

Universidad Complutense de Madrid

Las matemáticas de la inteligencia artificial

Joaquin López Herraiz

This text is disseminated via the Open Education Resource (OER) LibreTexts Project (<https://LibreTexts.org>) and like the hundreds of other texts available within this powerful platform, it is freely available for reading, printing and "consuming." Most, but not all, pages in the library have licenses that may allow individuals to make changes, save, and print this book. Carefully consult the applicable license(s) before pursuing such effects.

Instructors can adopt existing LibreTexts texts or Remix them to quickly build course-specific resources to meet the needs of their students. Unlike traditional textbooks, LibreTexts' web based origins allow powerful integration of advanced features and new technologies to support learning.



The LibreTexts mission is to unite students, faculty and scholars in a cooperative effort to develop an easy-to-use online platform for the construction, customization, and dissemination of OER content to reduce the burdens of unreasonable textbook costs to our students and society. The LibreTexts project is a multi-institutional collaborative venture to develop the next generation of open-access texts to improve postsecondary education at all levels of higher learning by developing an Open Access Resource environment. The project currently consists of 14 independently operating and interconnected libraries that are constantly being optimized by students, faculty, and outside experts to supplant conventional paper-based books. These free textbook alternatives are organized within a central environment that is both vertically (from advance to basic level) and horizontally (across different fields) integrated.

The LibreTexts libraries are Powered by [NICE CXOne](#) and are supported by the Department of Education Open Textbook Pilot Project, the UC Davis Office of the Provost, the UC Davis Library, the California State University Affordable Learning Solutions Program, and Merlot. This material is based upon work supported by the National Science Foundation under Grant No. 1246120, 1525057, and 1413739.

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation nor the US Department of Education.

Have questions or comments? For information about adoptions or adaptations contact info@LibreTexts.org. More information on our activities can be found via Facebook (<https://facebook.com/Libretexts>), Twitter (<https://twitter.com/libretexts>), or our blog (<http://Blog.Libretexts.org>).

This text was compiled on 04/12/2025

TABLE OF CONTENTS

[Licensing](#)

[Texto Preliminar](#)

- [Página del Título](#)
- [Página de Información](#)
- [Tabla de Contenido](#)

[1: Introducción a la Inteligencia Artificial y las Redes Neuronales](#)

- [1.1: Introducción General](#)
- [1.2: Perspectiva Histórica](#)
- [1.3: Aplicaciones presentes y futuras](#)
- [1.4: Objetivos del Curso](#)

[2: Introducción al Cálculo Multivariante](#)

[3: Composición de Funciones. Regla de la Cadena](#)

[4: Aproximación Lineal de una Función Escalar de Varias Variables](#)

[5: Optimización y Método del Gradiente Descendiente](#)

[6: Redes Neuronales](#)

- [6.1: Perceptrón](#)
- [6.2: Funciones de Pérdida](#)
- [6.3: Perceptrón Multicapa](#)

[7: Retropropagación](#)

[8: Entrenamiento de una Red Neuronal](#)

[Apéndices](#)

- [Índice](#)
- [Glosario](#)

[Index](#)

[Glossary](#)

[Detailed Licensing](#)

Licensing

A detailed breakdown of this resource's licensing can be found in [Back Matter/Detailed Licensing](#).

CHAPTER OVERVIEW

Texto Preliminar

[Página del Título](#)

[Página de Información](#)

[Tabla de Contenido](#)

This page titled [Texto Preliminar](#) is shared under a [not declared](#) license and was authored, remixed, and/or curated by [Joaquín López Herraiz](#).

Universidad Complutense de Madrid

Las matemáticas de la inteligencia artificial

Joaquin López Herraiz

Este texto se difunde a través del Proyecto LibreTexts de Recursos Educativos Abiertos (REA) (<https://LibreTexts.org>) y, al igual que los cientos de otros textos disponibles en esta poderosa plataforma, está disponible gratuitamente para leer, imprimir y «consumir». La mayoría de las páginas de la biblioteca, pero no todas, tienen licencias que pueden permitir a las personas hacer cambios, guardar e imprimir este libro. Consulte cuidadosamente las licencias aplicables antes de perseguir dichos efectos.

Los profesores pueden adoptar textos de LibreTexts existentes o remezclarlos para crear rápidamente recursos específicos del curso que satisfagan las necesidades de sus alumnos. A diferencia de los libros de texto tradicionales, los orígenes basados en la web de LibreTexts permiten una poderosa integración de funciones avanzadas y nuevas tecnologías para apoyar el aprendizaje.



La misión de LibreTexts es unir a estudiantes, profesores y académicos en un esfuerzo cooperativo para desarrollar una plataforma en línea fácil de usar para la construcción, personalización y difusión del contenido de REA para reducir la carga de los costos irrazonables de los libros de texto para nuestros estudiantes y la sociedad. El proyecto LibreTexts es una empresa colaborativa multiinstitucional para desarrollar la próxima generación de textos de acceso abierto para mejorar la educación postsecundaria en todos los niveles de educación superior mediante el desarrollo de un entorno de recursos de acceso abierto. El proyecto consiste actualmente en 14 bibliotecas interconectadas y que funcionan de manera independiente y que están siendo optimizadas constantemente por estudiantes, profesores y expertos externos para reemplazar a los libros convencionales en papel. Estas alternativas de libros de texto gratuitos se organizan dentro de un entorno central que está integrado tanto vertical (desde el nivel avanzado hasta el nivel básico) como horizontalmente (en diferentes campos).

Las bibliotecas de LibreTexts son [Desarrollado por MindTouch®](#) y cuentan con el apoyo del Proyecto Piloto de Libros de Texto Abiertos del Departamento de Educación, la Oficina del Rector de UC Davis, la Biblioteca de UC Davis, el Programa de Soluciones de Aprendizaje Asequible de la Universidad Estatal de California y Merlot. Este material se basa en el trabajo apoyado por la Fundación Nacional de Ciencias bajo el número de subvención 1246120, 1525057 y 1413739. A menos que se indique lo contrario, el contenido de LibreTexts [CC POR-NC-SA 3.0](#).

Todas las opiniones, hallazgos y conclusiones o recomendaciones expresadas en este material son de los autores y no reflejan necesariamente los puntos de vista de la Fundación Nacional de Ciencias ni del Departamento de Educación de los EE. UU.

¿Tiene preguntas o comentarios? Para obtener información sobre adopciones o adaptaciones, póngase en contacto con info@LibreTexts.org. Puede encontrar más información sobre nuestras actividades en Facebook (<https://facebook.com/Libretexts>), Twitter (<https://twitter.com/libretexts>) o nuestro blog (<http://Blog.Libretexts.org>).



This page titled [Página de Información](#) is shared under a [not declared](#) license and was authored, remixed, and/or curated by [Joaquín López Herraiz](#).

Tabla de Contenido

El objetivo de este libro es mostrar la conexión existente entre los fundamentos de la inteligencia artificial y los contenidos de asignaturas como Cálculo Multivariable y Cálculo Diferencial de los grados de Física y Matemáticas.

- [1: Introducción a la Inteligencia Artificial y las Redes Neuronales](#)
Introducción al campo de la inteligencia artificial y las redes neuronales
- [2: Introducción al Cálculo Multivariable](#)
Introducción al Cálculo Multivariable.
- [3: Composición de Funciones. Regla de la Cadena](#)
Composición de funciones y regla de la cadena.
- [4: Aproximación Lineal de una Función Escalar de Varias Variables](#)
Aproximación lineal de funciones
- [5: Optimización y Método del Gradiente Descendiente](#)
Optimización y algoritmo de gradiente descendiente.
- [6: Redes Neuronales](#)
Redes Neuronales. Perceptrón y Perceptrón Multicapa
- [7: Retropropagación](#)
Algoritmo de retropropagación para entrenar redes neuronales
- [8: Entrenamiento de una Red Neuronal](#)
Entrenamiento de una red neuronal

This page titled [Tabla de Contenido](#) is shared under a [not declared](#) license and was authored, remixed, and/or curated by [Joaquín López Herraiz](#).

1: Introducción a la Inteligencia Artificial y las Redes Neuronales

1.1. Introducción General

1.2. Perspectiva Histórica

1.3. Aplicaciones presentes y futuras

1.4. Objetivos del Curso

This page titled [1: Introducción a la Inteligencia Artificial y las Redes Neuronales](#) is shared under a [not declared](#) license and was authored, remixed, and/or curated by [Joaquín López Herraiz](#).

1.1: Introducción General

This page is a draft and is under active development.

La inteligencia artificial, en especial el aprendizaje automático con redes neuronales, ha experimentado una importante revolución en los últimos tiempos. Esto se ha debido al gran aumento de la cantidad de datos disponibles, el incremento de la capacidad de cálculo de los ordenadores (especialmente con el uso de tarjetas gráficas para cálculo científico), así como importantes mejoras en los algoritmos.

Estos avances están abriendo nuevas posibilidades en muchas áreas, tanto científicas como tecnológicas. Por ejemplo, en el campo de la física de partículas se están empleando para mejorar los análisis de datos en laboratorios como el CERN, también se están aplicando en el análisis de datos astrofísicos, y en el ámbito de la física médica, han demostrado llegar a igualar e incluso superar a expertos en tareas de segmentación y reconocimientos de lesiones en imágenes. También en el campo de la robótica, imitando comportamientos humanos, o en las entrañas de sistemas de recomendación y bots.

Todo esto hace que muchas empresas y proyectos de investigación estén estudiando cómo trasladar y aplicar estas nuevas potentes herramientas a su campo, y haya una importante demanda de expertos en estos temas en el mercado laboral.

A pesar de estos avances, estas materias están solo empezando a formar parte integral de la formación de los estudiantes de física o matemáticas y en general, se circunscriben al ámbito de cursos avanzados y/o de especialización. Sin embargo, el conocer sus fundamentos puede ayudar a los estudiantes a tener una formación más adaptada a las demandas actuales del mercado de trabajo, así como ver aplicaciones prácticas de muchos de los contenidos que se tratan en asignaturas como Cálculo Multivariable y Cálculo Diferencial.

This page titled [1.1: Introducción General](#) is shared under a [not declared](#) license and was authored, remixed, and/or curated by [Joaquín López Herraiz](#).

1.2: Perspectiva Histórica

Como sucede en muchas disciplinas, existen diversas interpretaciones sobre el instante en el que se puede determinar el origen de la inteligencia artificial (IA), y las técnicas que se engloban dentro del aprendizaje automático. Esto se debe a que depende en cierta medida de la definición concreta que se use para definir IA o aprendizaje automático, y que se apoya en distintas disciplinas previas como la lógica matemática, la filosofía, la computación, neurociencia... Una lista de sus antecedentes históricos se puede consultar, por ejemplo en [Wikipedia](#).

En cualquier caso, se puede considerar que Inteligencia Artificial fue un concepto principalmente filosófico hasta mediados del siglo XX. En 1950, Alan Turing propuso lo que se conoce como [prueba de Turing](#) una especie de «juego de imitación» en el que participan dos personas y una computadora. Un interrogador plantea preguntas y debe determinar si las respuestas que reciben proceden de una persona o una computadora. Si la computadora es capaz de dar respuestas tan "humanas" que no permitan identificarlas como provenientes de una máquina, según Turing, es inteligente. Éste enfoque ofrece una respuesta sobre la cuestión de cómo estudiar la inteligencia artificial, pero no ofrece las herramientas sobre cómo llegar a crear ese tipo de inteligencia.

En 1956, ya apareció el término "Inteligencia Artificial" en una conferencia en Dartmouth (New Hampshire) y se plantearon unos objetivos muy ambiciosos a corto plazo, que no se cumplieron. Tal como se ha visto posteriormente, la posibilidad de crear este tipo de máquinas inteligentes requiere de una capacidad de cálculo muy lejos de lo que permitían la tecnología de aquella época.

En 1973, la falta de resultados hizo que los gobiernos de EEUU e Inglaterra dejaran de financiar proyectos de IA, y esa falta de financiación se conoce como el "Invierno de la IA". Al comienzo de los años 80, el gobierno de Japón impulsó un ambicioso plan con miles de millones de dólares de inversión en IA, pero a finales de los años 80, de nuevo la falta de los resultados esperados hizo que se volviera a retirar ese tipo de financiación.

Hubo que esperar al siglo XXI para que las técnicas de aprendizaje automático pudieran ser aplicadas con éxito en muchos problemas académicos y de la industria. Los nuevos métodos que se han ido desarrollando en las últimas dos décadas, junto con la capacidad de cómputo y de memoria de las computadoras actuales ha permitido por fin que la IA tenga un gran impacto en la sociedad.

This page titled [1.2: Perspectiva Histórica](#) is shared under a [not declared](#) license and was authored, remixed, and/or curated by [Joaquín López Herraiz](#).

1.3: Aplicaciones presentes y futuras

This page is a draft and is under active development.

En la actualidad, el uso de la IA y más concretamente, algoritmos y métodos de aprendizaje automático (incluyendo redes neuronales y aprendizaje profundo) se encuentran muy extendidos, y están siendo usados en la gran mayoría de disciplinas científicas y actividades de la industria.

En muchos casos, todas estas técnicas lo que permiten es realizar un análisis de datos y de la información de un modo mucho más general que lo que se venía usando. Por ejemplo, permiten combinar datos heterogéneos como combinaciones de imágenes y datos numéricos, videos y señales de sensores.. En otros casos, su principal atractivo es eliminar la necesidad de disponer de un modelo teórico previo sobre un fenómeno para poder estudiarlo.

Esto está llevando a un cambio de paradigma en ciencia desde lo que se conoce como "model-driven" a lo que se conoce como "data-driven". Es decir, con la IA se da más importancia a los datos y a lo que se pueda determinar a partir de ellos, que a generar un modelo teórico sencillo y explicable con ellos. Este cambio ofrece importantes ventajas prácticas, pero por supuesto, esto no está exento de polémica y riesgos, y plantea una serie de importantes cuestiones tanto prácticas como filosóficas. Por ejemplo:

- ¿Cómo reconciliar los métodos "clásicos" con los que se obtienen con IA?
- ¿Hasta qué punto debemos seguir usando y desarrollando los métodos previos?
- ¿Cómo lograr "aprender" (en el sentido de generar un modelo que explique un fenómeno y que se pueda formular de manera inteligible para los humanos) de lo que ha "aprendido" una IA?

This page titled [1.3: Aplicaciones presentes y futuras](#) is shared under a [not declared](#) license and was authored, remixed, and/or curated by [Joaquín López Herraiz](#).

1.4: Objetivos del Curso

El objetivo de este curso es ofrecer los fundamentos de cálculo matemático en los que se basan las redes neuronales, que han sido la principal herramienta de la IA que ha permitido abarcar problemas complejos.

Existen muchos cursos sobre IA, enfocados principalmente a la programación y el uso de estas nuevas herramientas. También existen algunos cursos en inglés en otras universidades sobre las matemáticas de la IA, que en algunos casos incluyen también contenidos sobre estadística y otras disciplinas que también juegan un importante papel en la IA.

En nuestro caso, nos centraremos en las cuestiones referidas al cálculo, optimización de las redes neuronales, enfocado principalmente a cómo determinar los coeficientes de una red neuronal (lo que se conoce como "entrenamiento de la red neuronal") a partir de una serie de medidas.

This page titled [1.4: Objetivos del Curso](#) is shared under a [not declared](#) license and was authored, remixed, and/or curated by [Joaquín López Herraiz](#).

2: Introducción al Cálculo Multivariante

Presentamos inicialmente algunos conceptos como los de función multivariante, derivada parcial, gradiente, así como la composición de funciones de varias variables y la regla de la cadena para su derivación. Todos ellos serán imprescindibles en el desarrollo subsiguiente, en el que se introduce un algoritmo de aprendizaje supervisado muy utilizado para entrenar redes neuronales. Nos valdremos para ello del método del gradiente descendente, en cuya aplicación efectiva se emplea la estrategia de retropropagación, indicada para el cálculo del gradiente de funciones mediante la aplicación recursiva de la regla de la cadena.

Definición 2.1

Definición 1. Una función $\mathbf{f}: \mathbb{R}^n \rightarrow \mathbb{R}^k$ lleva $\mathbf{x} \in \mathbb{R}^n$ en $\mathbf{y} = \mathbf{f}(\mathbf{x}) \in \mathbb{R}^k$. Se dice que \mathbb{R}^n es el dominio de \mathbf{f} y para cada $\mathbf{x} \in \mathbb{R}^n$, el vector $\mathbf{y} = \mathbf{f}(\mathbf{x})$ es la imagen de \mathbf{x} por \mathbf{f} . El conjunto de todas las imágenes se denomina también imagen de \mathbf{f} .

- $\mathbf{x} = (x_1, \dots, x_n)^T$.
- $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \dots, f_k(\mathbf{x}))^T$ y cada $f_i(\mathbf{x})$ es una función coordenada de \mathbf{f} .
- Si $k = 1$, $f(\mathbf{x}) \in \mathbb{R}$.

Observación 2.1

Los vectores $\mathbf{x} \in \mathbb{R}^n$ son vectores columna. Por comodidad, usamos la notación $\mathbf{x} = (x_1, \dots, x_n)^T$ para indicar el traspuesto del vector fila (x_1, \dots, x_n) . En ocasiones preferiremos la notación $\mathbf{f}(x, y)$, que resulta más simple que $\mathbf{f}((x, y)^T)$.

Ejemplo 2.1

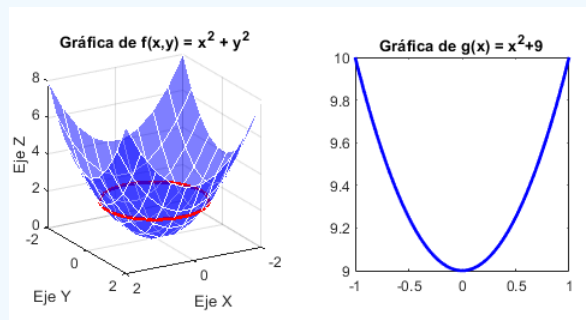
Funciones Lineales

Para números reales fijos a_1, a_2, a_3 , la función lineal f de dominio \mathbb{R}^3 dada por $f(x_1, x_2, x_3) = a_1x_1 + a_2x_2 + a_3x_3$ cumple que fijadas dos de las variables.

Por ejemplo $x_1 = -3$ y $x_2 = 5$, la función $g(x_3) = -3a_1 + 5a_2 + a_3x_3$ es una función lineal.

Ejemplo 2.2

Consideramos la función $f(x, y) = x^2 + y^2$, con dominio \mathbb{R}^2 e imagen en \mathbb{R} . El valor mínimo que toma la función es 0, en el punto $(0, 0)$. En todos los puntos (x, y) que estén sobre la circunferencia centrada en $(0, 0)$ y radio $r > 0$, la función f toma el valor r^2 . Si fijamos una de las variables, por ejemplo $y = 3$, obtenemos la función $g(x) = x^2 + 9$, cuya gráfica es una parábola. Podemos representar la función f , que es la superficie de \mathbb{R}^3 de la gráfica siguiente:

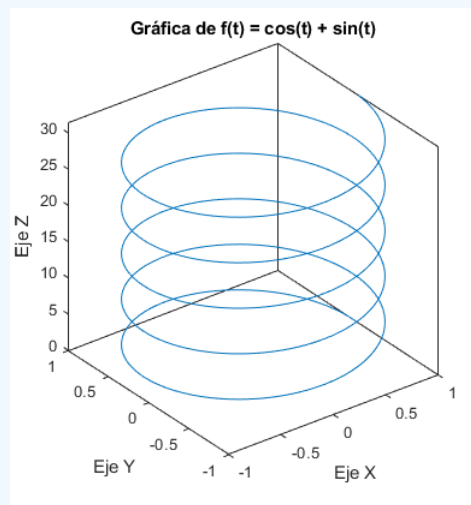


Observación 2.2

Cuando trabajemos con una función $f: \mathbb{R}^n \rightarrow \mathbb{R}$ y fijemos $n - 1$ variables, obtenemos una función con dominio e imagen en \mathbb{R} , que podemos analizar con las técnicas ya conocidas.

✓ Ejemplo 2.3

La función $\mathbf{f}: \mathbb{R} \rightarrow \mathbb{R}^2$ dada por $\mathbf{f}(t) = (\cos(t), \sin(t))$ viene definida por las funciones coordenadas $f_1(t) = \cos(t)$ y $f_2(t) = \sin(t)$. La gráfica de \mathbf{f} es una hélice contenida en \mathbb{R}^3 . La imagen de \mathbf{f} es el conjunto $C := \{(x, y) \in \mathbb{R}^2 : x^2 + y^2 = 1\}$, ya que para todo $t \in \mathbb{R}$ se cumple que $\cos^2(t) + \sin^2(t) = 1$.



✓ Ejemplo 2.4

Nos encontraremos con la función sigmoide $\sigma(x) = \frac{1}{1 + e^{-x}}$, que es derivable y (¡Compruébalo!) su derivada es $\sigma'(x) = \sigma(x)(1 - \sigma(x))$

Revisamos para una función $f: \mathbb{R} \rightarrow \mathbb{R}$ (es decir: $n = k = 1$), la noción de derivada en un punto x .

✎ Definición 2.2

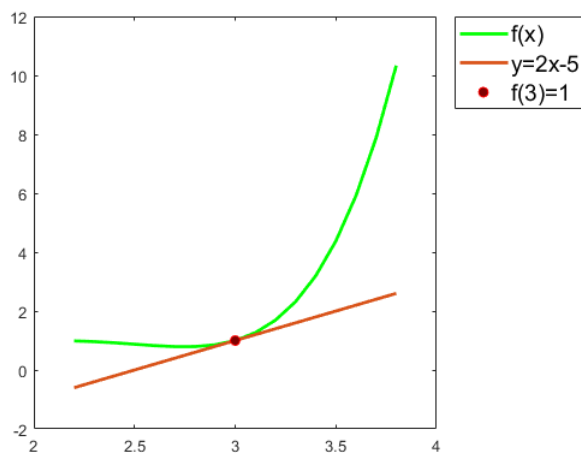
La derivada de f en x , que notamos $f'(x)$, viene dada (¡Cuando existe!) por:

$$\lim_{h \rightarrow 0} \frac{f(x+h) - f(x)}{h} \quad (2.1)$$

Recuerda que $f'(x)$ es la pendiente de la recta que más se asemeja a la gráfica de la función entre las que pasan por el punto $(x, f(x))$. Como la recta que une los puntos (a, b) y (c, d) tiene pendiente $\frac{d-b}{c-a}$, la recta que une $(x, f(x))$ con $(x+h, f(x+h))$ tiene como pendiente $\frac{f(x+h) - f(x)}{h}$, por lo que $f'(x)$ es el límite de las pendientes de estas rectas.

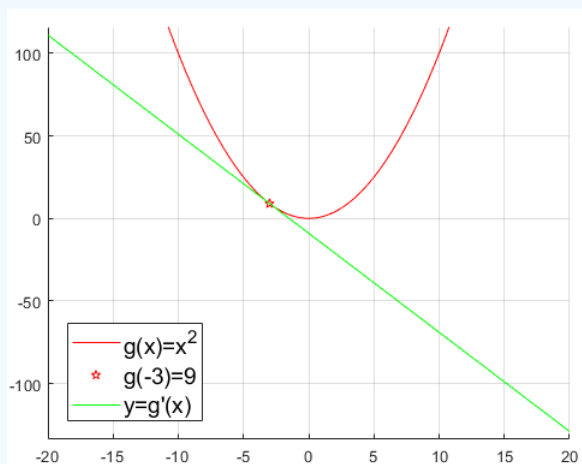
✓ Ejemplo 2.5

Si $f: \mathbb{R} \rightarrow \mathbb{R}$ cumple que $f(3) = 1$ y $f'(3) = 2$, el valor $f(3)$ no puede ser mínimo (ni máximo), ya que la gráfica cerca de 3 es similar a la recta $y = f(3) + f'(3)(x - 3) = 1 + 2(x - 3)$; es decir, $y = 2x - 5$.



✓ Ejemplo 2.6

Si pensamos en la función $g(x) = x^2$ y $x = -3$, como $g'(-3) = -6$, a la derecha de -3 la función tomará valores menores que $g(-3) = 9$. El valor mínimo de la función, que es $g(0) = 0$, se alcanza en un punto donde la derivada es nula.



Para comprender cómo varía una función $f : \mathbb{R}^n \rightarrow \mathbb{R}$ que depende de varias variables empezamos fijando todas las variables salvo una. Consideramos la función resultante (que tiene dominio e imagen en \mathbb{R}). Para el ejemplo $f(x, y) = x^2 + y^2$, fijada la variable y obtenemos una función $g_1(x) = x^2 + y^2$, que es decreciente para $x < 0$ y creciente para $x > 0$, ver la figura del ejemplo Ejemplo 2 (se da lo análogo para $g_2(y) = x^2 + y^2$, con x fijo). Nos centramos, por ejemplo, en el punto del plano $(-3, 4)$, para el que $f(-3, 4) = 25$, queremos encontrar valores h_1 y h_2 para los que $f(-3 + h_1, 4 + h_2) < 25$. Podemos conseguirlo siempre $-3 < -3 + h_1 < 0$ y $0 < 4 + h_2 < 4$. Mientras la derivada $g'_1(-3) < 0$ nos alienta a elegir $h_1 > 0$, la derivada $g'_2(4) > 0$ nos impulsa a elegir $h_2 < 0$. La misma estrategia podrá ser utilizada para buscar mínimos de funciones escalares acudiendo a las nociones de derivada parcial y gradiente de la función $f : \mathbb{R}^n \rightarrow \mathbb{R}$. La derivada parcial con respecto a una variable x_j reflejará la tasa de variación de f en la dirección de ese eje.

Definición 2.3

La derivada parcial de la función $f : \mathbb{R}^n \rightarrow \mathbb{R}$ con respecto a la variable x_j en $\mathbf{x} = (x_1, \dots, x_{j-1}, x_j, x_{j+1}, \dots, x_n)^T$, que denotamos $\frac{\partial}{\partial x_j} f(\mathbf{x})$, viene dada por

$$\lim_{h \rightarrow 0} \frac{f(x_1, \dots, x_{j-1}, x_j + h, x_{j+1}, \dots, x_n) - f(x_1, \dots, x_{j-1}, x_j, x_{j+1}, \dots, x_n)}{h} \quad (2.2)$$

✓ Ejemplo 2.7

La función $f(x, y) = x^2 + y^2$ tiene como derivadas parciales $\frac{\partial}{\partial x} f(x, y) = 2x$ y $\frac{\partial}{\partial y} f(x, y) = 2y$. La función $f(x_1, x_2, x_3) = a_1 x_1 + a_2 x_2 + a_3 x_3$ tiene como derivada parcial con respecto a x_j el número a_j , para $1 \leq j \leq 3$. La función $f(x, y) = \sin(xy^3) + x^2 y$ tiene como derivadas parciales $\frac{\partial}{\partial x} f(x, y) = y^3 \cdot \cos(xy^3) + 2xy$ y $\frac{\partial}{\partial y} f(x, y) = 3xy^2 \cos(xy^3) + x^2$.

✎ Definición 2.4

Dada una función $f: \mathbb{R}^n \rightarrow \mathbb{R}$ con derivadas parciales $\frac{\partial}{\partial x_j} f(\mathbf{x})$, el gradiente de f se denota por $\nabla f(\mathbf{x})$ y tiene por coordenadas las funciones $\frac{\partial}{\partial x_j} f(\mathbf{x})$, con $1 \leq j \leq n$.

✓ Ejemplo 2.8

La función $f(x, y) = x^2 + y^2$ tiene como gradiente la función $\nabla f(x, y) = (2x, 2y)^T$. La función $f(x_1, x_2, x_3) = a_1 x_1 + a_2 x_2 + a_3 x_3$ tiene como gradiente el vector constante $\nabla f(\mathbf{x}) = (a_1, a_2, a_3)^T$.

Observación 2.3

Como veremos más adelante, el hecho de que la dirección del gradiente $\nabla f(\mathbf{x})$ sea la de máximo crecimiento de la función f en cada punto \mathbf{x} y la dirección dada por su opuesto, $-\nabla f(\mathbf{x})$, la de mayor decrecimiento, permite establecer una estrategia para minimizar la función escalar $f(\mathbf{x})$. Para la función $f(x, y) = x^2 + y^2$ en el punto $(-3, 4)$ hemos elegido valores (h_1, h_2) de signos opuestos al gradiente $(-6, 8)$ para obtener puntos $(-3 + h_1, 4 + h_2)$ en los que f toma valores menores que $f(-3, 4) = 25$.

Si trabajamos con una función $\mathbf{f}: \mathbb{R}^n \rightarrow \mathbb{R}^k$ de k coordenadas f_1, f_2, \dots, f_k , cuando existan las n derivadas parciales de cada una de ellas podemos considerar la función matricial de k filas y n columnas formada por los k gradientes ∇f_k . Se trata de la *matriz jacobiana* de la función \mathbf{f} .

This page titled [2: Introducción al Cálculo Multivariente](#) is shared under a [not declared](#) license and was authored, remixed, and/or curated by [Joaquín López Herraiz](#).

3: Composición de Funciones. Regla de la Cadena

Si la imagen de una función \mathbf{f} está contenida en el dominio de una función \mathbf{g} , puede aplicarse \mathbf{g} a los valores $\mathbf{f}(\mathbf{x})$ y obtener la composición $\mathbf{g} \circ \mathbf{f}$.

Por ejemplo, $\mathbf{f}: \mathbb{R} \rightarrow \mathbb{R}^2$ dada por $\mathbf{f}(t) = (\cos(t), \sin(t))$ puede componerse con la función $g(x, y) = x^2 + y^2$ o con $g(x, y) = \sin(xy^3) + x^2y$, obteniendo, respectivamente, las funciones $g \circ \mathbf{f}: \mathbb{R} \rightarrow \mathbb{R}$ dadas por $(g \circ \mathbf{f})(t) = g(\cos(t), \sin(t)) = \cos^2(t) + \sin^2(t) \equiv 1$ y $(g \circ \mathbf{f})(t) = g(\cos(t), \sin(t)) = \sin(\cos(t) \sin^3(t)) + \cos^2(t) \sin(t)$. Para las funciones $\mathbf{f}(x_1, x_2, x_3) = a_1x_1 + a_2x_2 + a_3x_3$ y $\mathbf{h}(t) = (\cos(t), \sin(t))$ podemos considerar la composición $\mathbf{h} \circ \mathbf{f}: \mathbb{R}^3 \rightarrow \mathbb{R}^2$ que se expresa por

$$(\mathbf{h} \circ \mathbf{f})(x_1, x_2, x_3) = \mathbf{h}(a_1x_1 + a_2x_2 + a_3x_3) = (\cos(a_1x_1 + a_2x_2 + a_3x_3), \sin(a_1x_1 + a_2x_2 + a_3x_3)) \quad (3.1)$$

pero no $\mathbf{f} \circ \mathbf{h}$, ya que la imagen de \mathbf{h} está contenida en \mathbb{R}^2 y el dominio de \mathbf{f} es \mathbb{R}^3 . En general, consideramos composiciones de funciones $\mathbf{g} \circ \mathbf{f}$, donde $\mathbf{f}: \mathbb{R}^n \rightarrow \mathbb{R}^k$ y $\mathbf{g}: \mathbb{R}^k \rightarrow \mathbb{R}^m$; analizamos las m coordenadas de la composición y, para cada una de ellas, estudiamos las respectivas derivadas parciales con respecto a las n variables. Por ello, en los ejemplos nos centraremos en el caso $m = 1$.

Regla de la Cadena

Dadas dos funciones derivables f y g con dominio e imagen en \mathbb{R} y su composición $g \circ f$, se calcula la derivada de la función $g \circ f$ a través de la denominada regla de la cadena. En concreto, $(g \circ f)'(x) = g'(f(x))f'(x)$.

✓ Ejemplo 3.9

La función $f(x) = x^2$ se compone con la función $g(x) = e^x$ y da lugar a $(g \circ f)(x) = e^{x^2}$, cuya derivada es $(g \circ f)'(x) = e^{x^2} 2x$. Análogamente, $(f \circ g)(x) = (e^x)^2$ (de hecho, $(f \circ g)(x) = e^{2x}$) y por la regla de la cadena $(f \circ g)'(x) = 2e^x e^x = 2e^{2x}$.

Veamos cómo operar en el caso en que las funciones tengan dominio o imagen en \mathbb{R}^d , con $d > 1$.

- $\mathbf{f}: \mathbb{R}^2 \rightarrow \mathbb{R}$ y $g: \mathbb{R} \rightarrow \mathbb{R}$. Utilizamos las variables s, t para \mathbf{f} y la variable x para g . Entonces $y = g(x) = (g \circ \mathbf{f})(s, t) := F(s, t)$ y

$$\frac{\partial}{\partial s} F = \frac{d}{dx} g \frac{\partial}{\partial s} f \quad (3.2)$$

Análogamente,

$$\frac{\partial}{\partial t} F = \frac{d}{dx} g \frac{\partial}{\partial t} f \quad (3.3)$$

✓ Ejemplo 3.10

Si $\mathbf{f}(s, t) = s(1 - t^3)$ y $g(x) = e^x$, la función $F(s, t) = (g \circ \mathbf{f})(s, t) = g(s(1 - t^3)) = e^{s(1 - t^3)}$ tiene como gradiente $\nabla F(s, t) = (e^{s(1 - t^3)}(1 - t^3), e^{s(1 - t^3)}s(-3t^2))^T$.

- $\mathbf{f} = (f_1, f_2)^T: \mathbb{R}^2 \rightarrow \mathbb{R}^2$ y $g: \mathbb{R}^2 \rightarrow \mathbb{R}$. Utilizamos las variables s, t para \mathbf{f} y las variables x, y para g . Entonces $z = g(x, y) = (g \circ \mathbf{f})(s, t) := F(s, t)$ y

$$\frac{\partial}{\partial s} F = \frac{\partial}{\partial x} g \frac{\partial}{\partial s} f_1 + \frac{\partial}{\partial y} g \frac{\partial}{\partial s} f_2 \quad (3.4)$$

Análogamente,

$$\frac{\partial}{\partial t} F = \frac{\partial}{\partial x} g \frac{\partial}{\partial t} f_1 + \frac{\partial}{\partial y} g \frac{\partial}{\partial t} f_2 \quad (3.5)$$

✓ Ejemplo 3.11

Si $(x, y) = \mathbf{f}(s, t) = (s^2t, s(4 - 3t))$, $g(x, y) = \cos(x - 2y)$ y F es la función $F(s, t) := (g \circ \mathbf{f})(s, t) = g(s^2t, s(4 - 3t)) = \cos(s^2t - 2s(4 - 3t))$ admite derivadas parciales. Calcula directamente la derivada parcial con respecto a cada una de las variables y hazlo usando la regla de la cadena multivariante para ver que coinciden.

- Si $\mathbf{f} = (f_1, \dots, f_k)^T : \mathbb{R}^n \rightarrow \mathbb{R}^k$ y $\mathbf{g} : \mathbb{R}^k \rightarrow \mathbb{R}$ admiten derivadas parciales, las de la función $F = \mathbf{g} \circ \mathbf{f}$ vienen dadas, para cada $1 \leq j \leq n$, por

$$\frac{\partial}{\partial x_j} F = \sum_{i=1}^k \frac{\partial}{\partial x_i} g \frac{\partial}{\partial x_j} f_i \quad (3.6)$$

✓ Ejemplo 3.12

La composición de $\mathbf{f}(s, t) = (e^{s-t}, \cos(s^2t))$ y $g(x, y) = (x^2, x - y^3)$ da lugar a la función $F(s, t) = ((e^{s-t})^2, e^{s-t} - (\cos(s^2t)))$. Utiliza de nuevo la doble vía para comprobar la fórmula de las derivadas parciales en este caso.

This page titled [3: Composición de Funciones. Regla de la Cadena](#) is shared under a [not declared](#) license and was authored, remixed, and/or curated by [Joaquín López Herraiz](#).

4: Aproximación Lineal de una Función Escalar de Varias Variables

This page is a draft and is under active development.

Recordemos que, bajo ciertas condiciones de derivabilidad, una función real puede aproximarse por una recta cuando se producen cambios pequeños (ε) en la variable independiente (**aproximación por la recta tangente**, figura 5),

$$f(x + \varepsilon) \approx f(x) + \varepsilon f'(x) \quad (4.1)$$

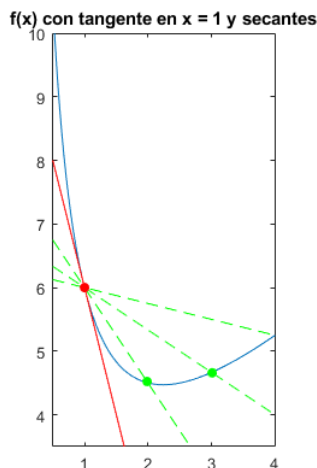


Fig. 5 - Aproximación de una función por su recta tangente

✓ Ejemplo 4.13

La función $f(x) = x^2$ puede aproximarse, en puntos cerca de $x = 1$ por la recta:

$$f(1 + \varepsilon) \approx f(1) + f'(1)\varepsilon = 1 + 2\varepsilon \quad (4.2)$$

Del mismo modo, para una función $f : \mathbb{R}^n \rightarrow \mathbb{R}$ que depende de varias variables, tendremos:

$$f(x_1 + \varepsilon_1, x_2 + \varepsilon_2, \dots, x_n + \varepsilon_n) \approx f(x_1, x_2, \dots, x_n) + \sum_{i=1}^n \varepsilon_i \frac{\partial f}{\partial x_i}(x_1, x_2, \dots, x_n) \quad (4.3)$$

Esta expresión puede escribirse de forma más compacta usando la definición de gradiente (Definición 4) y llamando $\varepsilon = (\varepsilon_1, \varepsilon_2, \dots, \varepsilon_n)$, vector fila donde se agrupan los pequeños cambios que se producen en cada una de las variables independientes:

$$f(\mathbf{x} + \varepsilon) \approx f(\mathbf{x}) + \varepsilon \nabla f(\mathbf{x}) \quad (4.4)$$

En este caso, si tenemos $n = 2$, aproximamos la función cerca de un punto, por su plano tangente.

✓ Ejemplo 4.14

La función $f(x, y) = x^2 + xy + y^2$, en un entorno de $(1, 0)$, tomará valores cercanos a la aproximación de primer orden. Para calcularla necesitamos $f(1, 0)$ y $\nabla f(1, 0)$ que valen

- $f(1, 0) = 1$
- $\nabla f(1, 0) = (2x + y, x + 2y) |_{x=1, y=0} = (2, 1)$

y se emplean para determinar el plano tangente en el punto $(1, 0, f(1, 0))$, cuya ecuación es $2x + y = 2$. El vector normal al plano es $\nabla f(1, 0)$.

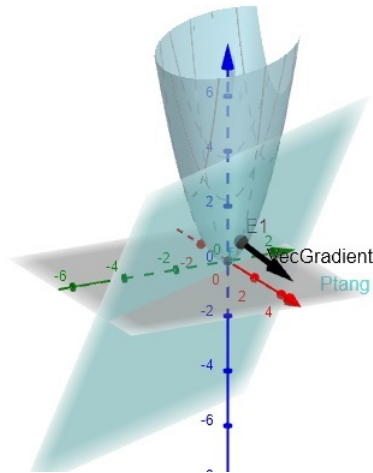


Fig. 6 - Aproximación de una función por su plano tangente

This page titled [4: Aproximación Lineal de una Función Escalar de Varias Variables](#) is shared under a [not declared](#) license and was authored, remixed, and/or curated by [Joaquín López Herraiz](#).

5: Optimización y Método del Gradiente Descendiente

Vamos a describir el método del gradiente descendente, que sirve para encontrar un punto donde la función $f: \mathbb{R}^n \rightarrow \mathbb{R}$ alcanza un valor mínimo.

El proceso se puede resumir en:

- Comenzamos eligiendo un valor arbitrario inicial de \mathbf{x} .
 - Buscamos la dirección \mathbf{v} en la que f decrece más rápidamente alrededor de \mathbf{x} . A \mathbf{v} le llamamos *dirección de máximo descenso*.
 - Variamos “un poco” los valores iniciales en la dirección \mathbf{v} :
- $$\mathbf{x} \rightarrow \mathbf{x} + \varepsilon \mathbf{v} \quad (5.1)$$
- Repetimos.

Nos detenemos ahora en el segundo paso de este proceso: cómo determinar la dirección de descenso más rápido. Como se ha visto en la sección anterior:

$$f(\mathbf{x} + \mathbf{v}) \approx f(\mathbf{x}) + \mathbf{v} \nabla f(\mathbf{x}). \quad (5.2)$$

Por la definición de producto escalar, suponiendo que \mathbf{v} es un vector de módulo 1, la expresión anterior se escribe

$$f(\mathbf{x} + \mathbf{v}) \approx f(\mathbf{x}) + \mathbf{v} \nabla f(\mathbf{x}) = f(\mathbf{x}) + \|\nabla f(\mathbf{x})\| \cos(\theta), \quad (5.3)$$

donde θ es el ángulo que forman los vectores \mathbf{v} y $\nabla f(\mathbf{x})$.

Con vistas a que f decrezca lo más rápido posible, debemos elegir θ de forma que $\cos(\theta)$ sea lo menor posible, esto es, $\cos(\theta) = -1$.

Esto ocurre cuando θ vale π (180°) o, lo que es lo mismo, si \mathbf{v} tiene el sentido opuesto a $\nabla f(\mathbf{x})$.

El proceso queda reescrito como:

- Comenzamos eligiendo un valor arbitrario inicial de \mathbf{x} .
 - La dirección \mathbf{v} que hace descender más rápidamente a f alrededor de \mathbf{x} es $-\nabla f(\mathbf{x})$. Calculamos $\nabla f(\mathbf{x})$.
 - Variamos un poco los valores iniciales en la dirección \mathbf{v} :
- $$\mathbf{x} \rightarrow \mathbf{x} - \varepsilon \nabla f(\mathbf{x}) \quad (5.4)$$
- Repetimos.

Para aplicar este algoritmo iterativo es necesario fijar dos parámetros: el número de iteraciones y el valor de ε (que se conoce como *tasa de aprendizaje*).

- Se dirá que el algoritmo converge si la sucesión de soluciones en el proceso iterativo se acerca a la solución óptima (esto es, el mínimo global). La cantidad de iteraciones que se necesita para que el método del gradiente descendente converja puede variar mucho (dependiendo del problema, desde unas pocas iteraciones hasta miles de ellas). En cualquier caso, la tasa de aprendizaje condicionará el número mínimo de iteraciones necesario para la convergencia.
- La tasa de aprendizaje ε es una constante, con $0 < \varepsilon < 1$, que afectará de forma significativa la efectividad del algoritmo: un valor demasiado pequeño puede llevar a que el algoritmo precise un número elevado de iteraciones antes de converger; mientras que si el valor asignado es demasiado grande, puede saltarse el mínimo global (pasando a tomar valores mayores) y no proporcionar una buena solución.

This page titled 5: Optimización y Método del Gradiente Descendiente is shared under a [not declared](#) license and was authored, remixed, and/or curated by Joaquín López Herraiz.

CHAPTER OVERVIEW

6: Redes Neuronales

This page is a draft and is under active development.

[6.1: Perceptrón](#)

[6.2: Funciones de Pérdida](#)

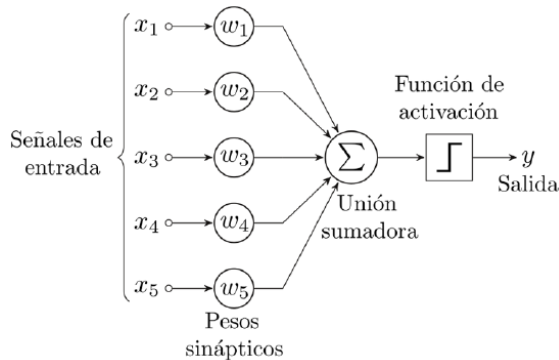
[6.3: Perceptrón Multicapa](#)

This page titled [6: Redes Neuronales](#) is shared under a [not declared](#) license and was authored, remixed, and/or curated by [Joaquín López Herraiz](#).

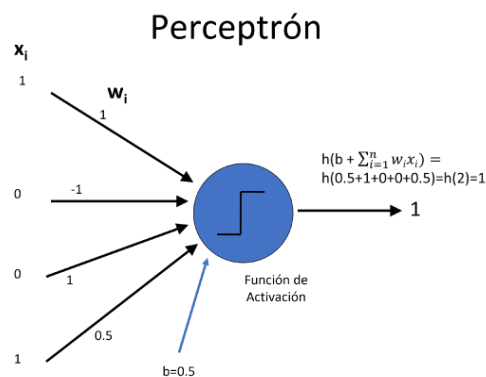
6.1: Perceptrón

El perceptrón fue propuesto en 1958 por [Frank Rosenblatt](#) en un proyecto del [laboratorio de aeronáutica de la Universidad de Cornell](#) financiado por la oficina de investigación naval de los EEUU. Se planteó inicialmente como una máquina, más que un programa o un algoritmo, dado que los pesos que se iban optimizando se actualizaban con motores eléctricos y potenciómetros.

En la actualidad, se considera el perceptrón como la unidad fundamental de una red neuronal, siendo el esquema principal del perceptrón el que se muestra a continuación:



Es decir, un perceptrón toma una serie de señales de entrada x , a las que multiplica por unos pesos w y obtiene la suma total de ese resultado. A ese valor se le aplica una función de activación y finalmente se obtiene un valor de salida. Ese valor de salida puede, a su vez, ser la entrada de otro perceptrón posterior. Un ejemplo de cálculo con cuatro valores de entrada, a los que se aplica una serie de pesos y un bias, y una función de activación de tipo escalón se muestra en la siguiente figura.



Existen muchas funciones de activación. Por ejemplo, una con valor de umbral b se puede definir como:

$$f\left(\sum_{j=1}^J w_j x_j\right) = \begin{cases} 1 & \text{si } \sum_{j=1}^J w_j x_j > b \\ 0 & \text{si } \sum_{j=1}^J w_j x_j \leq b \end{cases} = y$$

This page titled [6.1: Perceptrón](#) is shared under a [not declared](#) license and was authored, remixed, and/or curated by [Joaquín López Herraiz](#).

6.2: Funciones de Pérdida

This page is a draft and is under active development.

Consideramos que estamos trabajando con aprendizaje supervisado, en el que se dispone de datos de referencia de entrada y salida, y nuestro objetivo es entrenar una red para que aprenda a realizar el mapeo entre ambos conjuntos. Las funciones de pérdida, funciones de coste, o funciones de error, determinan cómo cuantificamos las diferencias entre las salidas de la red neuronal (estimaciones de los valores) que denominaremos (p), y las medidas de referencia (y). El entrenamiento de la red consistirá en la minimización de esta función, que denominaremos $L[\phi]$.

Existen muchas posibles formas de definir esta función de pérdida, pero es importante seleccionar la más apropiada para cada problema, en función del tipo de datos, el problema que estemos afrontando (clasificación, regresión...) así como el tipo de error estadístico que puedan tener nuestras medidas. Una adecuada selección de la función de pérdida es fundamental para obtener una correcta solución.

Un listado completo de las funciones de pérdida se puede consultar en referencias como "[Loss Functions](#)". Algunas de las funciones de pérdida más comunes son:

1) Mean Squared Error: Es la forma más tradicional de calcular la desviación. Se calcula como:

$$MSE = \sum_i |y_i - p_i|^2$$

2) Mean Average Error: Es parecida al MSE, pero tiene la ventaja de que las desviaciones más grandes entre el valor que arroja la red neuronal y el valor esperado contribuyen menos, al no estar elevado al cuadrado.

$$MAE = \sum_i |y_i - p_i|$$

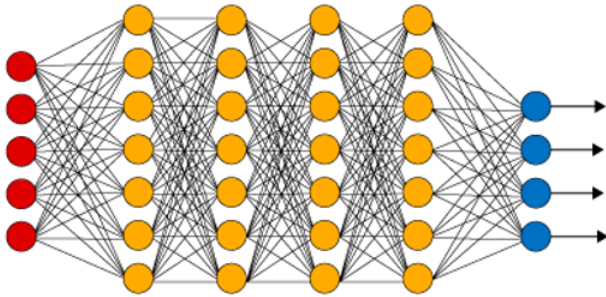
3) Binary Cross Entropy (BCE): Entropía binaria cruzada, especialmente útil en redes que se usan para clasificación binaria (en casos de clasificación multiclase, se denomina entropía cruzada multiclase). Recordemos que para este tipo de problemas se usaba la función sigmoide como función de activación, generalmente.

$$BCE = \sum_i -(1 - y_i) \log[1 - \text{sig}(f(x_i, \phi))] - y_i \log[\text{sig}(f(x_i, \phi))]$$

This page titled [6.2: Funciones de Pérdida](#) is shared under a [not declared](#) license and was authored, remixed, and/or curated by [Joaquín López Herraiz](#).

6.3: Perceptrón Multicapa

El perceptrón multicapa consiste en la concatenación de capas de perceptrones en un esquema similar al que se muestra a continuación:



En este esquema de una red neuronal del tipo perceptrón multicapa, las entradas figuran en rojo, las salidas en azul, y los distintos perceptrones en amarillo.

Como se puede observar, en este tipo de red neuronal (denominado feed-forward en inglés), los perceptrones se organizan en capas (las distintas columnas en la figura), de manera que los perceptrones o neuronas de cada capa realizan operaciones de manera simultánea sobre los datos precedentes (ya sean de entrada o de una capa previa). El alto número de capas típico de estas redes que se emplea en problemas complejos es lo que llevó a calificarlas como "redes profundas" y derivó en el concepto de "Aprendizaje profundo" o Deep Learning.

Existen muchos otros tipos de redes neuronales, que se diferencian principalmente en la forma en la que se establecen las conexiones entre los perceptrones o neuronas de la red, estableciendo conexiones recursivas, saltando una serie de capas.. El diseño de las conexiones óptimas de los perceptrones de una red es un importante campo de investigación en la actualidad.

This page titled [6.3: Perceptrón Multicapa](#) is shared under a [not declared](#) license and was authored, remixed, and/or curated by [Joaquín López Herraiz](#).

7: Retropropagación

This page is a draft and is under active development.

Idea intuitiva de la retropropagación

El algoritmo de retropropagación es un método muy utilizado en redes neuronales para minimizar la conocida como función de pérdida o de error de la red. Esta función proporciona, durante el entrenamiento de la red neuronal, una medida de cuánto se desvían los valores proporcionados por la red de los valores reales o deseados. Los parámetros desconocidos de la red, que son las incógnitas del problema y suelen denominarse pesos, se modifican de tal forma que los valores de salida de la red se ajusten lo más posible a los deseados. Para ello, se utiliza el método del gradiente descendente, que permite minimizar el error en función de los pesos. Puesto que el error se calcula al final de la red (en la capa de salida) es más fácil calcular el efecto de las variaciones de los pesos (y, por tanto, ajustarlos) comenzando desde el final, esto es, cambiando los pesos que llegan a la capa de salida. Sucesivamente, se modifican los de cada capa intermedia en dirección hacia la capa inicial. En este proceso será de utilidad la regla de la cadena.

✓ Ejemplo 7.15

Vamos a considerar las funciones escalares $f(x) = x^2$, $g(y) = \sin y$ y $h(z) = \exp(-z)$. La composición de las tres funciones $(h \circ g \circ f)(x)$ da lugar a $J(x) = \exp(-\sin x^2)$. Vamos a calcular $\frac{dJ}{dx}$ utilizando el algoritmo de retropropagación. Aunque la composición de funciones se realiza hacia delante:

$$x \rightarrow y = f(x) = x^2 \rightarrow z = g(y) = \sin y \rightarrow J(x) = h(z) = \exp(-z) \quad (7.1)$$

el cálculo de la derivada lo realizamos hacia atrás, utilizando la regla de la cadena:

1. Empezamos por $h(z) = \exp(-z)$:

$$\frac{dJ}{dz} \rightarrow -\exp(-z) \quad (7.2)$$

2. Después por $z = g(y) = \sin(y)$:

$$\frac{dJ}{dy} \rightarrow \frac{dJ}{dz} \frac{dz}{dy}, \quad \frac{dz}{dy} = \cos(y) \quad (7.3)$$

3. Por último por $y = f(x) = x^2$:

$$\frac{dJ}{dx} \rightarrow \frac{dJ}{dy} \frac{dy}{dx}, \quad \frac{dy}{dx} = 2x \quad (7.4)$$

4. Así, resulta

$$\frac{dJ}{dx}(x) = -2x \cos(x^2) \exp(-\sin(x^2)) \quad (7.5)$$

✓ Ejemplo 7.16

Sean $\mathbf{x} = (x_1, x_2)$ y las funciones: $f(x_1, x_2) = a_1 x_1 + a_2 x_2$ con a_1 y a_2 constantes, $g(y) = 1 + e^{-y}$ y $h(z) = \frac{1}{z}$. La composición de las tres funciones $(h \circ g \circ f)(x)$ da lugar a $J(x_1, x_2) = \frac{1}{1 + e^{-(a_1 x_1 + a_2 x_2)}}$. Esto es:

$$(x_1, x_2) \rightarrow y = f(x_1, x_2) = a_1 x_1 + a_2 x_2 \rightarrow z = g(y) = 1 + e^{-y} \rightarrow J(\mathbf{x}) = h(z) = \frac{1}{z} \quad (7.6)$$

El objetivo en este caso es el cálculo del gradiente de la función J , $\nabla J(\mathbf{x})$, que tiene por coordenadas $\frac{\partial}{\partial x_i} J(\mathbf{x})$, para $i = 1, 2$.

1. Empezamos por $h(z) = \frac{1}{z}$:

$$\frac{\partial J}{\partial z} \rightarrow -\frac{1}{z^2} \quad (7.7)$$

2. Después tomamos $z = g(y) = 1 + e^{-y}$:

$$\frac{\partial J}{\partial y} \rightarrow \frac{\partial J}{\partial z} \frac{\partial z}{\partial y}, \quad \frac{\partial z}{\partial y} = -e^{-y} \quad (7.8)$$

3. Por último, para $y = f(x_1, x_2) = a_1 x_1 + a_2 x_2$:

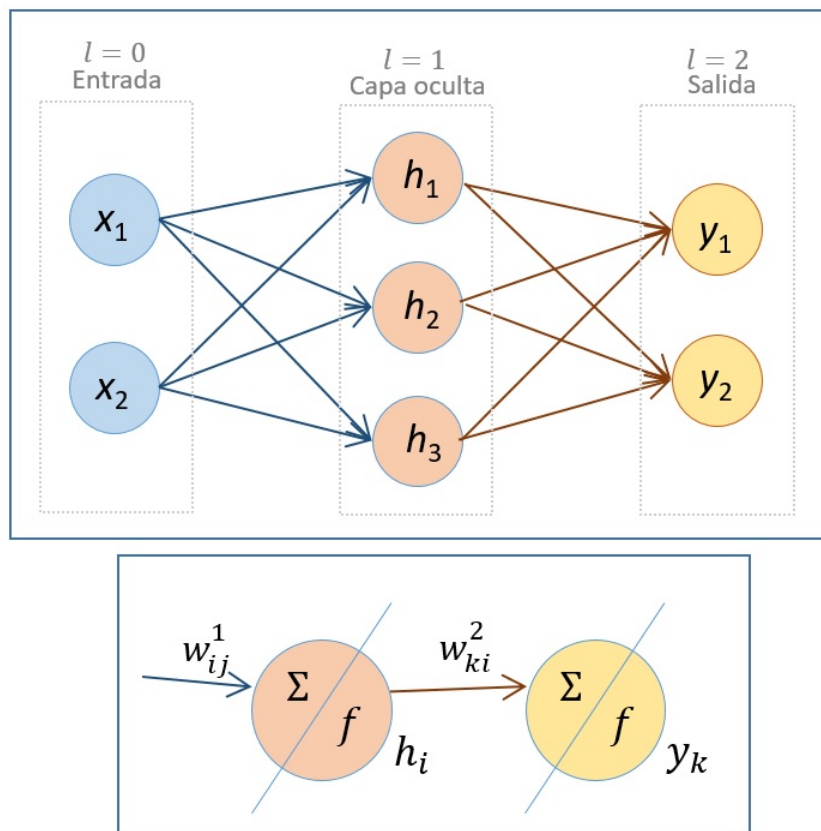
- $\frac{\partial J}{\partial x_1} \rightarrow \frac{\partial J}{\partial y} \frac{\partial y}{\partial x_1}, \quad \frac{\partial y}{\partial x_1} = a_1$
- $\frac{\partial J}{\partial x_2} \rightarrow \frac{\partial J}{\partial y} \frac{\partial y}{\partial x_2}, \quad \frac{\partial y}{\partial x_2} = a_2$

This page titled [7: Retropropagación](#) is shared under a [not declared](#) license and was authored, remixed, and/or curated by [Joaquín López Herraiz](#).

8: Entrenamiento de una Red Neuronal

This page is a draft and is under active development.

En lo que sigue, vamos a analizar el proceso de entrenamiento en una RNA de *dos capas* (ver figura 3). La red consta de una capa de entrada (correspondiente a $l = 0$, con dos neuronas), una capa intermedia *oculta* (correspondiente a $l = 1$, con tres neuronas) y la capa de salida ($l = 2$ y dos neuronas). El objetivo es calcular los pesos de la RNA, utilizando el algoritmo de retropropagación de la siguiente forma:



Ejemplo de red neuronal

- Las entradas (en $l = 0$), $\mathbf{x} = (x_1, x_2)^T$ son valores constantes. Vamos a denotar por h_i la salida de la neurona i de la capa $l = 1$, que serán a su vez entradas de la capa $l = 2$. Análogamente, denotamos por y_1 e y_2 las salidas de la capa $l = 2$ (salidas de la RNA).
- Los parámetros que debemos ajustar son los pesos (enlaces entre neuronas de diferentes capas). Se denotan por w_{ij}^l , donde el superíndice l hace referencia a la capa a la que llega y los subíndices i y j a las neuronas de las capas l y $l - 1$, respectivamente, que conecta este peso.
- En las neuronas de la capa oculta y de la capa de salida se realizan dos procesos:
 - El primero, que simbolizamos con Σ , corresponde en ambas capas a la combinación lineal de la salida de la capa anterior con los pesos correspondientes. Esto es:
 - En la neurona i ($i = 1, 2, 3$) de la capa $l = 1$: $a_i^1 = x_1 w_{i1}^1 + x_2 w_{i2}^1$.
 - En la neurona k ($k = 1, 2$) de la capa $l = 2$: $a_k^2 = h_1 w_{k1}^2 + h_2 w_{k2}^2 + h_3 w_{k3}^2$.
 - El segundo, que representamos en la figura con f , corresponde a la función de activación que suponemos que es la misma para $l = 1, 2$:

$$\sigma(u) = \frac{1}{1 + e^{-u}} \quad (8.1)$$

Recordemos que la derivada de esta función es:

$$\sigma'(u) = \frac{e^{-u}}{(1 + e^{-u})^2} = \sigma(u)(1 - \sigma(u)) \quad (8.2)$$

Entonces, tenemos que las salidas de cada capa son:

- $h_i = \sigma(a_i^1), i = 1, 2, 3$
- $y_k = \sigma(a_k^2), k = 1, 2$

- Definimos la función de pérdida o de error, como se hace usualmente, mediante la desviación cuadrática media

$$L(y_1, y_2) = \frac{1}{2} \sum_{k=1}^2 (y_k - t_k)^2 \quad (8.3)$$

siendo t_k ($k = 1, 2$) los valores deseados correspondientes a cada salida y_k de la RNA.

Puesto que nuestro objetivo es minimizar la función de pérdida, debemos buscar los pesos que lo consiguen, utilizando el algoritmo de retropropagación. Se trata de un proceso iterativo en el que se parte de unos valores iniciales para los pesos y estos se van modificando mediante el método del gradiente descendente. Con los valores iniciales de los pesos, a partir de los valores \mathbf{x} se calculan los de $\mathbf{a}^1, \mathbf{h}, \mathbf{a}^2$ e \mathbf{y} . Con ellos, partiendo de la capa de salida, vamos hacia atrás:

1. Aplicamos la regla de la cadena

$$\frac{\partial L}{\partial w_{ki}^2} \rightarrow \frac{\partial L}{\partial y_k} \frac{\partial y_k}{\partial a_k^2} \frac{\partial a_k^2}{\partial w_{ki}^2} \quad (8.4)$$

con ($k = 1, 2$):

- $\frac{\partial}{\partial w_{ki}^2} a_k^2(w_{k1}^2, w_{k2}^2, w_{k3}^2) = h_i \quad (i = 1, 2, 3)$
- $\frac{\partial y_k}{\partial a_k^2} = \frac{\partial}{\partial a_k^2} \sigma(a_k^2) = \sigma(a_k^2)(1 - \sigma(a_k^2)) = y_k(1 - y_k)$
- $\frac{\partial}{\partial y_k} L(y_1, y_2) = (y_k - t_k)$

Resulta, sustituyendo:

$$\frac{\partial L}{\partial w_{ki}^2} \rightarrow (y_k - t_k) y_k(1 - y_k) h_i \quad (8.5)$$

2. Siguiendo el método del gradiente descendente, actualizamos los pesos w_{ki}^2 en esta capa. Llamando $\mathbf{w}^2 = (w_{ki}^2)_{k=1,2;i=1,2,3}$

$$\mathbf{w}^2 \leftarrow \mathbf{w}^2 - \varepsilon \nabla_{\mathbf{w}^2} L \quad (8.6)$$

3. Aplicamos la regla de la cadena en la capa oculta:

$$\frac{\partial L}{\partial w_{ij}^1} \rightarrow \frac{\partial L}{\partial h_i} \frac{\partial h_i}{\partial a_i^1} \frac{\partial a_i^1}{\partial w_{ij}^1} \quad (8.7)$$

con ($i = 1, 2, 3$):

- $\frac{\partial}{\partial w_{ij}^1} a_i^1(w_{i1}^1, w_{i2}^1) = x_j \quad (j = 1, 2)$
- $\frac{\partial h_i}{\partial a_i^1} = \frac{\partial}{\partial a_i^1} \sigma(a_i^1) = \sigma(a_i^1)(1 - \sigma(a_i^1)) = h_i(1 - h_i)$
- Para calcular $\frac{\partial L}{\partial h_i}$ tenemos en cuenta la dependencia de las funciones respecto a las distintas variables: $L(y_1, y_2), y_k(a_k^2)$ y $a_k^2(h_1, h_2, h_3)$. Así, aplicando la regla de la cadena:

$$\frac{\partial L}{\partial h_i} \rightarrow \frac{\partial L}{\partial y_1} \frac{\partial y_1}{\partial a_1^2} \frac{\partial a_1^2}{\partial h_i} + \frac{\partial L}{\partial y_2} \frac{\partial y_2}{\partial a_2^2} \frac{\partial a_2^2}{\partial h_i} \quad (8.8)$$

Ya habíamos obtenido anteriormente las expresiones para $\frac{\partial L}{\partial y_k}$ y $\frac{\partial y_k}{\partial a_k^2}$, por lo que sólo queda calcular el último factor de cada sumando:

$$\blacksquare \frac{\partial a_2^k}{\partial h_i} = w_{ki}^2$$

Sustituyendo:

$$\frac{\partial L}{\partial w_{ij}^1} \rightarrow x_j h_i (1 - h_i) \sum_{k=1}^2 [(y_k - t_k) y_k (1 - y_k) w_{ki}^2] \quad (8.9)$$

4. Finalmente, se actualizan los pesos w_{ij}^1 . Llamando $\mathbf{w}^1 = (w_{ij}^1)_{i=1,2,3; j=1,2}$

$$\mathbf{w}^1 \leftarrow \mathbf{w}^1 - \varepsilon \nabla_{\mathbf{w}^1} L \quad (8.10)$$

Con esto se ha dado un paso de la iteración de la actualización de los pesos. A continuación, se volvería a repetir el proceso partiendo de los valores de los datos \mathbf{x} .

En la práctica se suele contar para el entrenamiento con un conjunto de datos (no solo uno) y la función que se minimiza entonces es la suma de todas las “pérdidas” que se tienen al considerar todos esos datos.

This page titled [8: Entrenamiento de una Red Neuronal](#) is shared under a [not declared](#) license and was authored, remixed, and/or curated by [Joaquín López Herraiz](#).

CHAPTER OVERVIEW

Apéndices

[Índice](#)

[Glosario](#)

This page titled [Apéndices](#) is shared under a [not declared](#) license and was authored, remixed, and/or curated by [Joaquín López Herraiz](#).

Índice

C

composicion de funciones

3: Composición de Funciones. Regla de la Cadena

R

regla de la cadena

3: Composición de Funciones. Regla de la Cadena

This page titled [Índice](#) is shared under a [not declared](#) license and was authored, remixed, and/or curated by [Joaquín López Herraiz](#).

Glosario

Sample Word 1 Sample Definition 1	This page titled Glosario is shared under a not declared license and was authored, remixed,
--	---

[Index](#)

C

[composicion de funciones](#)

3: [Composición de Funciones](#), [Regla de la Cadena](#)

R

[regla de la cadena](#)

3: [Composición de Funciones](#), [Regla de la Cadena](#)

Glossary

Sample Word 1 | Sample Definition 1

Detailed Licensing

Overview

Title: [Las matemáticas de la inteligencia artificial](#)

Webpages: 32

All licenses found:

- [Undeclared](#): 100% (32 pages)

By Page

- [Las matemáticas de la inteligencia artificial](#) - *Undeclared*
 - [Front Matter](#) - *Undeclared*
 - [TitlePage](#) - *Undeclared*
 - [InfoPage](#) - *Undeclared*
 - [Table of Contents](#) - *Undeclared*
 - [Licensing](#) - *Undeclared*
 - [Texto Preliminar](#) - *Undeclared*
 - [Página del Título](#) - *Undeclared*
 - [Página de Información](#) - *Undeclared*
 - [Tabla de Contenido](#) - *Undeclared*
 - [1: Introducción a la Inteligencia Artificial y las Redes Neuronales](#) - *Undeclared*
 - [1.1: Introducción General](#) - *Undeclared*
 - [1.2: Perspectiva Histórica](#) - *Undeclared*
 - [1.3: Aplicaciones presentes y futuras](#) - *Undeclared*
 - [1.4: Objetivos del Curso](#) - *Undeclared*
 - [2: Introducción al Cálculo Multivariante](#) - *Undeclared*
 - [3: Composición de Funciones. Regla de la Cadena](#) - *Undeclared*
 - [4: Aproximación Lineal de una Función Escalar de Varias Variables](#) - *Undeclared*
 - [5: Optimización y Método del Gradiente Descendiente](#) - *Undeclared*
 - [6: Redes Neuronales](#) - *Undeclared*
 - [6.1: Perceptrón](#) - *Undeclared*
 - [6.2: Funciones de Pérdida](#) - *Undeclared*
 - [6.3: Perceptrón Multicapa](#) - *Undeclared*
 - [7: Retropropagación](#) - *Undeclared*
 - [8: Entrenamiento de una Red Neuronal](#) - *Undeclared*
 - [Apéndices](#) - *Undeclared*
 - [Índice](#) - *Undeclared*
 - [Glosario](#) - *Undeclared*
 - [Back Matter](#) - *Undeclared*
 - [Index](#) - *Undeclared*
 - [Glossary](#) - *Undeclared*
 - [Detailed Licensing](#) - *Undeclared*