



FASTBOOK 01

# **Conceptos principales y arquitecturas básicas: el modelo feed forward network**

Introducción al Deep Learning



# 01. Conceptos principales y arquitecturas básicas: el modelo feed forward network

Hasta ahora hemos visto el funcionamiento de diferentes algoritmos de machine learning, pues bien, en este fastbook, daremos un paso más y descubriremos de qué trata el deep learning. Será nuestra primera aproximación a este subconjunto del machine learning, veremos cómo se diferencia de este, el tipo de datos que emplea y algunos ejemplos y casos de uso.

Además, nos adentraremos en el primer modelo de deep learning, el feed forward network o redes de alimentación hacia delante. Estas redes son las más básicas y servirán para entender y asentar el funcionamiento de las redes neuronales y diversos procesos asociados a estas.

Por lo tanto, los objetivos de aprendizaje marcados en este tema son los siguientes:

- Que entendamos qué es el deep learning, así como las diferentes clases de redes neuronales que hay para procesar diferentes tipos de datos.
- Analizar cómo este tipo de redes puede aprender este tipo de redes y cuáles son las diferentes funciones o métodos que operan internamente en estas.
- Conocer qué tipo de datos se puede introducir en una red neuronal en base a los diferentes casos de uso y algunos ejemplos actuales.
- Qué son las redes neuronales de alimentación hacia delante (feed forward networks), cuál es la arquitectura de estas redes, sus funciones y los procesos que las comprenden.

Y al final de esta unidad, podremos decir que ya estamos capacitados para construir un modelo basado en deep learning, para su entrenamiento y posterior evaluación.

Por último, solo he de añadir que la idea de este fastbook es que tuviera un toque muy pragmático, priorizando el entendimiento general de las técnicas y los procesos para facilitar la posterior aplicación práctica. Sin embargo, si deseas profundizar en los aspectos más teóricos o que han quedado fuera del marco de este texto, te recomiendo las siguientes lecturas:

- Skansi, S. (2018). *Introduction to Deep Learning: From Logical Calculus to Artificial Intelligence*. Springer Publishing.
- Aggarwal, C. (2018). *Neural Networks and Deep Learning: A Textbook*. Springer Publishing.

*Autor: Jorge Sánchez Casanova*

- Deep learning
- Tipos de datos en DL
- Estructura básica de una red neuronal
- Arquitecturas y modelos
- Medición desempeño de un algoritmo de DL
- Casos de usos y limitaciones
- Definición de redes de alimentación hacia adelante
- Funcionamiento y aprendizaje del modelo
- Conclusiones

# Deep learning



Antes de entrar en materia, es necesario entender y diferenciar tres conceptos: inteligencia artificial, machine learning y deep learning.



La **inteligencia artificial (IA)** se puede definir como la capacidad de una máquina para realizar funciones cognitivas asociadas al cerebro humano, como el aprendizaje, la percepción, el razonamiento o la interacción con el entorno.

Sin embargo, como se muestra en la siguiente imagen, tanto el **aprendizaje automático o machine learning (ML)** como el **aprendizaje profundo o deep learning (DL)** son subcampos dentro de la IA.

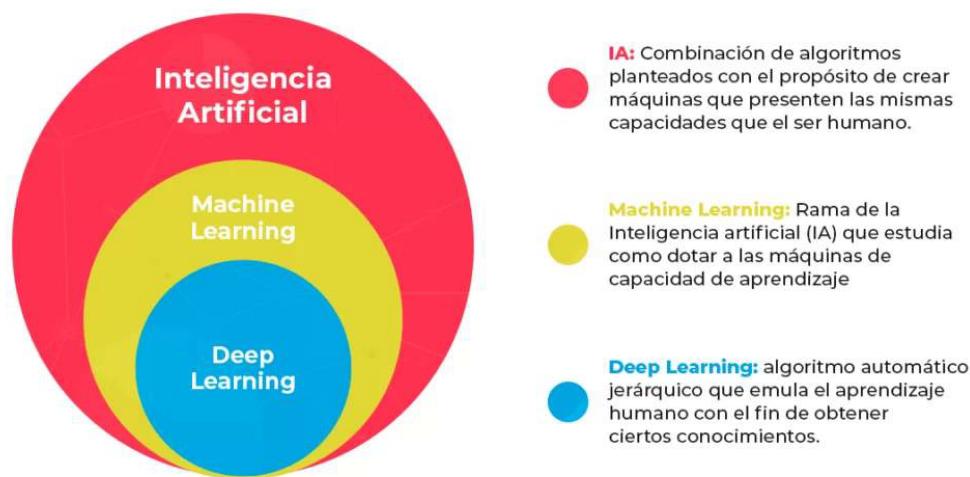


Figura 1. Clasificación de la IA, ML y DL.

Fuente: <https://www.xeridia.com/blog/redes-neuronales-artificiales-que-son-y-como-se-entrenan-parte-i>.

- En cuanto al **ML**, podemos definirlo como la **capacidad de la máquina para aprender por sí** misma utilizando un conjunto de datos, modificando y ajustando los parámetros a medida que procesa la información y aprende de ella.
- El **deep learning** es un **conjunto de algoritmos de ML**, basado en un tipo de modelo concreto (como son las redes neuronales) compuesto por varias capas, a través de las cuales se extrae información de alto nivel para tomar finalmente una decisión. El **deep learning es una tecnología que está en auge**, prueba de ello es la gran cantidad de aplicaciones de las que disponemos y sus avances (casi) diarios.

Para entender previamente su potencial, vamos a ver un par de ejemplos sobre las tareas complejas que hoy en día realizan los algoritmos basados en el aprendizaje profundo.

- El **rescalado y el recoloreado de imágenes** mediante el uso de deep learning representa una innovadora aplicación de esta tecnología en el ámbito del procesamiento de imágenes. Estas técnicas permiten modificar la resolución y los colores de una imagen de manera inteligente, preservando los detalles y las estructuras visuales.

---

**El rescalado implica ajustar las dimensiones de la imagen sin perder calidad, mientras que el recoloreado se refiere a convertir una imagen en blanco y negro a color.**

---

Utilizando técnicas de deep learning, especialmente arquitecturas como **las redes generativas adversarias (GAN)**, estas tareas pueden llevarse a cabo de manera más eficiente y realista. Este enfoque ha encontrado aplicaciones en la restauración de fotografías antiguas, la mejora de la calidad visual en sistemas de visualización y la generación de contenido visualmente atractivo en diversas industrias. Podemos ver un ejemplo de rescalado de imágenes y otro de recoloreado en las siguientes figuras.

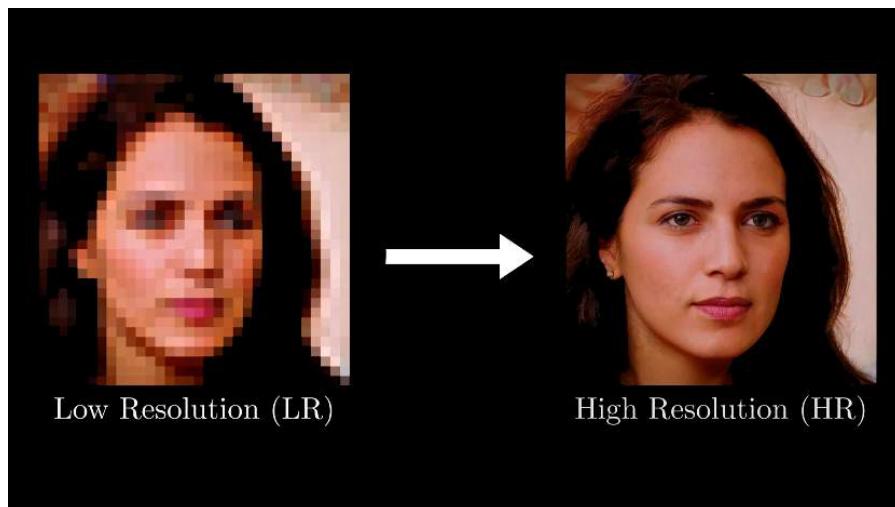


Figura 2. Rescalado de imágenes basado en IA.  
Fuente: <https://www.redsharknews.com/this-new-ai-system-could-be-science-fiction-but-its-very-real-indeed>.

---

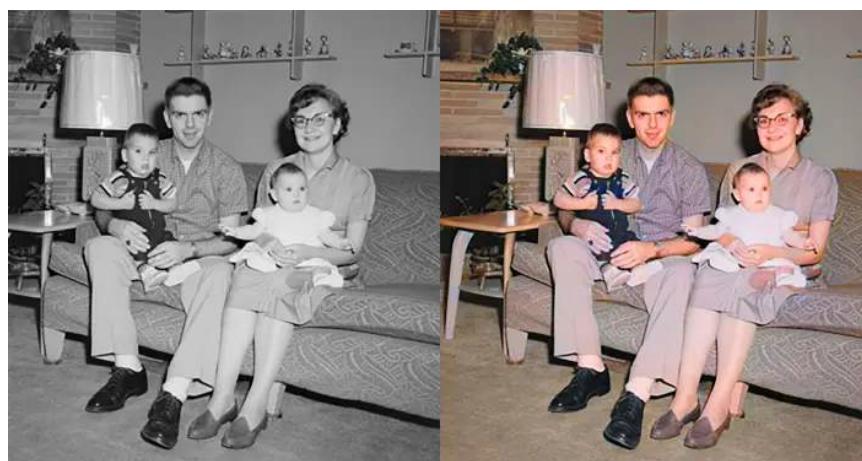


Figura 3. Recoloreado de una imagen en blanco y negro empleando IA.  
Fuente: <https://vocalremover.org/splitter-ai>.

---

La **separación de pistas de una canción** mediante el uso de deep learning representa otra fascinante aplicación en el campo de procesamiento de audio. Esta técnica permite descomponer una grabación musical en sus componentes individuales, aislando las distintas pistas como voces, instrumentos y percusiones.

---

**A través de algoritmos avanzados, particularmente las redes neuronales profundas como las redes neuronales convolucionales (CNN) y los modelos basados en Recurrent Neural Networks (RNN), esta tarea se realiza de manera más precisa y efectiva.**

---

La separación de pistas ha encontrado aplicaciones en la producción musical, la remasterización de grabaciones antiguas y la creación de versiones instrumentales, ofreciendo a los productores y músicos una **herramienta valiosa** para explorar y manipular los elementos sonoros de una composición.

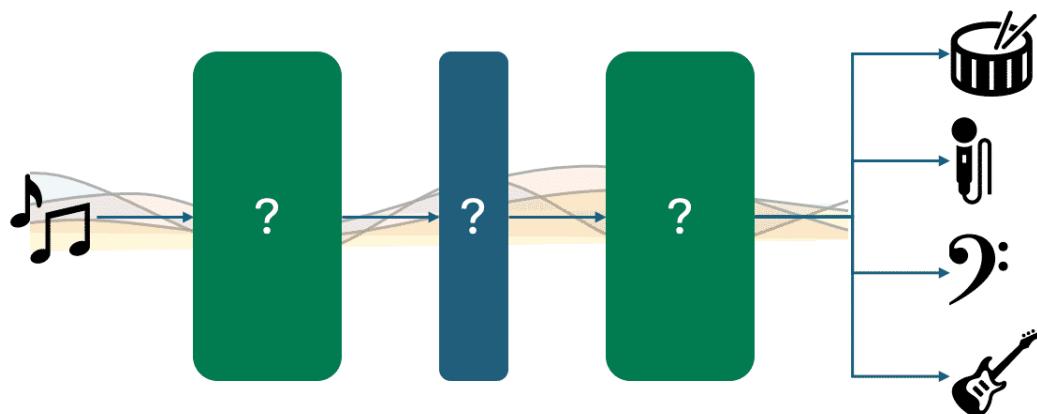


Figura 4. Separación de las pistas en una canción.

Fuente: <https://vanceai.com/colorize-photo/>.

---

**Estos ejemplos reflejan una característica muy importante acerca de los casos de uso donde deep learning se ha aplicado con más éxito: mediante el uso de datos no estructurados. En el siguiente apartado, vamos a entrar en detalle sobre la tipología de datos más común en retos resueltos mediante el aprendizaje profundo.**

# Tipos de datos en DL



El deep learning puede resolver prácticamente cualquier problema, incluyendo la clasificación de datos, su agrupación o la realización de predicciones sobre ellos. En cuanto a los tipos de datos para resolver estos problemas, el deep learning se aplica mejor a datos no estructurados, como son las imágenes, el vídeo, el sonido o el texto.

Una imagen es simplemente un conjunto de píxeles, un mensaje es tan solo un conjunto de texto.

No obstante, estos datos no están organizados en una **base de datos relacional típica por filas y columnas**, lo que dificulta especificar sus características manualmente.

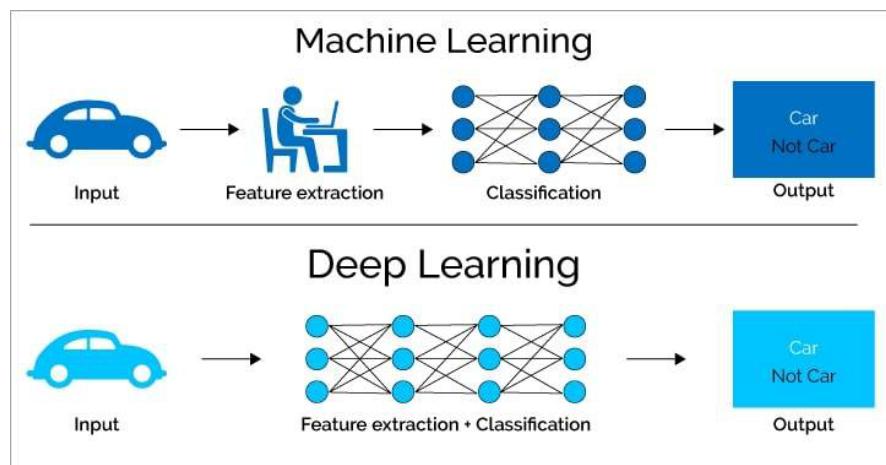


Figura 5. Diferencia entre los datos de entrada en machine learning y deep learning.

Fuente: <https://www.zerotosingularity.com/deep-learning-day/ai-deep-learning-primer/>.

A la hora de pensar en los datos para emplear algoritmos de deep learning, hay que tener en cuenta los siguientes puntos:

### **La relevancia de los datos**

Es importante utilizar datos de entrenamiento que sean directamente relevantes para el problema a resolver. Recuerda que las redes neuronales aprenden solo de los datos con los que se entranan.

### **La clasificación o regresión con conjuntos etiquetados**

Para construir soluciones de aprendizaje profundo que clasifiquen o predigan datos, se requiere un conjunto de datos etiquetado. Etiquetar datos como, por ejemplo, indicar si una imagen es una flor o un panda, es crucial.

### **El formato y preprocesamiento**

Los datos deben presentarse en forma de vectores con la misma longitud al ingresar a la red neuronal. Por ejemplo, en un caso donde se trabaje con imágenes, es necesario ajustar el tamaño de las imágenes mediante el preprocesamiento de datos.

## **Los requisitos mínimos de datos**

---

Aquí tenemos el caso de cuanto más, mejor. Empezar un problema con una complejidad media con 100.000 instancias en total (entre los 3 subconjuntos) es recomendable. Sin embargo, estos subconjuntos han de estar equilibrados, es decir, si vamos a realizar una clasificación de perro vs. gato, lo normal sería tener una distribución de los datos de aproximadamente el 50% de datos por categoría.

---

**¡Y no lo olvides! El cumplimiento de estos aspectos, desde la relevancia de los datos hasta la adecuada accesibilidad y manipulación, es esencial para el éxito en el desarrollo de soluciones basadas en aprendizaje profundo.**

# Estructura básica de una red neuronal



---

Las características del deep learning incluyen algoritmos que **analizan datos con una estructura lógica similar a la de un ser humano**. Para este propósito, los algoritmos de deep learning se basan en estructuras llamadas **redes neuronales artificiales** (ANN, por sus siglas en inglés).

---

**Estos algoritmos imitan el mecanismo de aprendizaje de los organismos biológicos utilizando neuronas artificiales.**

---

Las neuronas naturales están conectadas entre sí por sinapsis; sin embargo, las artificiales están conectadas a través de pesos, que cumplen el mismo propósito. La siguiente imagen muestra un ejemplo de una neurona artificial.

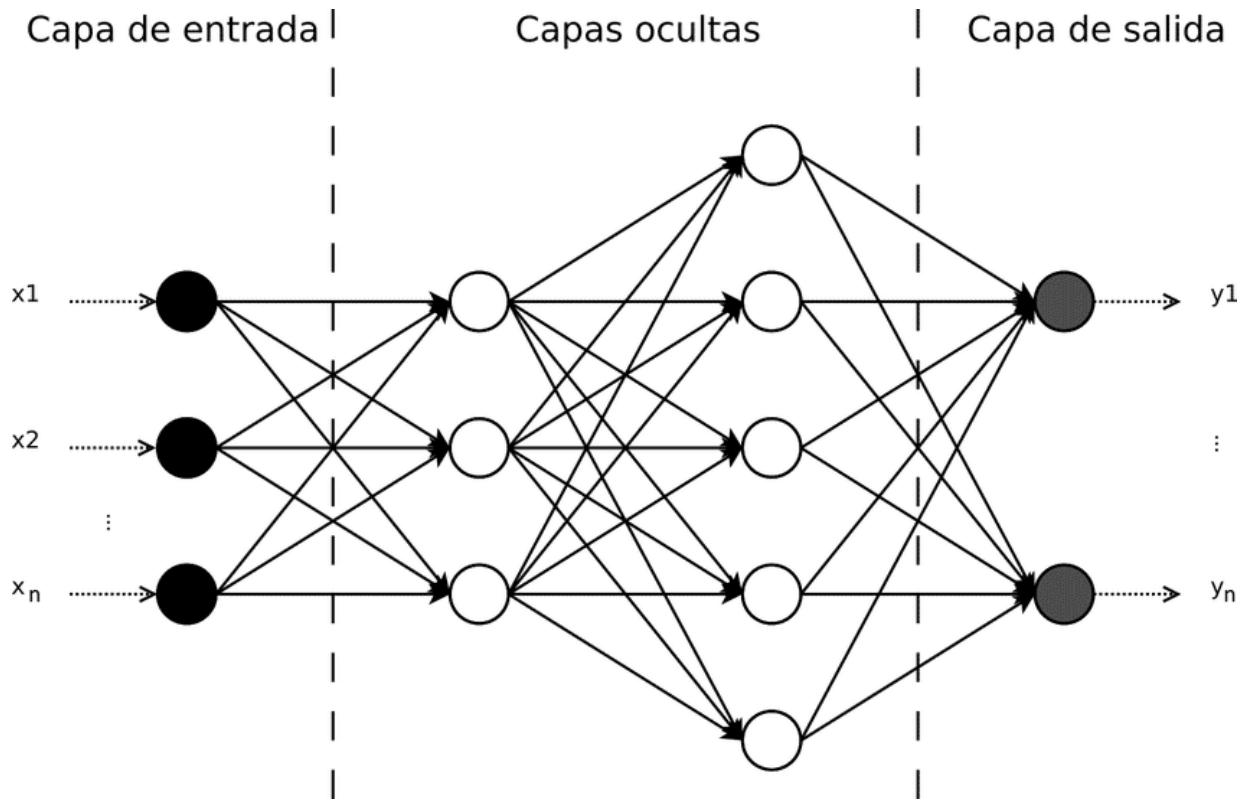


Figura 6. Neurona Artificial.

Fuente: W. De Mulder, S. Bethard, and M. F. Moens, *A survey on the application of recurrent neural networks to statistical language modeling*, Computer Speech and Language, vol. 30, no. 1, pp. 61–98, 2015. doi:10.1016/j.csl.2014.09.005.

---

**Aunque la unidad mínima utilizada por las redes neuronales artificiales (ANN) es la neurona, estas redes se forman utilizando capas, que no son otra cosa que conjuntos de neuronas. Cuantas más neuronas, mayor es la capa.**

A pesar de que las neuronas funcionan de manera independiente, se puede controlar su funcionamiento mediante la función de activación, la cual determina la salida de una neurona y el bias o sesgo. Es la función de activación la que determina si la neurona debe activarse (enviar una señal) o no, con respecto a la entrada que recibe, y en qué medida.

Por otro lado, **el bias o sesgo** es un **valor constante** que se suma a la combinación lineal de las entradas ponderadas por los pesos. Introduce flexibilidad en la capacidad de la red para aprender y modelar patrones más complejos y no lineales.

---

**Existen diferentes funciones de activación y la elección  
de la función de activación puede tener un impacto  
significativo en el rendimiento y la capacidad de  
aprendizaje de la red neuronal.**

---

Estas serían algunas de las funciones de activación comunes y sus características, ¡toma nota!

### **Función sigmoide**

---

- Rango de salida entre 0 y 1.
- Usada comúnmente en la capa de salida de las redes neuronales para los problemas de clasificación binaria.

## Función tangente hiperbólica (tanh)

- Rango de salida entre -1 y 1.
- Similar a la función sigmoide, pero con un rango extendido.

## Rectified Linear Unit (ReLU)

- Salida igual a cero para entradas negativas y lineal para entradas positivas.
- Ampliamente utilizada en las capas ocultas debido a su simplicidad y eficacia.

## Leaky ReLU

- Similar a ReLU, pero permite una pequeña pendiente para entradas negativas.
- Aborda el problema de las neuronas muertas en ReLU, donde las neuronas con entradas negativas siempre producen cero.

## Función de activación softmax

- Utilizada en la capa de salida para problemas de clasificación multiclas.
- Convierte las salidas en probabilidades, asegurando que la suma de las salidas sea igual a 1.

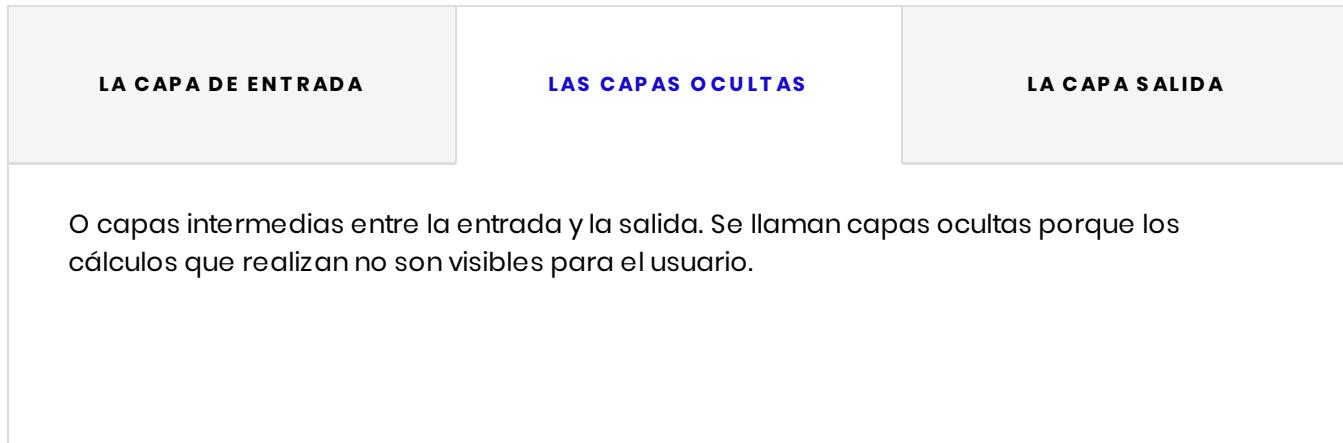
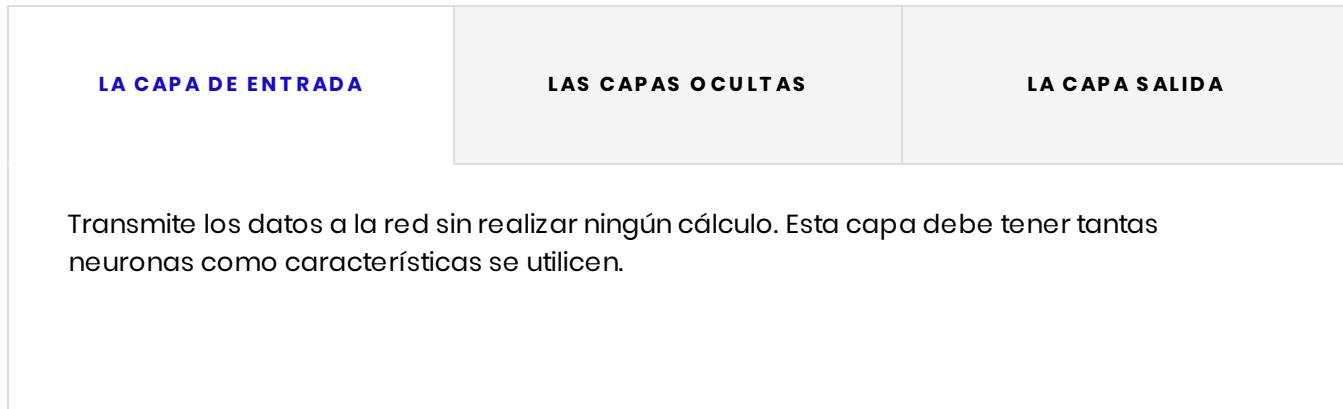
La elección de la función de activación depende del tipo de problema que se esté abordando y puede influir en la capacidad de la red para aprender patrones en los datos.

---

**Experimentar con diferentes funciones de activación es común durante el proceso de diseño y ajuste de una red neuronal.**

---

Como ya se ha dicho, las neuronas se agrupan en capas, sin embargo, para crear una red neuronal necesitamos **diferentes capas que se pueden agrupar en tres categorías:**



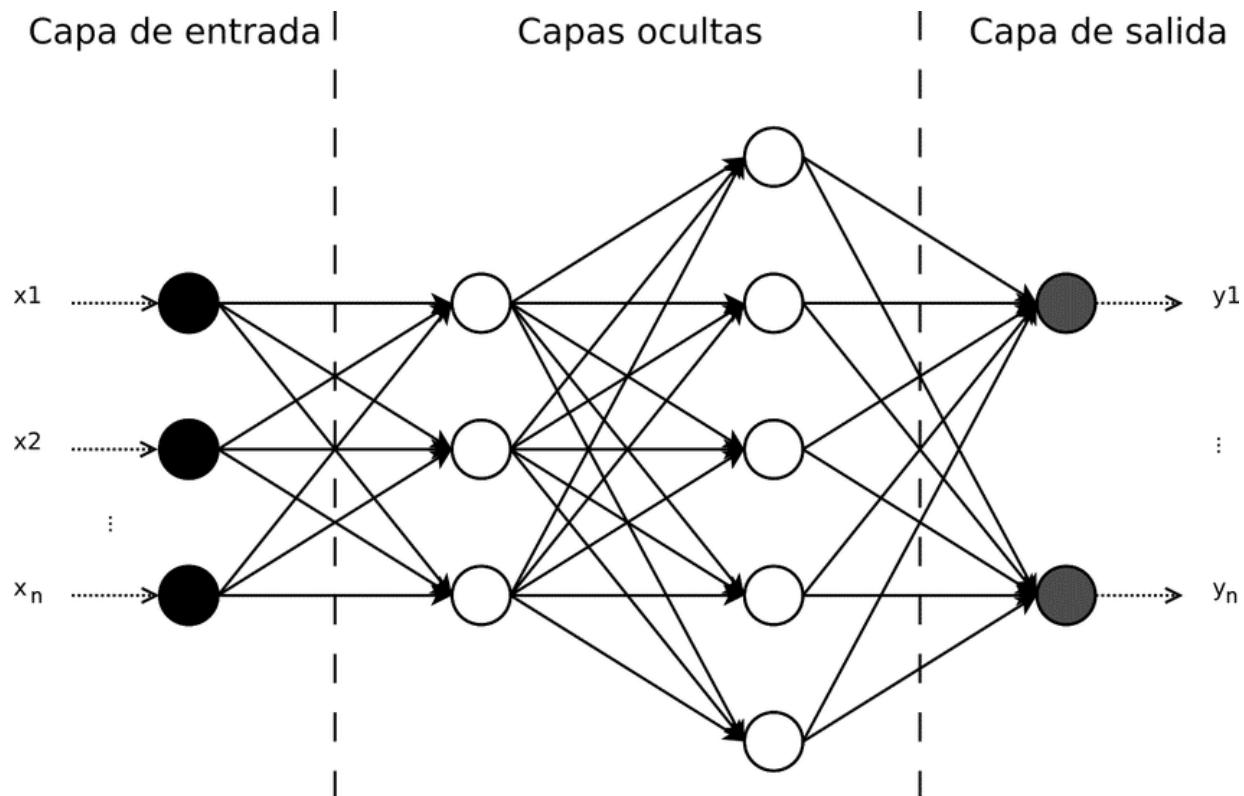
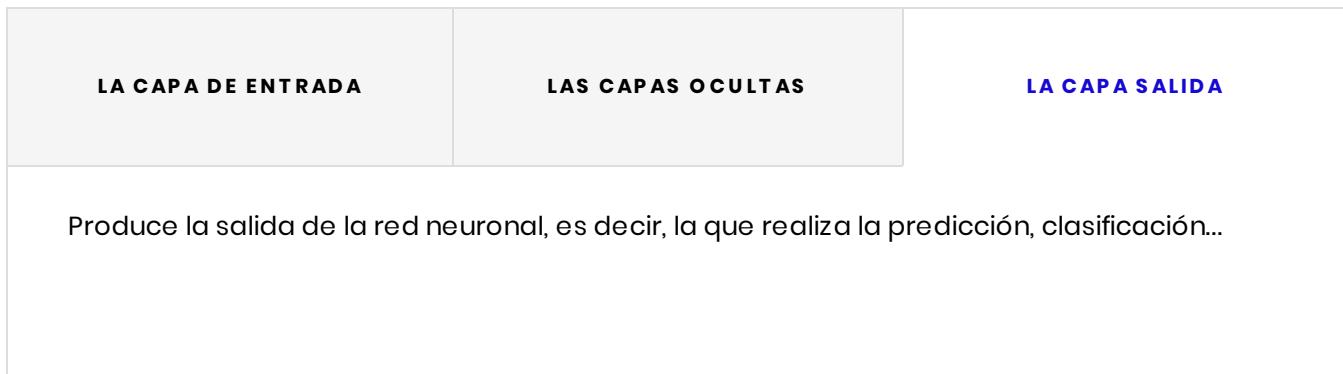


Figura 7. Capas en una red neuronal.

Fuente: [https://www.researchgate.net/figure/Red-neuronal-artificial-de-cuatro-capas\\_fig1\\_323985249](https://www.researchgate.net/figure/Red-neuronal-artificial-de-cuatro-capas_fig1_323985249).

Cuanto más profunda sea la red, es decir, cuantas más capas tenga, más información se puede inferir de los datos. Por ejemplo, una red superficial puede detectar líneas y, a medida que aumenta la profundidad, puede llegar a distinguir formas o siluetas. Una ANN estima la función de salida propagando los valores de las neuronas de entrada a las neuronas de salida y utilizando los pesos como parámetros intermedios.

---

**El aprendizaje se logra modificando los pesos que conectan las neuronas en un proceso conocido como entrenamiento.**

---

Las redes neuronales multicapa generalmente **se entrena utilizando un algoritmo de backpropagation o propagación hacia atrás de los errores**, que consta de dos fases principales: la fase hacia adelante y la fase hacia atrás.

1

**Fase hacia adelante:** en esta fase, se alimenta a la red neuronal con los datos de entrenamiento, lo que desencadena una cascada de cálculos a través de las capas utilizando los pesos actuales. El valor predicho se compara con la instancia de entrenamiento, y se calcula la derivada de la pérdida con respecto a la salida. Esta derivada se utiliza luego en la fase hacia atrás.

2

**Fase hacia atrás:** el objetivo de este paso es aprender el gradiente de la función de pérdida con respecto a los diferentes pesos de la red. Estos gradientes se utilizan para actualizar los pesos. Este proceso se conoce como fase hacia atrás, ya que los gradientes se aprenden de atrás hacia adelante (comenzando desde la capa de salida).

Una vez que la red neuronal ha sido adecuadamente entrenada, se puede **utilizar para predecir o clasificar datos**.

# Arquitecturas y modelos



Dependiendo de las capas utilizadas en una red, podemos encontrar **tres tipos básicos de redes**: feed-forward, recurrentes y convolucionales. **Las redes más simples se basan exclusivamente en capas totalmente conectadas**, en las cuales todas las salidas están conectadas a las entradas de la siguiente capa. Las redes feed-forward las veremos en detalle más adelante, el resto se explicarán brevemente a continuación.

## Redes neuronales recurrentes

Las redes neuronales (NN) aprenden sobre el conjunto de datos de entrada en cada iteración; sin embargo, ¿qué sucede cuando los datos de entrada son una secuencia temporal, pronósticos meteorológicos o de acciones?

Pues que las redes neuronales clásicas no tienen memoria sobre la información que ha sido procesada, por lo tanto, les es imposible predecir valores futuros sin conocer los valores anteriores. Para superar esto, **las redes neuronales recurrentes (RNN) mantienen información de interacciones pasadas**. Yes que, mientras que las redes clásicas o hacia adelante toman decisiones basadas solo en la entrada actual, las redes recurrentes toman decisiones basadas en las entradas actuales y anteriores.

Fíjate en la comparación entre ambas redes que muestra la siguiente imagen.

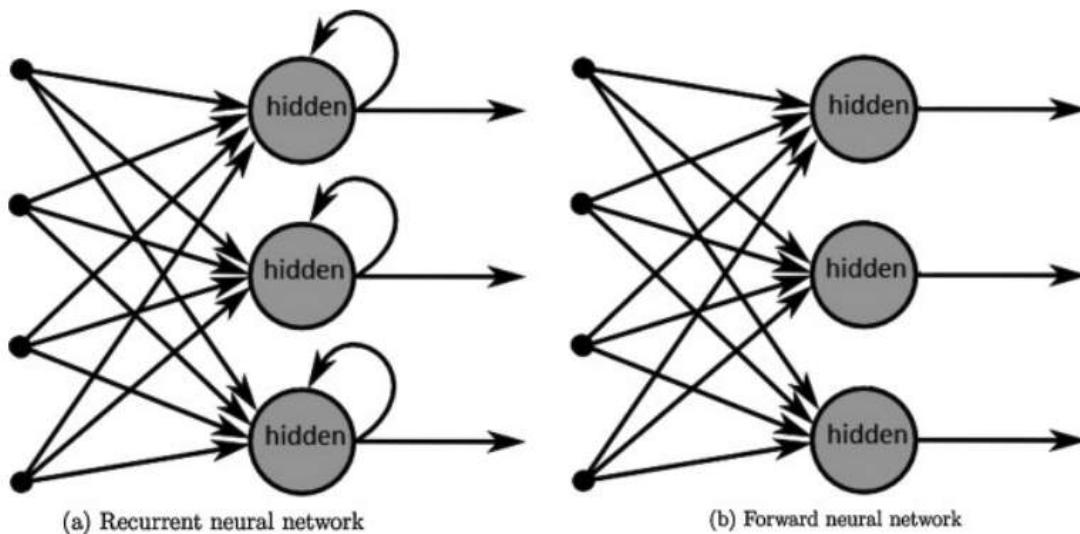


Figura 8. Red neuronal recurrente vs. red neuronal de propagación hacia adelante.

Fuente: W. De Mulder, S. Bethard, and M. F. Moens, *A survey on the application of recurrent neural networks to statistical language modeling*, Computer Speech and Language, vol. 30, no. 1, pp. 61–98, 2015. doi: 10.1016/j.csl.2014. 09.005.

En las **redes neuronales tradicionales, hablamos de neuronas**; sin embargo, para las redes neuronales recurrentes (RNN), el término cambia a celdas, que son una combinación de neuronas y compuertas de activación. Los **tres tipos de celdas más conocidos son**: la RNN simple, las celdas de memoria a corto plazo (LSTM) y las unidades recurrentes agrupadas (GRU). La siguiente figura muestra un ejemplo de todas ellas.

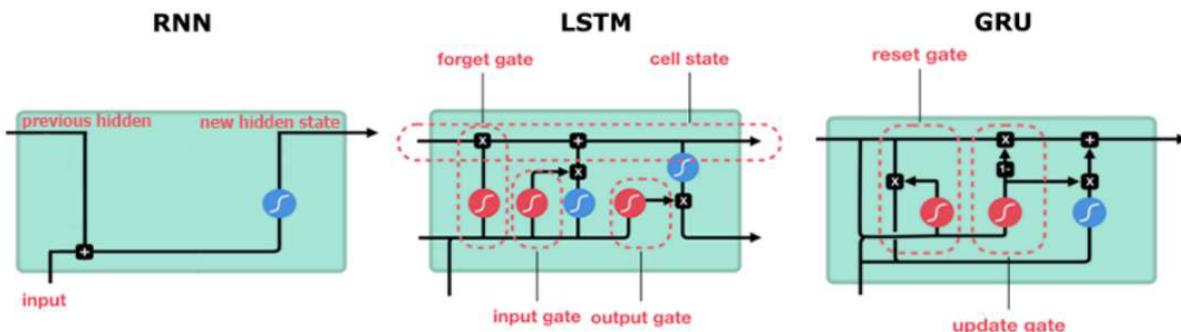


Figura 9. Tipos de neuronas recurrentes.

Fuente: <https://towardsdatascience.com/>.

## La red neuronal recurrente (RNN)

Primero combina la entrada actual y el estado oculto (que actúa como la memoria de la celda anterior), formando un vector que contiene no solo la información actual sino también la información de las entradas anteriores. El nuevo vector pasa a través de la activación tangente hiperbólica ( $\tanh$ ), regulando los valores entre -1 y 1. Este vector es el estado oculto para la siguiente celda de la RNN. Aunque las celdas RNN básicas son muy simples, funcionan bastante bien; sin embargo, debido a la disminución del gradiente de la función de pérdida, es difícil resolver dependencias temporales a largo plazo. A pesar de esto, estas celdas tienen un buen rendimiento en secuencias cortas.

## Las celdas LSTM

Siguen el mismo flujo de trabajo que las celdas RNN: los datos se procesan hacia adelante a través de las celdas; sin embargo, las LSTM muestran más operaciones que permiten olvidar o retener información. En comparación con las RNN, las celdas LSTM tienen una entrada adicional ( $C_{t-1}$ ) y una salida ( $C_t$ ). Esto se llama estado de la celda y actúa como una autopista de información, transfiriendo la información a través de toda la red. De esta manera, la red puede mantener la información durante períodos prolongados, reduciendo el problema de la memoria a corto plazo causado por la disminución del gradiente. En las LSTM, no toda la información anterior se alimenta a la siguiente celda, ya que la compuerta de olvido controla la cantidad de información.

## La celda GRU

Es una versión más nueva y simple que la LSTM, que utiliza el estado oculto en lugar del estado de la celda para transferir información. Al igual que en LSTM, la compuerta de actualización decide qué información retener y cuál olvidar. La compuerta de reinicio controla la cantidad de información pasada que se olvida. Este tipo de celda funciona de manera similar a LSTM, sin embargo, como las operaciones realizadas por las celdas son bastante menores, son más rápidas que las LSTM.

## Redes neuronales convolucionales

Como se mencionó anteriormente, al crear una red neuronal, en la capa de entrada se necesitan tantas neuronas como características se utilicen. Pero ¿qué sucede cuando se desea utilizar una imagen? En este caso, se debería crear una capa con tantas neuronas como píxeles en la imagen.

Con la calidad de las cámaras actuales, tanto fotográficas como de vídeo, este hecho resulta inviable (una imagen típica puede tener alrededor de doce millones de píxeles). Si se intentara crear una red de tal tamaño, el tiempo necesario para entrenarla sería enorme. Para resolver este problema, se utilizan las redes neuronales convolucionales (CNN).

---

Las CNN son redes neuronales aplicadas, principalmente, a problemas o retos relacionados con el procesamiento de imágenes. Funcionan extrayendo las características representativas de la imagen.

En la siguiente imagen que presentamos, se puede ver un ejemplo de una CNN; la parte antes de *flattened* o aplanado se encarga de extraer las características, mientras que la parte posterior se encarga de clasificarlas empleando dichas características.

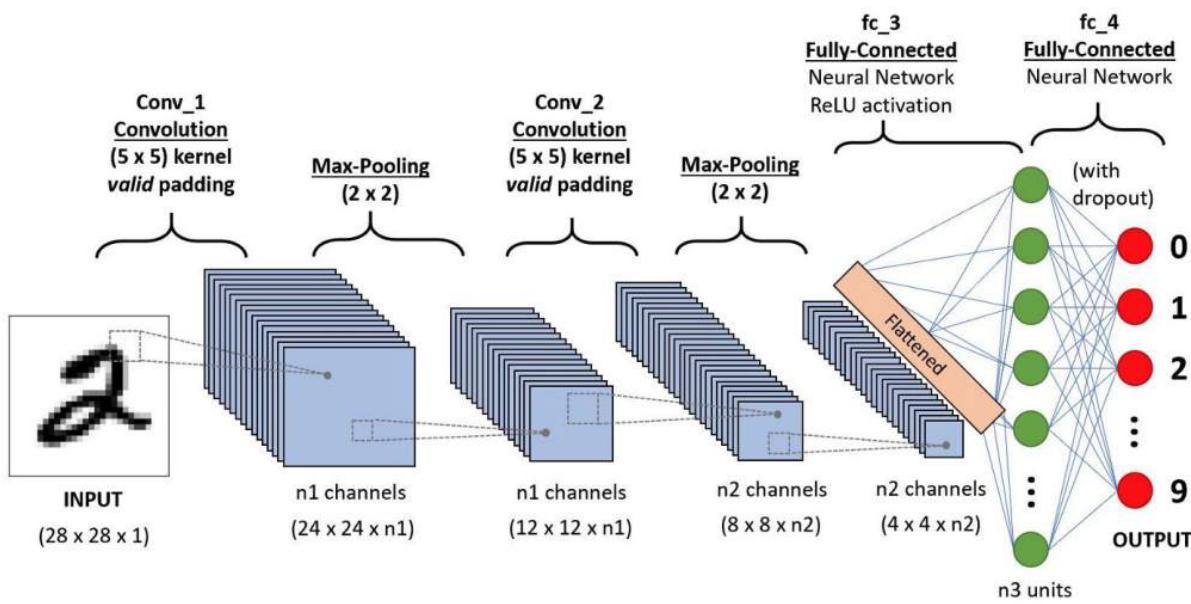


Figura 10. Estructura de una red neuronal convolucional.

Fuente: P. Ratan, *What is the Convolutional Neural Network Architecture?* 2020.  
[analyticsvidhya.com](http://analyticsvidhya.com).

---

En la parte de clasificación, encontramos capas convolucionales y de *max-pooling*. El propósito de las capas convolucionales es extraer características de alto nivel (como bordes o formas).

Las capas de *max-pooling* supervisan la extracción de las características dominantes y así reducen el costo computacional. Una vez extraídas las características relevantes, se aplana para clasificarlas con una red de capas conectadas.

## Entrenamiento y optimización de las redes neuronales

Al trabajar con algoritmos de aprendizaje automático y profundo, tenemos dos tipos de parámetros:

- **Hiperparámetros:** los parámetros que el usuario puede ajustar antes del entrenamiento.
- **Parámetros del modelo:** los parámetros intrínsecos del modelo que son aprendidos por el modelo durante el entrenamiento y no pueden ser modificados por el usuario.

Los hiperparámetros afectan la estructura inicial del modelo, por lo que influyen en el proceso de aprendizaje de los datos. Los valores de estos hiperparámetros difieren de un problema a otro y encontrar el mejor valor para ellos es un problema de optimización.

---

**Para encontrar una solución óptima para los hiperparámetros, se pueden utilizar tanto la búsqueda aleatoria (RS, por sus siglas en inglés) como la búsqueda en rejilla (GS, por sus siglas en inglés). Ambos métodos funcionan de manera similar: seleccionas valores para los hiperparámetros que evalúas y entrenas y pruebas el modelo.**

El proceso de entrenamiento y evaluación generalmente se realiza varias veces, eligiendo aleatoriamente los subconjuntos (*k-fold*) y proporcionando el resultado final como el promedio de todas las ejecuciones. Una vez que se conoce la precisión del modelo con estos parámetros, se registra en una tabla y se cambian los hiperparámetros. Al final, se elige la combinación de hiperparámetros que brinda la mejor precisión.

### **La diferencia entre RS y GS radica en la elección de los valores de hiperparámetros.**

- Por un lado, **en RS se establece un rango de valores** y el algoritmo elige aleatoriamente un valor dentro de ese rango.
- Por otro lado, **GS requiere que los valores de los hiperparámetros** estén detallados y específicamente especificados de antemano y elige combinaciones aleatorias de estos valores.

En la siguiente imagen vemos un ejemplo de GS frente a RS.

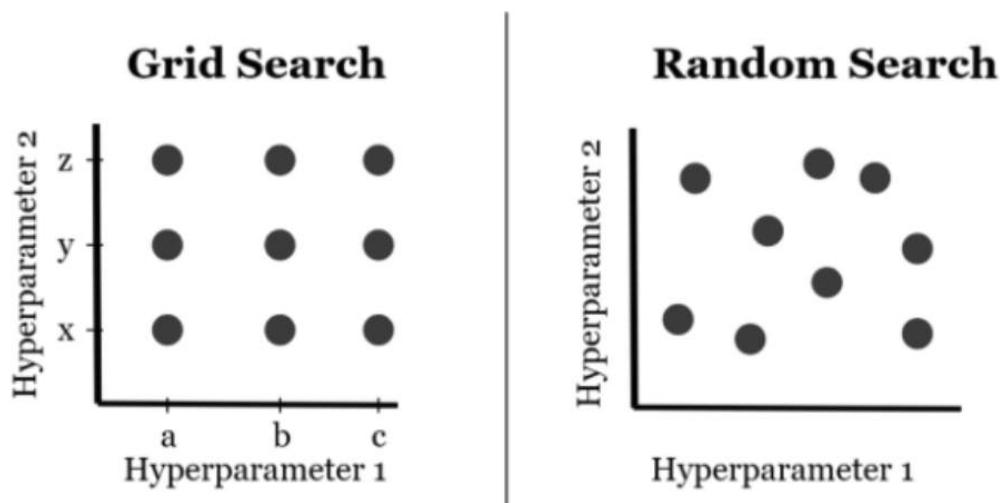


Figura 11. Random search vs. grid search.

Fuente: [https://www.researchgate.net/figure/Schema-representant-les-methodes-Grid-Search-et-Random-Search\\_fig33\\_348232336](https://www.researchgate.net/figure/Schema-representant-les-methodes-Grid-Search-et-Random-Search_fig33_348232336).

Ambos algoritmos trabajan con un conjunto limitado de valores; sin embargo, debido a que en cada iteración RS utiliza valores diferentes, se prueban más combinaciones de hiperparámetros y es posible encontrar una combinación que no habríamos elegido de antemano. Sin embargo, al probar con valores aleatorios, puede suceder que las combinaciones probadas no ofrezcan un buen rendimiento.

Pues bien, **ambos algoritmos resuelven el problema de optimizar los modelos**, y con suficientes iteraciones, ambos encuentran la solución óptima.

# Medición desempeño de un algoritmo de DL



---

Una vez que se ha entrenado un algoritmo se necesita saber el rendimiento de este, para ello hace uso de un subconjunto de datos.

---

**Este subconjunto tiene la etiqueta correspondiente a la salida esperada de la red.**

---

Para medir este desempeño se emplean diferentes métricas que miden la calidad de la salida del modelo. La métrica por usar depende de si la tarea que **nos ocupa es de clasificación o regresión**.

Similar a las métricas que hemos estudiado en machine learning, vamos a revisar, a continuación, **las más empleadas para la evaluación de los modelos de deep learning**.

1

**Exactitud (accuracy):** probablemente la métrica más sencilla y directa para tareas de clasificación. Es la proporción de predicciones correctas respecto al total de predicciones. Aunque la exactitud puede ser útil, es importante tener en cuenta que puede ser engañosa si las clases están desequilibradas.

2

**Precisión (precision)**: mide la precisión de las predicciones positivas en una tarea de clasificación. Es la proporción de verdaderos positivos (elementos correctamente etiquetados como positivos), sobre todo, la suma de verdaderos positivos y falsos positivos (elementos incorrectamente etiquetados como positivos).

3

**Sensibilidad (recall)**: esta métrica mide la fracción de positivos que fueron correctamente identificados en una tarea de clasificación, es decir, la proporción de verdaderos positivos respecto a la suma de verdaderos positivos y falsos negativos (positivos incorrectamente etiquetados como negativos).

4

**F1-Score**: media armónica de la precisión y el *recall*. Intenta encontrar un equilibrio entre precisión y *recall* en una tarea de clasificación. Es particularmente útil cuando hay una distribución desigual de clases, ya que busca un equilibrio entre precisión y *recall*.

5

**Error absoluto medio (MAE)**: promedio de las diferencias absolutas entre los valores predichos y reales. Esta métrica es altamente relevante para tareas de regresión, proporcionando una idea de cuán incorrectas fueron las predicciones. La medida otorga un peso igual a todos los errores ya sean grandes o pequeños.

6

**Error cuadrático medio (MSE)**: promedio de las diferencias al cuadrado entre los valores predichos y reales. Se utiliza comúnmente en tareas de regresión. Elevar al cuadrado la diferencia amplifica el impacto de errores grandes en la métrica de error general, lo que significa que el modelo se penaliza más severamente por hacer predicciones que difieren significativamente del valor real correspondiente.

7

**Raíz del error cuadrático medio (RMSE)**: raíz cuadrada del MSE. Es una métrica popular para tareas de regresión. Al aplicar la raíz cuadrada al MSE, la métrica de error vuelve a la misma unidad que la variable de salida, lo que puede facilitar su interpretación en comparación con el MSE.

8

**R-cuadrado:** representa la proporción de varianza que ha sido explicada por las variables independientes en el modelo. Esta métrica de regresión proporciona una indicación de la bondad del ajuste y, por lo tanto, una medida de cuán bien es probable que el modelo prediga muestras no vistas, a través de la proporción de varianza explicada.

9

**Matriz de confusión:** una matriz de confusión es un diseño tabular que visualiza el rendimiento de un algoritmo. Cada fila de la matriz representa las instancias en una clase real, mientras que cada columna representa las instancias en una clase predicha o viceversa. El nombre proviene de que facilita ver si el sistema está confundiendo dos clases. Esta matriz se aplica principalmente a tareas de clasificación.

10

**Curva ROC y AUC:** la curva ROC (característica operativa del receptor) es un gráfico que ilustra la tasa de verdaderos positivos (TPR) frente a la tasa de falsos positivos (FPR) en varios ajustes de umbral. El AUC (área bajo la curva) mide toda el área bidimensional debajo de la curva ROC completa, desde (0,0) hasta (1,1). Un modelo con un rendimiento predictivo perfecto tendrá un AUC de 1, mientras que un modelo con rendimiento predictivo aleatorio tendrá un AUC de 0.5. Esto es típicamente relevante para problemas de clasificación.

---

**A la hora de evaluar un modelo es importante hacerlo empleando más de una métrica, y hacerlo con objetividad y crítica. Hay que asegurarse que el conjunto de evaluación esté balanceado y sea representativo. Aunque a todos nos gusta ver una métrica de 0,99 de precisión, este resultado rara vez se encuentra en los problemas del mundo real.**

# Casos de usos y limitaciones



Qualentum Lab

---

Numerosas técnicas de deep learning se han adoptado tanto en el ámbito académico como en la industria para abordar desafíos, a continuación, podemos encontrar algunos ejemplos de aplicaciones del deep learning.

## Reconocimiento de voz

La aplicación del deep learning ha divergido y se ha entendido en la capacidad de sus éxitos en el reconocimiento del habla o la comunicación. En las últimas décadas, **los algoritmos de deep learning se han aplicado ampliamente** en áreas como modelado acústico y ASR (reconocimiento automático del habla).

## Visión artificial

---

La visión artificial tiene como objetivo hacer que las computadoras comprendan y procesen con precisión datos visuales, como vídeos e imágenes.

El objetivo principal de la visión artificial es proporcionar a las computadoras la capacidad de funcionar de manera similar al cerebro humano. Teóricamente, la visión artificial se refiere al control lógico que estudia cómo separar la información de las imágenes en sistemas artificiales.

Los subdominios de la visión artificial **incluyen la detección y reconocimiento de objetos**, estimación de objetos, posición de objetos, detección de eventos, reconstrucción de escenas, restauración de imágenes, edición de imágenes, mejora de videos y aprendizaje estadístico. Por lo tanto, en la visión artificial, los modelos de deep learning son muy útiles.

## Reconocimiento de patrones

El reciente avance en modelos de deep learning ha proporcionado nuevas formas de abordar el problema del reconocimiento de patrones. **El reconocimiento de patrones es un área científica** que se centra en la identificación de secuencias en cada entrada.

Es un concepto general que abarca varios subdominios, como el etiquetado de habla, regresión, etiquetado y clasificación de secuencias. Existe una creciente necesidad de procesamiento y salida de información debido al desarrollo industrial, lo que plantea nuevas tendencias y desafíos para el reconocimiento de patrones.

## Detección

La detección en diagnósticos médicos, seguridad, objetos en imágenes, irregularidades financieras o fallas en un sistema se mejora mediante la aplicación de deep learning. Por lo tanto, las técnicas de deep learning desempeñan un papel esencial en la detección, por ejemplo, cuando se aplican al cáncer de mama han desbancado a los métodos tradicionales.

El rendimiento de estas técnicas se puede comparar de manera relativa con otros enfoques en la detección de delitos, como el perfil de ADN y la actividad y el uso de grandes datos para la detección de fraudes financieros. A pesar de las numerosas publicaciones sobre la utilización de las NN en diferentes desafíos médicos, hay pocas revisiones disponibles que expliquen la arquitectura para mejorar los métodos de detección en términos de rendimiento, precisión, sensibilidad y especificidad.

---

**La capacidad de detección es un subdominio comúnmente conocido o un cálculo en la visión artificial que busca comprender, ubicar, clasificar o diferenciar los objetos de la imagen objetivo. Por ejemplo, durante las tareas de detección, una imagen puede escanearse para conocer ciertas características o cualidades específicas, como el uso de la detección de imágenes en el diagnóstico médico de células o tejidos anormales.**

En la siguiente imagen, podemos observar el número de nuevas aplicaciones en función de la industria y de la labor llevada a cabo.

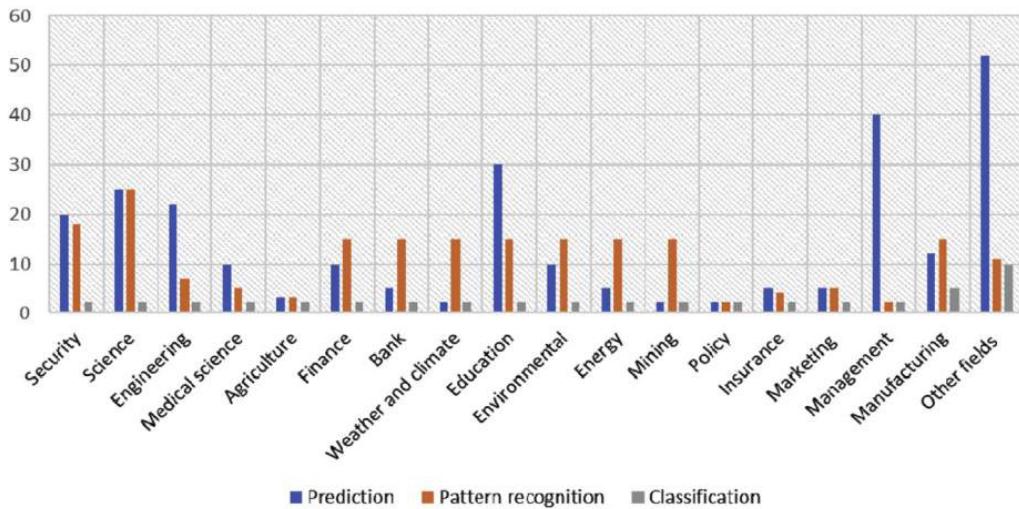


Figura 12. Número de aplicaciones de deep learning por sector.

Fuente: Abiodun Ol, Jantan A, Omolara AE, et al. *State-of-the-art in artificial neural network applications: A survey*. *Heliyon*. 2018 Nov;4(11):e00938. DOI: 10.1016/j.heliyon.2018.e00938.

---

## Limitaciones

Hemos visto algunos ejemplos de la aplicación del deep learning, sin embargo, como cualquier tecnología, el DL tiene sus limitaciones. Seguidamente se exponen algunos casos ejemplificadores.

1

### El problema de la caja negra

Una de las limitaciones más significativas del deep learning es el problema de la ‘caja negra’. Los algoritmos de deep learning **funcionan creando modelos matemáticos complejos que se entranan con grandes conjuntos de datos**. Si bien estos modelos pueden predecir resultados con precisión, puede ser desafiante entender cómo llegaron a esas predicciones.

Esta falta de transparencia puede ser problemática, especialmente en aplicaciones donde están en juego vidas humanas. Por ejemplo, si el modelo de deep learning de un automóvil autónomo comete un error, puede no estar claro por qué sucedió, lo que dificulta prevenir que vuelva a ocurrir.

2

## Sobreajuste

Otra limitación del deep learning es el sobreajuste. **Este fenómeno ocurre cuando un modelo es demasiado complejo y se ajusta demasiado a los datos de entrenamiento**, lo que hace que funcione mal con nuevos datos. Los modelos de deep learning son especialmente propensos al sobreajuste debido a su complejidad y la gran cantidad de parámetros que tienen.

Para prevenir el sobreajuste, los modelos de deep learning deben diseñarse cuidadosamente y entrenarse con conjuntos de datos grandes y diversos. Sin embargo, incluso con estas precauciones, el sobreajuste aún puede ocurrir, limitando la precisión de los modelos de deep learning.

3

## Falta de comprensión contextual

Los modelos de deep learning son excelentes para el reconocimiento de patrones y pueden **identificar objetos con precisión, reconocer el habla y traducir idiomas**. Sin embargo, carecen de comprensión contextual, lo que puede limitar su eficacia en ciertas aplicaciones.

Por ejemplo, un modelo de deep learning puede reconocer con precisión la señal de tráfico alto, pero puede no entender el contexto de la señal, como la necesidad de tener que detenerse en una intersección.

4

## Requisitos de datos

Los modelos de deep learning **requieren grandes cantidades de datos** para entrenarse de manera efectiva. Este requisito puede ser una limitación significativa en aplicaciones donde los datos son escasos o difíciles de obtener.

Por ejemplo, en el campo médico, puede ser difícil obtener conjuntos de datos grandes y diversos necesarios para entrenar modelos de deep learning de manera efectiva. Esta limitación puede dificultar el uso del deep learning en aplicaciones médicas, como el diagnóstico de enfermedades.

5

## Intensivo en cómputo

Los modelos de deep learning son **intensivos en cómputo y requieren hardware potente para entrenarse y ejecutarse de manera efectiva**. Este requisito puede ser una limitación significativa para organizaciones con recursos informáticos limitados.

Además, el consumo de energía de los modelos de deep learning puede ser significativo, lo que los hace poco amigables con el medio ambiente en algunos casos.

---

El deep learning ha revolucionado el campo de la inteligencia artificial y ha permitido avances significativos en diversas aplicaciones.

Sin embargo, es esencial comprender las limitaciones del deep learning y cómo pueden afectar su eficacia. Al comprender estas limitaciones, podemos desarrollar mejores modelos de deep learning y utilizarlos de manera más efectiva en aplicaciones del mundo real.

---

**Tras conocer las limitaciones del aprendizaje profundo, pasaremos a explicar en detalle las redes neuronales de alimentación hacia adelante o *feed-forward neural network*.**

# Definición de redes de alimentación hacia adelante



Las redes neuronales de alimentación hacia adelante, las ya mencionadas *feed-forward network*, son el tipo más común de red neuronal.

- Están compuestas por capas de neuronas, y la información fluye en una sola dirección desde la capa de entrada hasta la capa de salida.
- Cada neurona realiza un cálculo simple, y este valor se transmite a la siguiente neurona.
- La capa final de la red se corresponde con la salida, bien sea un valor de clasificación, bien de regresión.

Este tipo de redes se utiliza para una variedad de tareas como la clasificación de imágenes, el procesamiento de lenguaje natural y el reconocimiento de voz.

También se utiliza para tareas que involucran predecir valores continuos, como predecir el precio de una acción o la calificación de una película.

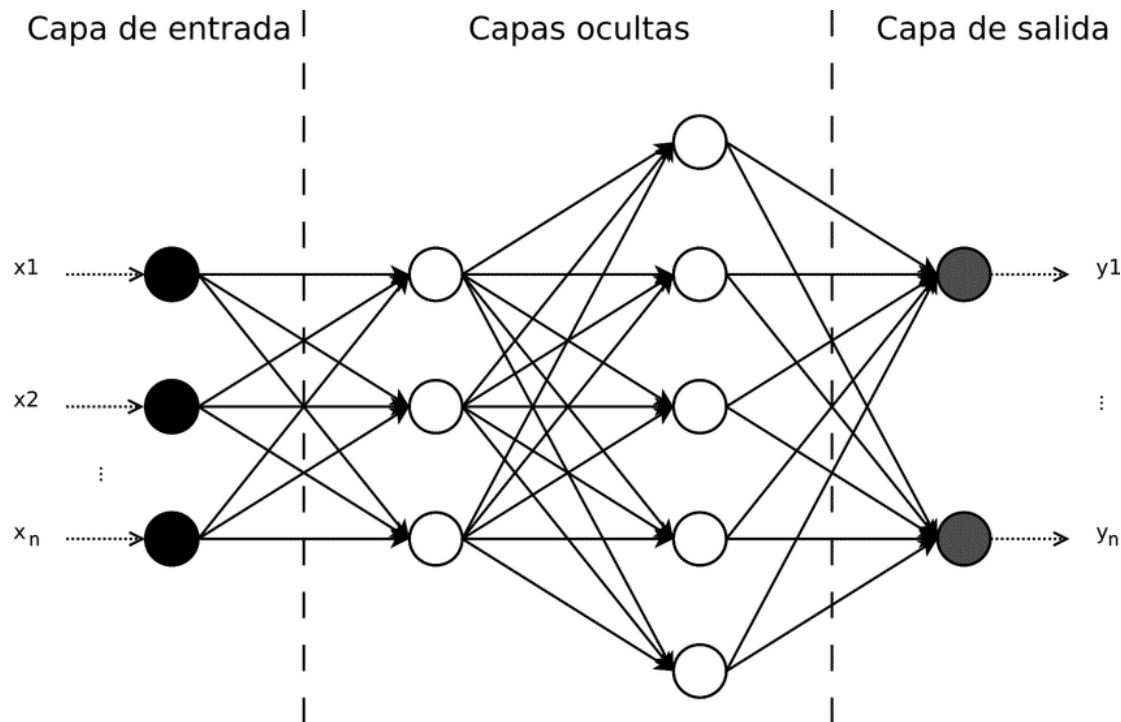


Figura 13. Arquitectura de una red neuronal de alimentación hacia delante de 4 capas.

Fuente: [https://www.researchgate.net/figure/Red-neuronal-artificial-de-cuatro-capas\\_fig1\\_323985249](https://www.researchgate.net/figure/Red-neuronal-artificial-de-cuatro-capas_fig1_323985249).

Como podemos ver en imagen anterior, la arquitectura de una red neuronal de alimentación hacia adelante suele ser **un conjunto de capas**.

- La **capa de entrada** es la primera y se encarga de recibir los datos de entrada.
- Las **capas intermedias**, llamadas capas ocultas, son las responsables de aprender y extraer las características de los datos.
- La **capa de salida** es la última y se encarga de generar la predicción de salida.

La principal fortaleza de este tipo de redes es que son **simples y eficientes**, además son relativamente fáciles de entrenar, lo que las convierte en una buena opción para muchas tareas. Sin embargo, las redes neuronales de alimentación hacia adelante no son tan efectivas para manejar datos secuenciales como las redes neuronales recurrentes ni para el procesado de imágenes como las redes neuronales convolucionales.

# Funcionamiento y aprendizaje del modelo



El funcionamiento de una red de propagación hacia delante requiere de **dos fases**:

- La propagación hacia delante o *forward propagation*.
- La retropropagación, propagación hacia atrás o *backpropagation*.

## Propagación hacia delante

En esta fase la información se propaga desde la capa de entrada hasta la capa de salida. Comienza con la inicialización de la red, donde las neuronas de la capa de entrada representan los atributos o parámetros de entrada y cada conexión tiene pesos iniciales aleatorios.

Los datos se introducen en la red a través de la capa de entrada, y cada neurona realiza las operaciones de multiplicación y activación mediante una función de activación, que puede ser lineal o no lineal.

La información procesada se propaga a través de capas ocultas, cada una ajustando sus pesos durante el entrenamiento para mejorar la capacidad predictiva de la red.

La configuración y los pesos de las conexiones determinan la capacidad de la red para aprender patrones, extraer información y hacer predicciones precisas, repitiendo este proceso para cada conjunto de datos durante el entrenamiento e inferencia.

Por último, la información llega a la capa de salida, donde se realizan operaciones finales de multiplicación y activación. La salida de esta capa representa la predicción final de la red para un conjunto de datos de entrada. Esta salida se compara con las etiquetas reales para calcular el error, utilizado en el proceso de retropropagación durante el entrenamiento con la finalidad de ajustar los pesos y minimizar el error en futuras predicciones.

## Propagación hacia atrás o retropropagación

---

La retropropagación es esencial para el entrenamiento efectivo de una red neuronal. Este proceso ajusta los pesos de la red para minimizar el error entre las predicciones de la red y valores reales.

A continuación, vamos a estudiar este proceso.

El proceso de retropropagación **comienza con el cálculo del error entre la salida predicha por la red y la salida real conocida**. Este error se utiliza como guía para ajustar los pesos de la red y mejorar las futuras predicciones. Este cálculo se realiza utilizando la función de pérdida o loss function.

- La retropropagación implica la **transmisión de este error desde la capa de salida hacia atrás**, a través de las capas ocultas, hasta la capa de entrada. En cada capa, se calcula la contribución al error y se ajustan los pesos en consecuencia. Durante este proceso, se utiliza el algoritmo de descenso del gradiente, que actualiza iterativamente los pesos en la dirección opuesta al gradiente del error. Esto significa que los pesos se ajustan para minimizar el error en las predicciones.
- La tasa de aprendizaje o *learning rate* es un **parámetro crucial en el descenso del gradiente**, ya que determina el tamaño de los pasos de ajuste de los pesos. Un valor demasiado alto puede hacer que la red oscile, mientras que un valor demasiado bajo puede hacer que el aprendizaje sea lento. Podemos observar un ejemplo de esto en la siguiente imagen.

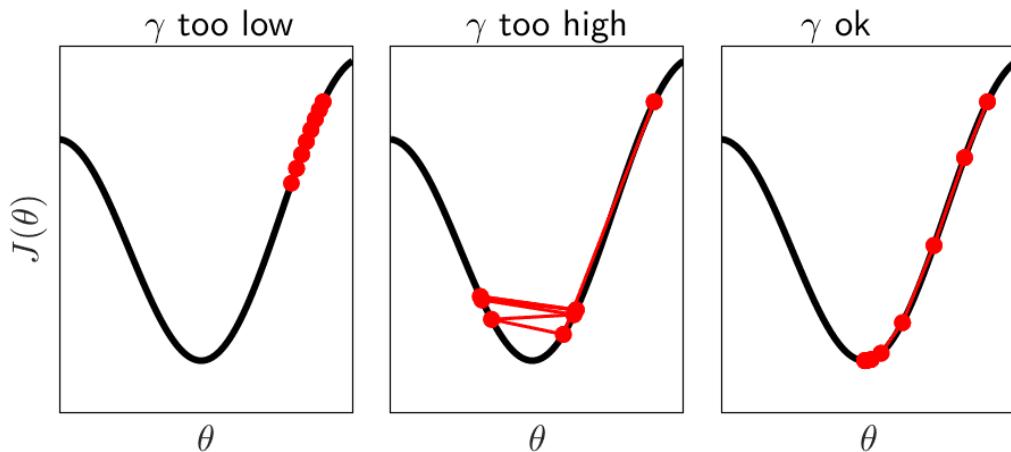


Figura 14. Tasa de aprendizaje alta, baja y correcta.

Fuente: [https://www.researchgate.net/figure/Red-neuronal-artificial-de-cuatro-capas\\_fig1\\_323985249](https://www.researchgate.net/figure/Red-neuronal-artificial-de-cuatro-capas_fig1_323985249).

---

Este proceso de retropropagación se repite para múltiples iteraciones (épocas) hasta que la red alcanza un nivel de rendimiento satisfactorio. Los pesos ajustados durante la retropropagación se utilizan luego en la fase de *forward propagation* para realizar predicciones más precisas. Así que recuerda...

- El *forward propagation* y el *backpropagation* son procesos cruciales en el entrenamiento efectivo de redes neuronales.
- El *forward propagation* permite que la información se propague desde la capa de entrada hasta la capa de salida, realizando operaciones de multiplicación y activación en cada capa. Esto facilita la generación de predicciones basadas en los pesos actuales de la red.
- El *backpropagation* desencadena la retroalimentación del error desde la capa de salida hacia atrás, ajustando los pesos de la red mediante el algoritmo de descenso del gradiente.
- Este proceso iterativo minimiza el error entre las predicciones y las salidas reales, permitiendo a la red aprender y mejorar su capacidad predictiva.

---

**En conjunto, el *forward propagation* y el *backpropagation* son esenciales para que la red neuronal aprenda patrones, optimice sus pesos y realice predicciones precisas en diversas tareas.**

## **Funciones de perdida, optimización y regularización**

Hemos visto la importancia de la propagación hacia delante y detrás en el proceso de aprendizaje de una red neuronal. Sin embargo, en este proceso se ven involucradas dos funciones más: la función de perdida y la función de optimización. Además, para favorecer el aprendizaje existen las técnicas de regularización. Todo esto se explica en las siguientes líneas.

1

### **Función de pérdida**

Una función de pérdida es aquella que compara los valores de salida deseados con los valores predichos por la red neuronal; mide cómo de bien el modelo de la red neuronal se ajusta a los datos de entrenamiento. Durante el entrenamiento, **el objetivo es minimizar esta pérdida entre las salidas predichas y las salidas deseadas**. Este proceso de minimización de la pérdida durante el entrenamiento es esencial para que la red neuronal aprenda de manera efectiva y realice predicciones más precisas en datos no vistos.

Las funciones de pérdida desempeñan un papel crucial en este proceso al cuantificar la discrepancia entre las predicciones del modelo y las salidas reales, proporcionando una guía para la actualización de pesos y sesgos.

En el aprendizaje supervisado, existen dos tipos de funciones de pérdida, correlacionadas con los **dos tipos principales de redes neuronales**: las de pérdida de regresión y las de clasificación.

#### LAS FUNCIONES DE PÉRDIDA DE REGRESIÓN

#### LAS FUNCIONES DE PÉRDIDA DE CLASIFICACIÓN

Se utilizan en redes neuronales de regresión, donde, dado un valor de entrada, el modelo predice un valor de salida correspondiente en lugar de etiquetas preseleccionadas. Como ejemplos de funciones de pérdida de regresión están el error cuadrático medio (MSE) y el error absoluto medio (MAE).

#### LAS FUNCIONES DE PÉRDIDA DE REGRESIÓN

#### LAS FUNCIONES DE PÉRDIDA DE CLASIFICACIÓN

Estas se utilizan en redes neuronales de clasificación, donde, dada una entrada, la red neuronal produce un vector de probabilidades de que el input pertenezca a varias categorías prestablecidas. Luego, se selecciona la categoría con la probabilidad más alta de pertenencia. La entropía cruzada binaria (*Binary Cross-Entropy*) y la entropía cruzada categórica (*Categorical Cross-Entropy*) son ejemplos de este tipo de funciones.

A continuación, te compartimos **más detalles de las funciones de perdida** más comunes.

- **Error cuadrático medio (MSE)**

---

El error cuadrático medio (MSE) es una de las funciones de pérdida más populares.

Calcula el promedio de las diferencias al cuadrado entre las salidas deseadas y las salidas predichas.

$$MSE = \frac{1}{n} \sum_{i=1}^n (y^{(i)} - \hat{y}^{(i)})^2$$

Esta función tiene varias propiedades que la hacen especialmente adecuada para calcular la pérdida. La diferencia se eleva al cuadrado, lo que significa que no importa si el valor predicho está por encima o por debajo del valor deseado; sin embargo, se penalizan los valores con un gran error.

---

El MSE también es una función convexa con un mínimo global claramente definido, lo que nos permite utilizar más fácilmente la optimización por descenso de gradiente para ajustar los valores de los pesos.

---

- **Error absoluto medio (MAE)**
- 

El error absoluto medio (MAE) encuentra el promedio de las diferencias absolutas entre las salidas deseadas y las salidas predichas.

$$MAE = \frac{1}{n} \sum_{i=1}^n |y^{(i)} - \hat{y}^{(i)}|$$

Esta función de pérdida se utiliza como alternativa al MSE en algunos casos. Como se ha dicho antes, el MSE es altamente sensible a los valores atípicos, lo que puede afectar a la pérdida porque la distancia se eleva al cuadrado. El MAE se utiliza en casos en los que los datos de entrenamiento tienen una gran cantidad de valores atípicos para mitigar este efecto. Al considerar las **diferencias absolutas en lugar de cuadráticas**, el MAE ofrece una medida más robusta de la pérdida en presencia de valores extremos o espurios.

- **Entropía cruzada binaria/pérdida logarítmica (Binary Cross-Entropy/Log Loss)**

Esta función de pérdida **se utiliza en modelos de clasificación binaria**, donde el modelo toma una entrada y debe clasificarla en una de las dos categorías preestablecidas.

$$CE\ Loss = \frac{1}{n} \sum_{i=1}^N - (y_i \cdot \log(p_i) + (1 - y_i) \cdot \log(1 - p_i))$$

Las redes neuronales de clasificación **funcionan al producir un vector de probabilidades** (la probabilidad de que la entrada dada se ajuste a cada una de las categorías preestablecidas) y, luego, se selecciona la categoría con la probabilidad más alta como la salida final.

En la clasificación binaria, solo hay dos posibles valores reales de  $y$ : - 0 o 1. Por lo tanto, para determinar con precisión la pérdida entre los valores reales y predichos, es necesario comparar el valor real (0 o 1) con la probabilidad de que la entrada se ajuste a esa categoría. La función de pérdida penaliza fuertemente las predicciones incorrectas, proporcionando una medida efectiva de la calidad de la clasificación binaria del modelo.

- **Pérdida de entropía cruzada categórica (Categorical Cross-Entropy Loss)**

En casos donde el número de clases es mayor que dos, utilizamos la entropía cruzada categórica. Esta sigue un proceso muy similar a la entropía cruzada binaria.

$$CE\ Loss = -\frac{1}{n} \sum_{i=1}^N \sum_{j=1}^M y_{ij} \cdot \log(p_{ij})$$

La entropía cruzada binaria es un caso especial de la entropía cruzada categórica, donde M = (2 - el número de categorías) es 2. En la entropía cruzada categórica, el modelo produce un vector de probabilidades para cada clase, y la función de pérdida compara estas distribuciones de probabilidad con las clases reales. Este tipo de pérdida es fundamental para problemas de clasificación con más de dos clases, y su objetivo es minimizar la discrepancia entre las predicciones del modelo y las verdaderas clases en el conjunto de datos.

La selección de la función de perdida **varía según la naturaleza del problema y la adecuada elección de esta contribuye a un entrenamiento más efectivo**, ya que guía la optimización de los pesos hacia la dirección correcta para minimizar la discrepancia entre las predicciones y las salidas reales.

## 2

### Algoritmos de optimización

En este apartado explicaremos los diferentes tipos de algoritmos de optimización más comunes. **Estos algoritmos se utilizan para ajustar los pesos de la red durante el entrenamiento con el objetivo de minimizar la función de pérdida**. El proceso de optimización busca encontrar los valores óptimos de los pesos que resulten en predicciones más precisas.

- **Descenso del gradiente (*Gradient Descent*)**

El descenso del gradiente es el algoritmo de optimización más básico, pero más utilizado.

---

El descenso del gradiente es un algoritmo de optimización de primer orden que depende de la primera derivada de una función de pérdida.

Calcula en qué dirección deben alterarse los pesos para que la función pueda llegar a un mínimo. A través de la retropropagación, la pérdida se transfiere de una capa a otra y los parámetros del modelo, también conocidos como pesos, se modifican según las pérdidas para minimizarlas.

- **Descenso del gradiente estocástico (*Stochastic Gradient Descent o SGD*)**

---

Es una variante del descenso del gradiente que intenta actualizar los parámetros del modelo con más frecuencia.

En este caso, los parámetros del modelo **se alteran después de calcular la pérdida en cada ejemplo de entrenamiento**. Por lo tanto, si el conjunto de datos contiene 1000 filas, SGD actualizará los parámetros del modelo 1.000 veces en un ciclo de conjunto de datos en lugar de una vez, como en el descenso del gradiente.

Debido a las actualizaciones frecuentes de los parámetros del modelo, estos tienen una alta varianza y fluctuaciones en las funciones de pérdida a diferentes intensidades.

- **Adam (Adaptive Moment Estimation)**
- 

Adam es un algoritmo de optimización que trabaja con momentos de primer y segundo orden.

La intuición detrás de Adam es que no queremos avanzar demasiado rápido solo porque podemos saltar sobre el mínimo, dicho de otro modo: queremos disminuir la velocidad un poco para una búsqueda más cuidadosa. Además de almacenar un promedio de decaimiento exponencial de los gradientes cuadrados pasados, Adam también mantiene un promedio de decaimiento exponencial de los gradientes pasados, denotado como  $m(t)$ .

---

**$m(t)$  y  $v(t)$  son valores del primer momento, que es la media, y el segundo momento, que es la varianza no centrada de los gradientes, respectivamente.**

---

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t},$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}.$$

En la siguiente tabla se exponen **las ventajas y desventajas** de los algoritmos vistos arriba:

|                               | Ventajas   | Desventajas   |
|-------------------------------|--|---|
| <b>Descenso del gradiente</b> | <ul style="list-style-type: none"><li>• Cálculos sencillos.</li><li>• Fácil implementación.</li><li>• Fácil comprensión.</li></ul> | <ul style="list-style-type: none"><li>• Puede quedar atrapado en mínimos locales.</li><li>• Los pesos se modifican después de calcular el gradiente en todo el conjunto de datos. Por lo tanto, si el conjunto de datos es demasiado grande, esto puede tardar años en converger hacia el mínimo.</li><li>• Requiere de una gran memoria para calcular el gradiente en todos los datos.</li></ul> |

|             | Ventajas   | Desventajas  |
|-------------|--|--|
| <b>SGD</b>  | <ul style="list-style-type: none"> <li>Actualizaciones frecuentes de los parámetros del modelo, por lo tanto, converge en menos tiempo.</li> <li>Requiere menos memoria, ya que no es necesario almacenar valores de funciones de pérdida.</li> <li>Puede encontrar nuevos mínimos.</li> </ul> | <ul style="list-style-type: none"> <li>Alta varianza en los parámetros del modelo.</li> <li>Puede seguir aumentando, incluso después de alcanzar el mínimo global.</li> <li>Para obtener la misma convergencia que el descenso del gradiente, es necesario reducir lentamente el valor de la tasa de aprendizaje.</li> </ul> |
| <b>ADAM</b> | <ul style="list-style-type: none"> <li>El método es rápido y converge rápidamente.</li> <li>Rectifica la tasa de aprendizaje que disminuye y la alta varianza.</li> </ul>  | <ul style="list-style-type: none"> <li>Resulta costoso a nivel computacional.</li> </ul>   |

---

**Como ya habrás deducido, la elección de la función de optimización dependerá del problema específico, la cantidad de datos disponibles y otros factores. No olvides tampoco que experimentar con diferentes algoritmos de optimización y ajustar sus hiperparámetros pueden ser decisivos para lograr un rendimiento óptimo en el entrenamiento de una red neuronal.**

# Conclusiones



Como hemos podido ver a lo largo del tema, el deep learning representa una faceta clave de la inteligencia artificial, destacando su capacidad para aprender patrones complejos a partir de los datos. Este enfoque ha demostrado ser exitoso en áreas como el reconocimiento de voz, la visión artificial y el diagnóstico médico.

Aunque las redes neuronales profundas han logrado avances impresionantes, no están exentas de desafíos. La opacidad de sus procesos de toma de decisiones, el riesgo de sobreajuste, la dependencia de grandes cantidades de datos y recursos computacionales intensivos son cuestiones que necesitan ser abordadas para mejorar la eficacia y ética de estas tecnologías.

No obstante, debemos reconocer que, a pesar de estas limitaciones, el deep learning ha transformado significativamente el panorama de la inteligencia artificial. Entender estas restricciones impulsa la necesidad de investigaciones continuas y desarrollos para superar obstáculos y aprovechar plenamente el potencial de estas herramientas en aplicaciones del mundo real.

También hemos visto, de manera detallada, el funcionamiento de las redes neuronales de alimentación hacia adelante, destacando su estructura y proceso de aprendizaje. Hemos descubierto la importancia de las funciones de pérdida en el aprendizaje supervisado mediante ejemplos específicos como el error cuadrático medio y la entropía cruzada. Además, hemos explorado los distintos algoritmos de optimización, como el descenso del gradiente y Adam, junto con sus ventajas y desventajas. Otro gran desafío es el del sobreajuste en las redes neuronales y las técnicas de regularización, como *Early Stopping* y *Dropout*, para mejorar la capacidad de generalización del modelo.

---

**Aunque todo se ha explicado en el marco de un tipo de red neuronal, estos conocimientos son comunes para otros tipos existentes, por lo que se pueden aplicar y extrapolar al resto.**

¡Enhorabuena! Fastbook superado



[Qualentum.com](http://Qualentum.com)