

# Redes de neuronas

Fundamentos de la  
inteligencia artificial

The background of the slide is a dynamic, abstract digital composition. It features a diagonal split between a white upper-left area and a dark blue/purple lower-right area. The dark area is filled with glowing blue and white binary code (0s and 1s), a faint DNA double helix, and a large, dark, reflective sphere. The overall aesthetic is high-tech and futuristic.

**tech**



# CONTENIDO

1. Objetivos

---
  2. Introducción

---
  3. Fundamentos biológicos

---
  4. Modelo computacional


---
  5. Redes de neuronas supervisadas y no supervisadas

---
  6. Perceptrón simple

---
  7. Perceptrón multicapa

---
  8. Resumen

---
  9. Bibliografía

---
- 

## OBJETIVOS

- Identificar los componentes biológicos de una neurona.
- Comprender la clasificación y características de las redes neuronales.
- Conocer las redes neuronales tanto supervisadas y no supervisadas.
- Diferenciar las estructuras perceptrón simple y perceptrón multicapas.

## INTRODUCCIÓN

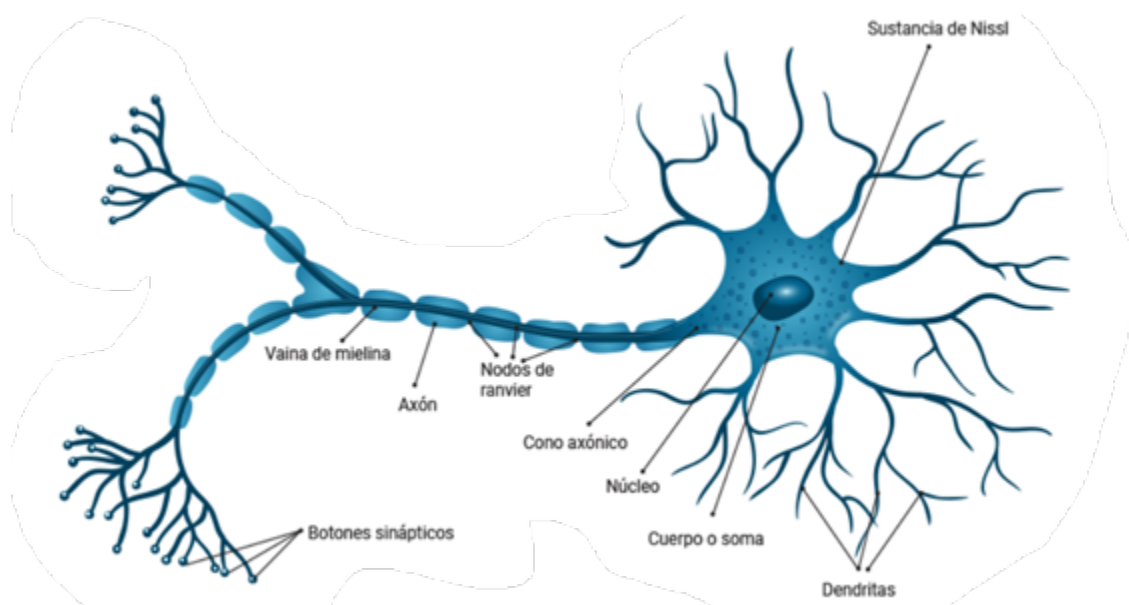
Las redes neuronales son un área muy importante de la inteligencia artificial, ya que están inspiradas en el conocido comportamiento del cerebro humano que incluye, principalmente, las neuronas, sus vínculos y conexiones, que estas se esfuerzan por crear modelos sintéticos que resuelvan problemas difíciles utilizando técnicas de ingeniería y algoritmos populares.

Asimismo, las redes neuronales surgen de la idea de imitar el funcionamiento de estas mismas en los organismos vivos, es decir, no son más que un conjunto de neuronas que se interconectan y trabajan juntas, sin funciones específicas para cada tipo. Al tomar la práctica, las neuronas hacen y fortalecen ciertas conexiones para aprender algo que permanecerá fijo en los tejidos.

## FUNDAMENTOS BIOLÓGICOS

“La teoría y modelado de las redes neuronales artificiales se inspira en la estructura y funcionamiento de los sistemas nerviosos, de los cuales las neuronas son fundamentales” [1].

Un sistema de procesamiento de información a gran escala que consta de unidades de procesamiento simples llamadas neuronas. Asimismo, en cierto modo, una neurona es un procesador de información muy simple, el cual se puede decir que está formada por un canal de entrada llamada dendritas, un procesador llamado soma que es donde se aloja el núcleo de la célula y finalmente por un canal de salida llamado axón.



**Figura 1.** Estructura general de una neurona [2]

Asimismo, una sola neurona del cerebro puede recibir unas 10 000 entradas y enviar su salida a cientos de otras neuronas a través de sus ramificaciones, ya que las conexiones entre las neuronas se llaman sinapsis. No es una conexión física, pero tiene un espacio de unos 2 mm., que son conexiones unidireccionales en las que la transmisión de información se produce eléctricamente dentro de las neuronas y químicamente entre neuronas. Gracias por una sustancia específica llamada neurotransmisor.

Las sinapsis entre las neuronas es clave para el procesamiento de la información. La mayoría de las neuronas tienen una estructura en forma de árbol llamada dendritas, que reciben información de otras neuronas a través de sinapsis. Las neuronas se componen de tres partes.

| Neurona   | Dendritas   | Axón   |
|---|---|--|
| <ul style="list-style-type: none"> <li>Es el cuerpo de la misma.</li> </ul> | <ul style="list-style-type: none"> <li>Reciben las entradas.</li> </ul> | <ul style="list-style-type: none"> <li>Es el encargado de llevar la salida de la neurona a las otras dendritas.</li> </ul> |

Figura 2. Parte de las neuronas

Sin embargo, hay que acotar que no todas las neuronas son iguales. Existen diferentes tipos, según el número de ramas dendríticas, la longitud del axón y otros detalles estructurales. Pero todas funcionan con los mismos principios básicos.

La forma perfecta de conectar dos neuronas no se conoce por completo. También depende del tipo particular de cada neurona. En general, las neuronas envían su axón a su salida hacia otros y este lleva información a través de las diferencias potencial.

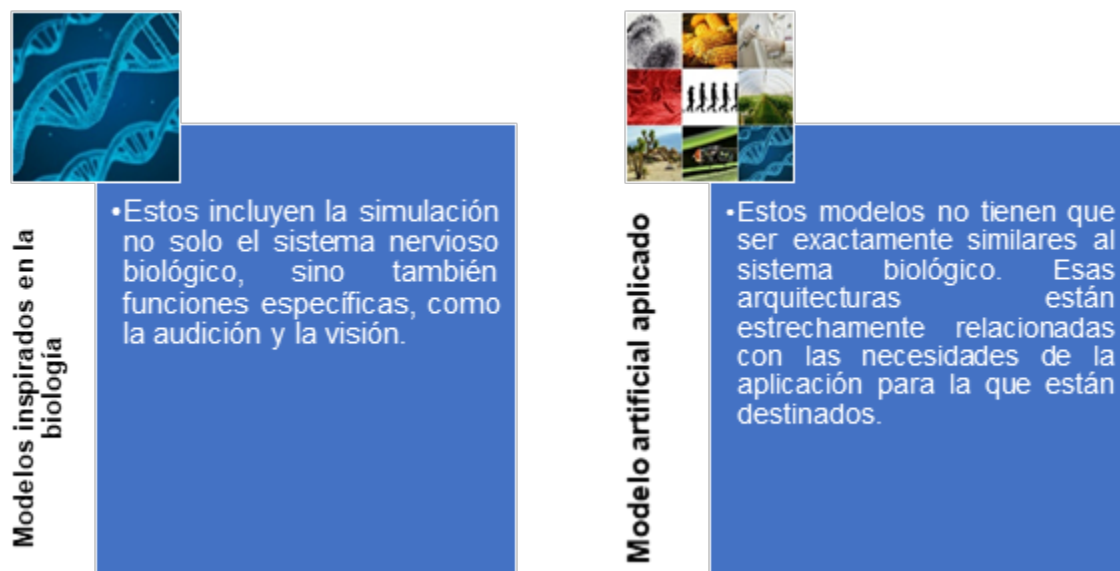


Figura 3. Clasificación de redes neuronales [5]

Se estima que el cerebro humano contiene más de 100 mil millones de neuronas y 10000,000000 sinapsis del sistema nervioso. Los estudios realizados en el campo de la anatomía del cerebro humano concluyen que el número promedio de sinapsis dentro y fuera de cada neurona es superior a 1000. Las neuronas biológicas tienen tiempos de conmutación casi un millón de veces más largos (milisegundos) que los componentes informáticos modernos. Sin embargo, las neuronas naturales tienen miles de veces más conectividad que las neuronas artificiales. Por tal motivo, el objetivo principal de las redes neuronales de tipo biológico es el desarrollo de operaciones, síntesis y procesamiento de información relacionada con los sistemas biológicos.

## MODELO COMPUTACIONAL

Las redes neuronales artificiales son unidades informáticas interconectadas a gran escala y adaptables neuronas o modelos informáticos paralelos que consisten en procesadores distribuidos a gran escala, que guarda el conocimiento experimental en paralelo el cual siempre está disponible para su uso. El primer modelo de red neuronal fue propuesto por McCulloch y Pitts en el año 1943. Desde la perspectiva de un modelo informático de actividad neuronal, este modelo era un modelo binario, donde cada neurona tiene una etapa o umbral dado y se utilizó como base para modelo posterior [3].

Las redes neuronales aplicadas generalmente se inspiran en las redes neuronales biológicas. Aunque tienen funciones y conexiones diferentes a las consideradas desde un punto de vista biológico.

Indicado lo anterior, se llegó a la primera clasificación de redes neuronales:

Sin embargo, en [5] se indica que las características principales de las redes neuronales juegan un papel importante durante el procesamiento de señales e imágenes. Estas utilizan una arquitectura que incluye elementos de procesamiento, interconexión adaptativa paralela combinada con una estructura jerárquica. Estas son las siguientes:

- **Autoorganización y adaptabilidad:** cuenta con algoritmos de aprendizaje adaptativo y autoorganización. De esta manera, permite brindar la mejor oportunidad para un tratamiento confiable.
- **Procesamiento no lineal:** desarrollar la capacidad de la red y aproxima las funciones, clasificando los modelos y mejorando la inmunidad para el ruido.
- **Procesamiento paralelo:** generalmente, esta característica utiliza muchos nodos de procesamiento, altamente interconectados.

El principal elemento computacional, es decir, modelo neuronal, suele ser conocido como un nodo o unidad. Este recibe información de otros dispositivos o fuentes de datos externas. Cada entrada tiene asociado un peso  $w$ , que se cambia en el llamado proceso siguiente de aprendizaje o aprender. Así, cada unidad aplica una función específica  $f$  a la suma de las entradas ponderadas al peso, es decir:

$$y_i = \sum_j w_{ij} y_j$$

Los resultados se pueden utilizar como salida para otras unidades.

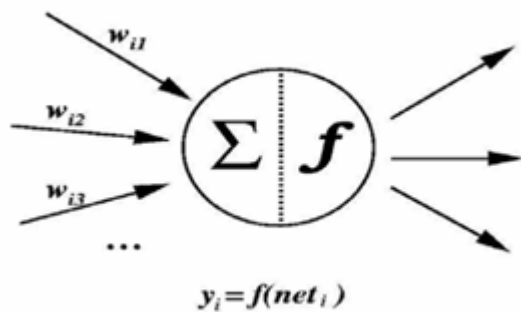


Figura 4. Resultado de salida [5]

El modelado con una red neuronal consta de dos pasos:



**Fase de entrenamiento:** se utiliza un conjunto de datos o modelos de entrenamiento para determinar los pesos, es decir, los parámetros, que definen el modelo de red neuronal. Este se calcula iterativamente en función del valor de entrenamiento para minimizar el error entre la salida producida por la red neuronal y la salida deseada.



**Fase de prueba:** en la fase anterior, el modelo puede haber estado demasiado cerca de las características que existen en el modelo de entrenamiento. Una de las características es generalizar el aprendizaje a nuevos casos (*overfitting*).

Figura 5. Pasos de redes neuronales

Se recomienda utilizar el segundo grupo para evitar problemas de sobreajuste de datos que no son de entrenamiento, grupos de validación que permiten esto controlan el proceso de aprendizaje.

Asimismo, en [1] se indica que hay tres funciones de transferencia típicas que determinan diferentes tipos de neuronas y son las siguientes:

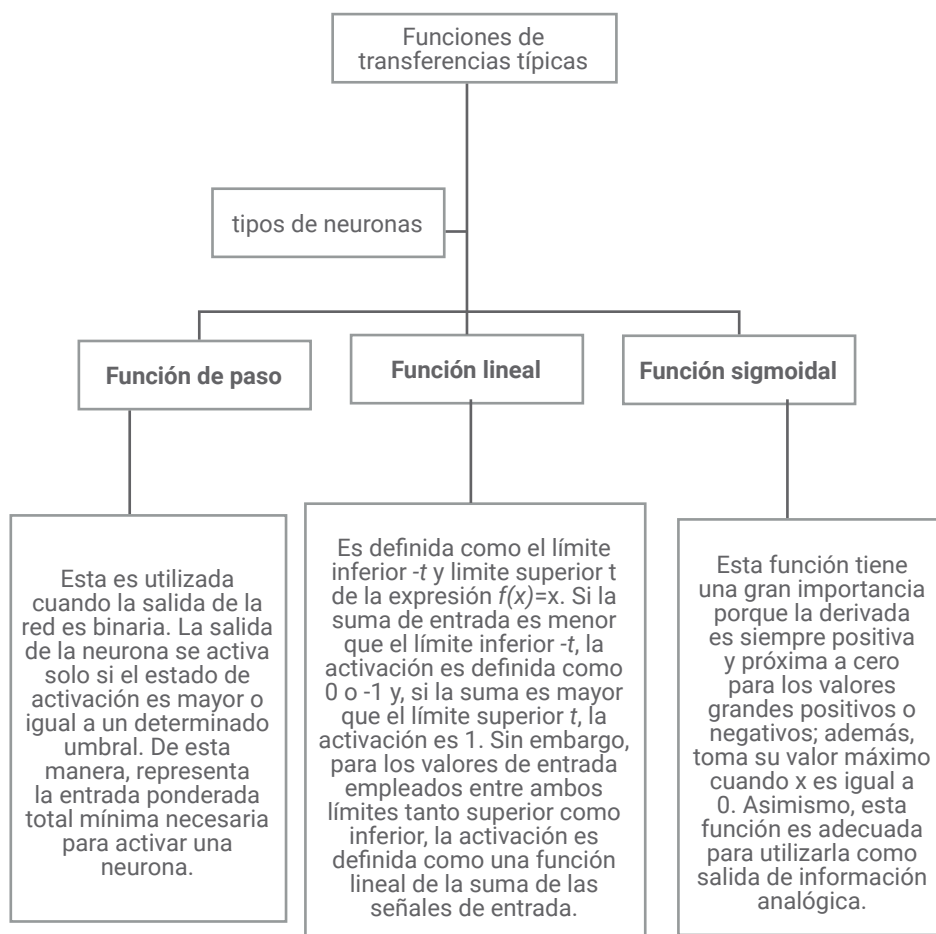


Figura 6. Tipos de neuronas [1]

Asimismo, la función sigmoidea es adecuada para utilizarla como salida de información analógica, ya que esta función, para los valores del estímulo de entrada (variable independiente), La importancia de esta función es que su derivada es siempre positiva y cercana a cero para los valores grandes positivos o negativos; además toma su valor máximo cuando  $x$  es 0. Esto hace que se puedan utilizar las reglas de aprendizaje en las cuales se usan derivadas. La expresión de esta función responde a la siguiente forma (figura 7):

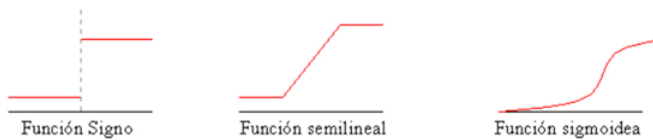


Figura 7. Estructura general del perceptrón [3]

Solamente se perfecciona el peso y los límites no se ajustarán. De esta manera, se facilita el aprendizaje, ya que es necesario tener en cuenta la presencia de una pseudoneurona conectada a la entrada de cada neurona, cuyo valor es el valor unitario en la salida  $y_o = -1$ , y peso de la conexión  $w_{oj} = t$ . Asimismo, la actualización de estado modular, es decir, evolución de la red neuronal, puede ser de dos tipos:

- **Modo asíncrono**, las neuronas están constantemente evaluando su estado, a medida que llega la información y de manera independiente.
- **Modo síncrono**, es un grupo de caras que definen la forma de la operación. No se retiene la historia de cómo se creó la operación síncrona..

Lo importante de todo esto es que en cada unidad se realiza un cómputo local a partir de los insumos que brindan los vecinos, pero no se requiere un control global de todas las unidades. En la práctica, la mayoría de las implementaciones de la Red Neuronal Artificial (ANN) en el *software* se implementan mediante el control síncrono para actualizar los módulos. Otro elemento muy importante de Red Neuronal Artificial (ANN) son las reglas de aprendizaje. Biológicamente, se acepta que la información almacenada en el cerebro está relacionada con el significado sináptico de las conexiones entre las neuronas. En otras palabras, el conocimiento está en las sinapsis.

En el caso de Red Neuronal Artificial (ANN), se puede suponer que el conocimiento se expresa en forma de pesos de conexiones entre neuronas. Cada proceso de aprendizaje implica algunos cambios en estas conexiones. De hecho, se puede decir que se aprendió cambiando el valor del peso de la red.

Hoy en día, existe un intercambio interesante entre el interaprendizaje de la neurociencia y la informática, lo que ha dado lugar a una nueva era que permiten a las máquinas imitar y aprender de los cerebros.

## REDES DE NEURONAS SUPERVISADAS Y NO SUPERVISADAS

No existe un algoritmo único para diseñar y entrenar modelos de redes neuronales porque los algoritmos de aprendizaje difieren en la capacidad de aprendizaje y el grado de razonamiento. Por tanto, las redes neuronales, generalmente, se clasifican de acuerdo con sus respectivos algoritmos o métodos de aprendizaje.

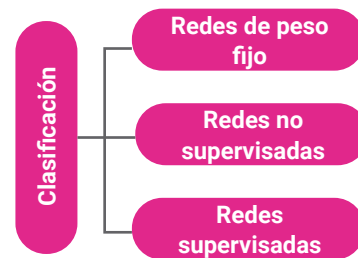


Figura 8. Clasificación de las redes neuronales.

Se debe tener en cuenta que las redes de pesos fijos no tienen algún tipo de aprendizaje. A continuación, se dará una breve explicación de las redes tanto supervisadas y no supervisadas.

### • Redes de neuronas supervisadas

“Las redes supervisado son el modelo de red más desarrollado desde el inicio de este tipo de diseño. Los datos de entrenamiento consisten en múltiples pares de modelos de entrenamiento de entrada y salida” [4].

El que se conozcan los resultados significa que el aprendizaje se realiza bajo la supervisión de un experto. Considerando el nuevo patrón de entrenamiento  $(m+1)$ , los pesos se ajustan de la siguiente manera:

$$w_{ij}^{(m+1)} = w_{ij}^{(m)} + \Delta w_{ij}^{(m)}$$

Posteriormente, se puede ver el diagrama de bloques del sistema de aprendizaje supervisado en la siguiente figura:

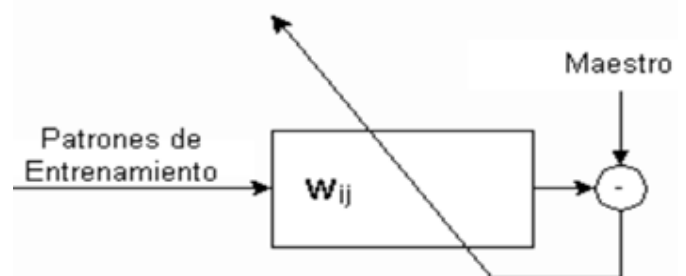


Figura 9. Diagrama esquemático de sistema de entrenamiento [4].

Por lo tanto, es necesario conocer tres características de los modelos supervisado:

- Una o más capas de neuronas ocultas que no forman parte de la capa de entrada o salida de la red. Esto permite que las redes aprendan y resuelvan problemas complejos.
- Las no linealidades reflejadas en la actividad neuronal son distinguibles.
- El modelo de interacción de red representa un alto grado de conectividad.

El algoritmo de aprendizaje de corrección de errores entrena la red basándose en muestras de entrada y salida para encontrar señales de error. Asimismo, el paradigma del aprendizaje supervisado es eficiente y encuentra soluciones a diversos problemas lineales y no lineales, como la clasificación, la gestión empresarial, la predicción, la predicción y la robótica.

#### • Redes de neuronas no supervisadas

“La red no supervisada utiliza datos de entrenamiento que consisten solo en modelos de entrada. Por lo tanto, esta red se forma sin la participación del maestro o experto” [4].

Asimismo, esta red aprende a adaptarse en función de la experiencia obtenida de patrones de aprendizaje anteriores.

Se presenta un diagrama típico de un sistema de aprendizaje no supervisado:

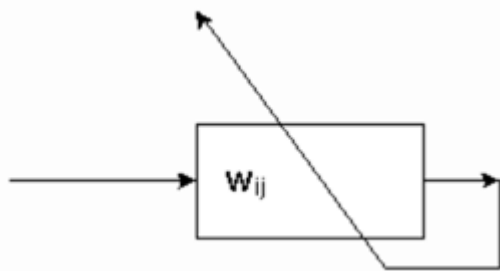


Figura 10. Diagrama esquemático de sistema de entrenamiento [4]

A continuación, se presentan las características de la red de neuronas no supervisadas

- Sin necesidad de maestros, profesores o especialistas, la topología, generalmente, se basa en un modelo de conectividad. Muestra cierto grado de autoorganización
- Las redes detectan de forma autónoma los datos de entrada:
  - Características
  - Patrones
  - Correlaciones
  - Categorías

- Toman menos tiempo para aprender que los generalmente supervisados.
- Arquitectura simple, generalmente:
  - Una sola capa
  - *Freed-Forward*
- Tipos principales:
  - Kohonen
  - Grossberg
- Solucionar los siguientes tipos de problemas:
  - Conocimiento
  - Análisis de componentes principales
  - Agrupación
  - Creación de prototipos, extracción de características, mapeo

Las neuronas tienen ese valor en la salida que se propaga por la red con conexiones unidireccionales a otras celdas de la red. Asociado a cada conexión, que es un peso sináptico,  $w_{ij}$  define los efectos de las neuronas  $j$ -ésima de la  $i$ -ésima neurona. La entrada de la  $i$ -ésima neurona de otras neuronas se acumula en el umbral externo para obtener el valor de la red. La mejor forma de determinarlo es continuar el conjunto de bases matemáticas  $f$  especificando el valor de activación. En este sentido, el resultado final puede expresarse en función de entradas y pesos.

La red conectada se representa matemáticamente mediante las funciones básicas  $u(w, x)$ , donde  $w$  es la matriz de pesos mientras que  $x$  es el vector de entrada. Sin embargo, en [4] se indica que las funciones base tienen dos formas típicas y son las siguientes:

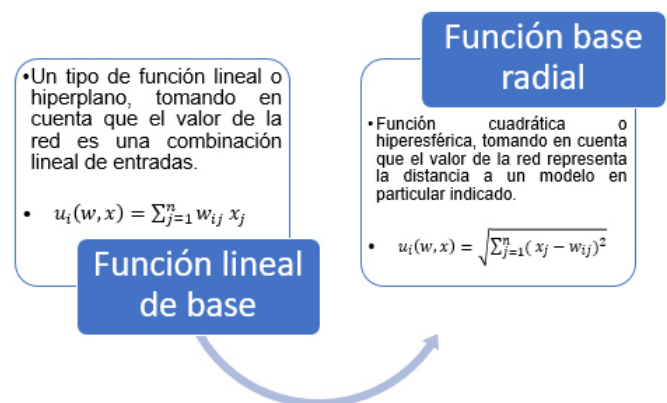


Figura 11. Formas típicas de las redes neuronales [4].



## PERCEPTRÓN SIMPLE

A partir de la idea del mecanismo biológico, y como anteriormente se comentó, McCulloch & Pitts establecieron un modelo artificial de neuronas en 1943. Esta idea fue adoptada por Rosenblatt quien, más tarde, creó una estructura llamada Perceptron. Un perceptrón es una estructura con una o más entradas y utiliza señales binarias de las neuronas de entrada y salida. Su función de activación es de tipo signo. Consta de una red con una capa de salida de  $n$  neuronas y otra capa de salida de  $m$  neuronas. Utiliza señales binarias de las neuronas de entrada y salida, y su función de activación es de tipo firmado [2].

Se utilizan varios tipos de técnicas de aprendizaje automático para que las redes neuronales aprendan. Sin embargo, el más común es el aprendizaje supervisado. En este caso, el algoritmo toma como entrada los datos previamente etiquetados. Es decir, se observa el aprendizaje y se dice al algoritmo cómo interpretar la información.

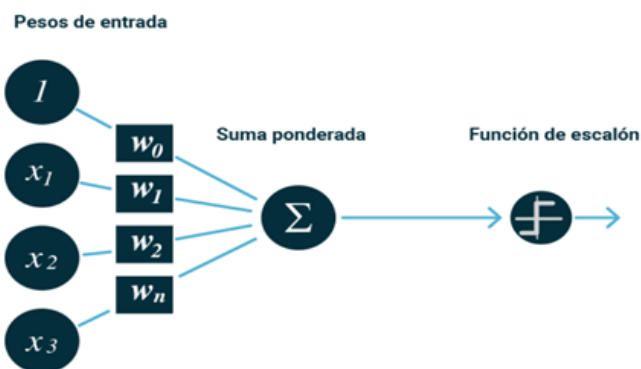


Figura 12. Estructura general del perceptrón [2]

La entrada al perceptrón actúa como una entrada a las dendritas o neuronas. La función de activación actúa como una célula somática que determina cuándo se activa una señal eléctrica y la salida modela un axón que, cuando se activa, envía información a las neuronas conectadas. Las redes neuronales se construyen agregando diferentes tipos de perceptrones en estructuras complejas.

Se debe tomar en cuenta que hay dos tipos de aprendizaje en el perceptrón:

- Uno usa la tasa de aprendizaje
- No usa la tasa de aprendizaje

Asimismo, el algoritmo de aprendizaje es el mismo para todas las neuronas. Sin embargo, la tasa de aprendizaje amortigua los cambios en los valores de peso.

Según [3], este modelo creado está constituido por los siguientes elementos:

- Conjunto de vectores de entrada  $x$  de  $n$  componentes
- Conjunto de pesos sinápticos  $w_{ij}$ . Esto simboliza la interacción entre la neurona presináptica  $j$  y la neurona postsináptica  $i$ .
- Regla de transferencia  $d(w_{ij}x_j(t))$ : para potencial postsináptico  $h_i(t)$
- Función de activación  $a_i(t) = f(a_i(t-1), h_i(t))$ : proporciona el estado de activación de la neurona según el valor pre y postsináptico.
- Función de salida  $f_i(t)$ : proporciona la salida  $y_i(t)$ , dependiendo del estado activo.

Las señales de entrada y salida pueden ser las siguientes:

- Señales binarias, es decir, 0,1 (neuronas de McCulloch y Pitts)
- Señales Bipolares, es decir, -1,1
- Números enteros o continuos
- Variables borrosas

Sin embargo, un perceptrón es un discriminador cúbico que utiliza la siguiente matriz

$$f(x) = \begin{cases} 1, & \text{si } w \cdot x - u > 0 \\ 0, & \text{en otro caso} \end{cases}$$

Para representar una red neuronal y, de esta manera, asignar una entrada  $x$  a un vector binario y un valor de salida  $f(x)$ , o valor binario, a través de una matriz específica.

Donde  $w$  es el vector de peso real y  $x$  es el producto escalar que calcula la suma ponderada.  $u$  es el umbral que representa el grado de supresión de neuronas y es un término constante que no depende del valor de entrada.

- El valor  $f(x)$  con valores (0,1) se utiliza para clasificar  $x$  como un caso positivo o negativo en el caso de problemas de clasificación binaria.
- Los umbrales pueden entenderse como un medio para compensar la función de activación o como un medio para establecer el nivel mínimo de actividad en el que una neurona se considera activa.

La suma ponderada de las entradas debe dar un valor mayor que  $u$  para que la neurona pase del estado 0 (negativo) al estado 1 (positivo).

En [6] se facilita un ejemplo de un perceptrón:

Considérese las funciones AND y OR, donde estas funciones son linealmente separables y pueden ser aprendidas por el perceptrón. Mientras que la función XOR no se puede entrenar con un solo perceptrón porque se necesitan al menos dos líneas para separar las clases (0,1), se debe usar al menos una capa adicional de perceptrón para formarlos.



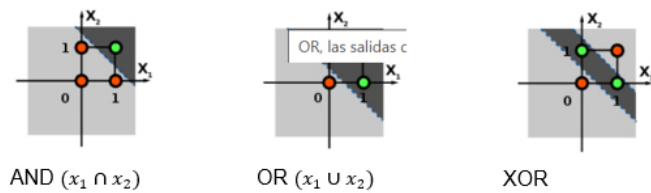


Figura 13. Funciones AND, OR, XOR [6]

Perceptrón aprende a realizar funciones NAND binarias utilizando las entradas  $x_1, x_2$ .

- Entrada:  $x_0, x_1, x_2$ , donde  $x_0$  permanece constante en el nivel 1.

- Umbral ( $t$ ): 0.5
- Bias ( $b$ ): 0
- Tasa de aprendizaje ( $r$ ): 0.1

Conjunto de entrenamiento que consta de las siguientes cuatro muestras:

$$\{(1,0,0),1\},\{(1,0,1),1\},\{(1,1,0),1\},\{(1,1,1),0\}$$

De ahora en adelante, el peso final de una iteración será el peso inicial de la próxima iteración. Cada ciclo de cada muestra de entrenamiento está marcado con una línea gruesa, tal como se presenta en la siguiente tabla:

| Entrada           |       |       |     | Pesos iniciales |       |       | Salida      |             |             |                   |                           | Error   | Corrección | Pesos finales     |                   |                   |
|-------------------|-------|-------|-----|-----------------|-------|-------|-------------|-------------|-------------|-------------------|---------------------------|---------|------------|-------------------|-------------------|-------------------|
| Valores de sensor |       |       |     | Salida deseada  |       |       | Sensor      |             |             | Suma              | Red                       |         |            |                   |                   |                   |
| $x_0$             | $x_1$ | $x_2$ | $z$ | $w_0$           | $w_1$ | $w_2$ | $c_0$       | $c_1$       | $c_2$       | $s$               | $n$                       | $e$     | $d$        | $w_0$             | $w_1$             | $w_2$             |
|                   |       |       |     |                 |       |       | $x_0 * w_0$ | $x_1 * w_1$ | $x_2 * w_2$ | $c_0 + c_1 + c_2$ | if $s > t$ then 1, else 0 | $z - n$ | $r * e$    | $\Delta(x_0 * d)$ | $\Delta(x_1 * d)$ | $\Delta(x_2 * d)$ |
| 1                 | 0     | 0     | 1   | 0               | 0     | 0     | 0           | 0           | 0           | 0                 | 0                         | 1       | +0.1       | 0.1               | 0                 | 0                 |
| 1                 | 0     | 1     | 1   | 0.1             | 0     | 0     | 0.1         | 0           | 0           | 0.1               | 0                         | 1       | +0.1       | 0.2               | 0                 | 0.1               |
| 1                 | 1     | 0     | 1   | 0.2             | 0     | 0.1   | 0.2         | 0           | 0           | 0.2               | 0                         | 1       | +0.1       | 0.3               | 0.1               | 0.1               |
| 1                 | 1     | 1     | 0   | 0.3             | 0.1   | 0.1   | 0.3         | 0.1         | 0.1         | 0.5               | 0                         | 0       | 0          | 0.3               | 0.1               | 0.1               |
| 1                 | 0     | 0     | 1   | 0.3             | 0.1   | 0.1   | 0.3         | 0           | 0           | 0.3               | 0                         | 1       | +0.1       | 0.4               | 0.1               | 0.1               |
| 1                 | 0     | 1     | 1   | 0.4             | 0.1   | 0.1   | 0.4         | 0           | 0.1         | 0.5               | 0                         | 1       | +0.1       | 0.5               | 0.1               | 0.2               |
| 1                 | 1     | 0     | 1   | 0.5             | 0.1   | 0.2   | 0.5         | 0.1         | 0           | 0.6               | 1                         | 0       | 0          | 0.5               | 0.1               | 0.2               |
| 1                 | 1     | 1     | 0   | 0.5             | 0.1   | 0.2   | 0.5         | 0.1         | 0.2         | 0.8               | 1                         | -1      | -0.1       | 0.4               | 0                 | 0.1               |
| 1                 | 0     | 0     | 1   | 0.4             | 0     | 0.1   | 0.4         | 0           | 0           | 0.4               | 0                         | 1       | +0.1       | 0.5               | 0                 | 0.1               |
| 1                 | 0     | 1     | 1   | 0.5             | 0     | 0.1   | 0.5         | 0           | 0.1         | 0.6               | 1                         | 0       | 0          | 0.5               | 0                 | 0.1               |
| 1                 | 1     | 0     | 1   | 0.5             | 0     | 0.1   | 0.5         | 0           | 0           | 0.5               | 0                         | 1       | +0.1       | 0.6               | 0.1               | 0.1               |
| 1                 | 1     | 1     | 0   | 0.6             | 0.1   | 0.1   | 0.6         | 0.1         | 0.1         | 0.8               | 1                         | -1      | -0.1       | 0.5               | 0                 | 0                 |
| 1                 | 0     | 0     | 1   | 0.5             | 0     | 0     | 0.5         | 0           | 0           | 0.5               | 0                         | 1       | +0.1       | 0.6               | 0                 | 0                 |
| 1                 | 0     | 1     | 1   | 0.6             | 0     | 0     | 0.6         | 0           | 0           | 0.6               | 1                         | 0       | 0          | 0.6               | 0                 | 0                 |
| 1                 | 1     | 0     | 1   | 0.6             | 0     | 0     | 0.6         | 0           | 0           | 0.6               | 1                         | 0       | 0          | 0.6               | 0                 | 0                 |
| 1                 | 1     | 1     | 0   | 0.6             | 0     | 0     | 0.6         | 0           | 0           | 0.6               | 1                         | -1      | -0.1       | 0.5               | -0.1              | -0.1              |
| 1                 | 0     | 0     | 1   | 0.5             | -0.1  | -0.1  | 0.5         | 0           | 0           | 0.5               | 0                         | 1       | +0.1       | 0.6               | -0.1              | -0.1              |
| 1                 | 0     | 1     | 1   | 0.6             | -0.1  | -0.1  | 0.6         | 0           | -0.1        | 0.5               | 0                         | 1       | +0.1       | 0.7               | -0.1              | 0                 |
| 1                 | 1     | 0     | 1   | 0.7             | -0.1  | 0     | 0.7         | -0.1        | 0           | 0.6               | 1                         | 0       | 0          | 0.7               | -0.1              | 0                 |
| 1                 | 1     | 1     | 0   | 0.7             | -0.1  | 0     | 0.7         | -0.1        | 0           | 0.6               | 1                         | -1      | -0.1       | 0.6               | -0.2              | -0.1              |
| 1                 | 0     | 0     | 1   | 0.6             | -0.2  | -0.1  | 0.6         | 0           | 0           | 0.6               | 1                         | 0       | 0          | 0.6               | -0.2              | -0.1              |
| 1                 | 0     | 1     | 1   | 0.6             | -0.2  | -0.1  | 0.6         | 0           | -0.1        | 0.5               | 0                         | 1       | +0.1       | 0.7               | -0.2              | 0                 |
| 1                 | 1     | 0     | 1   | 0.7             | -0.2  | 0     | 0.7         | -0.2        | 0           | 0.5               | 0                         | 1       | +0.1       | 0.8               | -0.1              | 0                 |
| 1                 | 1     | 1     | 0   | 0.8             | -0.1  | 0     | 0.8         | -0.1        | 0           | 0.7               | 1                         | -1      | -0.1       | 0.7               | -0.2              | -0.1              |
| 1                 | 0     | 0     | 1   | 0.7             | -0.2  | -0.1  | 0.7         | 0           | 0           | 0.7               | 1                         | 0       | 0          | 0.7               | -0.2              | -0.1              |
| 1                 | 0     | 1     | 1   | 0.7             | -0.2  | -0.1  | 0.7         | 0           | -0.1        | 0.6               | 1                         | 0       | 0          | 0.7               | -0.2              | -0.1              |
| 1                 | 1     | 0     | 1   | 0.7             | -0.2  | -0.1  | 0.7         | -0.2        | 0           | 0.5               | 0                         | 1       | +0.1       | 0.8               | -0.1              | -0.1              |
| 1                 | 1     | 1     | 0   | 0.8             | -0.1  | -0.1  | 0.8         | -0.1        | -0.1        | 0.6               | 1                         | -1      | -0.1       | 0.7               | -0.2              | -0.2              |
| 1                 | 0     | 0     | 1   | 0.7             | -0.2  | -0.2  | 0.7         | 0           | 0           | 0.7               | 1                         | 0       | 0          | 0.7               | -0.2              | -0.2              |
| 1                 | 0     | 1     | 1   | 0.7             | -0.2  | -0.2  | 0.7         | 0           | -0.2        | 0.5               | 0                         | 1       | +0.1       | 0.8               | -0.2              | -0.1              |
| 1                 | 1     | 0     | 1   | 0.8             | -0.2  | -0.1  | 0.8         | -0.2        | 0           | 0.6               | 1                         | 0       | 0          | 0.8               | -0.2              | -0.1              |
| 1                 | 1     | 1     | 0   | 0.8             | -0.2  | -0.1  | 0.8         | -0.2        | -0.1        | 0.5               | 0                         | 0       | 0          | 0.8               | -0.2              | -0.1              |
| 1                 | 0     | 0     | 1   | 0.8             | -0.2  | -0.1  | 0.8         | 0           | 0           | 0.8               | 1                         | 0       | 0          | 0.8               | -0.2              | -0.1              |
| 1                 | 0     | 1     | 1   | 0.8             | -0.2  | -0.1  | 0.8         | 0           | -0.1        | 0.7               | 1                         | 0       | 0          | 0.8               | -0.2              | -0.1              |

Tabla 1. Muestra de entrenamiento del perceptrón. [5]

## PERCEPTRÓN MULTICAPA

Ahora que se tiene información sobre las redes neuronales, se estudiarán los modelos de redes neuronales más importantes (antiguos y útiles). Este modelo es un perceptrón multicapa basado en un perceptrón simple que resuelve, de manera efectiva, problemas de clasificación y reconocimiento de patrones.

“Es una red de retropropagación que contiene al menos una capa oculta con suficientes unidades no lineales para aproximar una función o relación continua entre un conjunto de variables de entrada y salida. Esta propiedad hace que la red Perceptron Multicapa sea una herramienta no lineal universal y flexible” [5].

La estructura consta de tres capas, como se muestra en la (figura 14), el cual tiene una capa de entrada ( $L_1$ ), que consta de neuronas que introducen patrones de entrada en la red. No se realiza ningún procesamiento en estas neuronas, seguidamente conta de una capa intermedia u oculta ( $L_2$ ), que está desarrollada por neuronas donde las entradas provienen de la capa anterior y las salidas se envían a las neuronas de las capas posteriores. Finalmente, de una capa de salida ( $L_3$ ), que son neuronas cuyo valor de salida corresponde a la salida de toda la red.

Aunque la mayoría de los problemas han demostrado que las capas ocultas son suficientes, aquí se observar un perceptrón típico que consta de una capa de entrada, una capa oculta y una capa de salida.

Sin embargo, esta red cuenta con la característica de ser necesario estimar la función de transferencia del elemento de procesamiento (neurona).

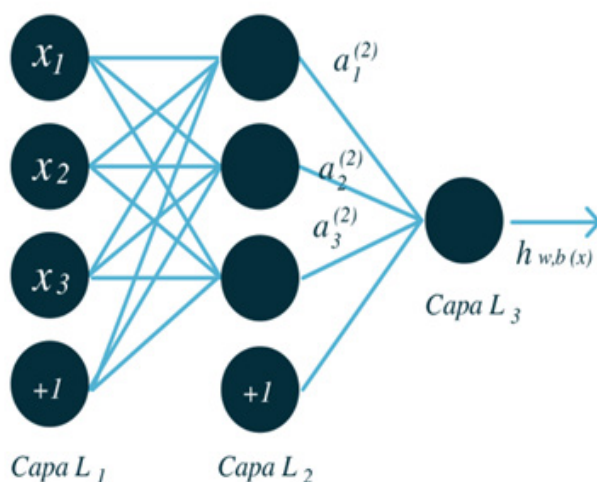


Figura 14. Red Neuronal Artificial (ANN) [2]

Las conexiones entre las neuronas son siempre directas: las conexiones son de las neuronas de esta capa a las neuronas de la capa siguiente. Sin embargo, no hay conexión lateral o trasera. Por lo tanto, la información siempre se transfiere de la capa de entrada a la capa de salida. Al considerar  $w_{ji}$ , se usa como el peso de la conexión entre la neurona de entrada  $i$  y la neurona intermedia  $j$  y  $v_{kj}$  como el peso de la conexión entre la  $j$  intermedia y la neurona de salida  $k$ .

En [7] se indica que los perceptrones multicapa se utilizan para resolver problemas como la asociación de imágenes, la segmentación de imágenes y la compresión de datos. Adicionalmente, se encuentran los siguientes:

- **Aprendizaje a través de la retropropagación**

El aprendizaje se realiza en el perceptrón, cambiando los pesos de conexión después de procesar cada elemento, según la cantidad de errores de salida en comparación con el resultado esperado. Este es un ejemplo de aprendizaje supervisado realizado por retro-propagación, que es una generalización del algoritmo LMS (raíz cuadrática media) en un perceptrón lineal.

Representa el error en el nodo  $j$  en el punto de datos  $n$  como el siguiente:

$$e_j(n) = d_j(n) - y_j(n)$$

Este es el valor objetivo  $y$ , el valor generado por el perceptrón. Al ajustar los pesos de los nodos en función de los ajustes que minimizan los errores en el resultado general dado por la expresión:

$$\epsilon(n) = \frac{1}{2} \sum_j e_j^2(n)$$

Usando el método de descenso más pronunciado, los cambios en cada peso son los siguientes:

$$\Delta w_{ji}(n) = -\eta \frac{\partial \epsilon(n)}{\partial v_j(n)} y_i(n)$$

Donde  $y_i$  es la salida de la neurona anterior y  $\eta$  es la tasa de aprendizaje. Ha sido cuidadosamente seleccionado para que los pesos converjan a la respuesta lo suficientemente rápido sin causar vibraciones. En la programación de aplicaciones, este parámetro suele estar entre 0,2 y 0,8.

La derivada calculada depende del campo de inducción local  $v_j$  que evoluciona por sí mismo. Para el nodo de salida, es fácil probar que esta derivada se puede simplificar de la siguiente manera:

$$\frac{\partial \epsilon(n)}{\partial v_j(n)} = e_j(n) \phi'(v_j(n))$$

Donde  $x$  es la derivada de la función de activación anterior y no cambia. El análisis para cambiar pesos en nodos ocultos es más complicado, pero la derivada correspondiente se puede mostrar de la siguiente manera:

$$\frac{\partial \epsilon(n)}{\partial v_j(n)} = \phi'(v_j(n)) \sum_k - \frac{\partial \epsilon(n)}{\partial v_k(n)} w_{ji}(n)$$

Esto depende de cambiar el peso del nodo  $k$  que representa la capa de salida. Por lo tanto, para cambiar el peso de la capa oculta, primero, se debe cambiar el peso de la capa de salida según la derivada de la función de activación. En segundo lugar, este algoritmo es la retro-propagación de la función de activación.

- **Reconocimiento de numérico**

La biblioteca sklearn de Python se puede usar para entrenar un perceptrón multicapa con el fin de que reconozca los números presentes en la imagen de entrada. Puede usar la clase MLPClassifier y el conjunto de datos MNIST.3.

## RESUMEN

Las redes neuronales son un área muy importante de la inteligencia artificial. La teoría y modelado de las redes neuronales artificiales se inspiran en la estructura y funcionamiento de los sistemas nervioso. Sin embargo, hay que acotar que no todas las neuronas son iguales. Existen diferentes tipos, según el número de ramas dendríticas, la longitud del axón y otros detalles estructurales, pero todas funcionan con los mismos principios básicos.

Las redes neuronales artificiales son unidades informáticas interconectadas a gran escala y adaptables neuronas o modelos informáticos paralelos que consisten en procesadores distribuidos a gran escala, que guarda el conocimiento experimental en paralelo el cual siempre está disponible para su uso.

Asimismo, no existe un algoritmo único para diseñar y entrenar modelos de redes neuronales porque los algoritmos de aprendizaje difieren en la capacidad de aprendizaje y el grado de razonamiento. Por tanto, las redes neuronales, generalmente, se clasifican de acuerdo con sus respectivos algoritmos o métodos de aprendizaje.

A partir de la idea del mecanismo biológico, establecieron un modelo artificial de neuronas en 1943. Esta idea fue adoptada por Rosenblatt quien, más tarde, creó una estructura llamada Perceptrón.

## BIBLIOGRAFÍA

- [1] M. C. Soberanis, "Inspiración biológica de las redes neuronales artificiales", 2018. [En línea]. Disponible en: <https://medium.com/soldai/inspiraci%C3%B3n-biol%C3%B3gica-de-las-redes-neuronales-artificiales-9af7d7b906a>
- [2] L. Alonso, "Redes Neuronales". [En línea]. Disponible en: <http://avellano.fis.usal.es/~lalonso/RNA/index.htm>
- [3] Monografía, enciclopedia libre, "Redes Neuronales". [En línea]. Disponible en: <https://www.monografias.com/trabajos38/redes-neuronales/redes-neuronales>
- [4] J. M. Marín, "Introducción a las redes neuronales aplicadas", 2009. [En línea]. Disponible en: <http://halweb.uc3m.es/esp/Personal/personas/jmmarin/esp/DM/tema3dm.pdf>
- [5] Wikipedia, enciclopedia libre, "Perceptrón", 2021. [En línea]. Disponible en: <https://es.wikipedia.org/wiki/Perceptr%C3%B3n>
- [6] Wikipedia, enciclopedia libre, "Perceptrón Multicapa", 2022. [En línea]. Disponible en: [https://es.wikipedia.org/wiki/Perceptr%C3%B3n\\_multicapa](https://es.wikipedia.org/wiki/Perceptr%C3%B3n_multicapa)