dando un paso en la dirección opuesta al gradiente,

$$W_{(nuevo)} = W_{(antiguo)} - \mu \cdot \nabla_w E(w)$$

buscando alcanzar un mínimo local en la función del error.

Cuando estudiamos la regresión lineal detectamos los inconvenientes del descenso del gradiente, y es la de quedarse "atascado" en un mínimo local, o bien si la tasa de aprendizaje, que mide el ritmo de aprendizaje, compromete la convergencia, si los pasos son muy grandes.

A pesar de estos inconvenientes, el método del descenso del gradiente sigue siendo el más usado. Entre otras razones por su uso en las redes multicapa donde puede retropapagarse el error siempre que la función de decisión sea derivable y, por consiguiente, puede hacerse llegar a todas las unidades de la red la parte correspondiente del error cometido por las unidades de la última capa, que son las únicas que podemos evaluar ( en los modelos supervisados)

## 4 Funciones de cálculo local